Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques

APPROX/RANDOM 2019, September 20–22, 2019,
Massachusetts Institute of Technology, Cambridge, MA, USA

Edited by

Dimitris Achlioptas

László A. Végh



Editors

Dimitris Achlioptas
UC Santa Cruz, California, USA
optas@soe.ucsc.edu

László A. Végh
London School of Economics and Political Science, London, UK
L.Vegh@lse.ac.uk

ACM Classification 2012
Mathematics of computing; Theory of computation

ISBN 978-3-95977-125-2

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at https://www.dagstuhl.de/dagpub/978-3-95977-125-2.

Publication date
September, 2019

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at https://portal.dnb.de.

License
This work is licensed under a Creative Commons Attribution 3.0 Unported license (CC-BY 3.0):
https://creativecommons.org/licenses/by/3.0/legalcode.
In brief, this license authorizes each and everybody to share (to copy, distribute and transmit) the work under the following conditions, without impairing or restricting the authors' moral rights:
- Attribution: The work must be attributed to its authors.

The copyright is retained by the corresponding authors.

Digital Object Identifier: 10.4230/LIPIcs.APPROX-RANDOM.2019.0

LIPIcs – Leibniz International Proceedings in Informatics

LIPIcs is a series of high-quality conference proceedings across all fields in informatics. LIPIcs volumes are published according to the principle of Open Access, i.e., they are available online and free of charge.

Editorial Board

- Luca Aceto (*Chair*, Gran Sasso Science Institute and Reykjavik University)
- Christel Baier (TU Dresden)
- Mikolaj Bojanczyk (University of Warsaw)
- Roberto Di Cosmo (INRIA and University Paris Diderot)
- Javier Esparza (TU München)
- Meena Mahajan (Institute of Mathematical Sciences)
- Dieter van Melkebeek (University of Wisconsin-Madison)
- Anca Muscholl (University Bordeaux)
- Luke Ong (University of Oxford)
- Catuscia Palamidessi (INRIA)
- Thomas Schwentick (TU Dortmund)
- Raimund Seidel (Saarland University and Schloss Dagstuhl – Leibniz-Zentrum für Informatik)

ISSN 1868-8969

https://www.dagstuhl.de/lipics

Contents

Preface
Dimitris Achlioptas and László A. Végh 0:xi

APPROX

The Query Complexity of Mastermind with ℓ_p Distances
Manuel Fernández V, David P. Woodruff, and Taisuke Yasuda 1:1–1:11

Tracking the ℓ_2 Norm with Constant Update Time
Chi-Ning Chou, Zhixian Lei, and Preetum Nakkiran 2:1–2:15

Submodular Optimization with Contention Resolution Extensions
Benjamin Moseley and Maxim Sviridenko 3:1–3:17

Prepare for the Expected Worst: Algorithms for Reconfigurable Resources Under Uncertainty
David Ellis Hershkowitz, R. Ravi, and Sahil Singla 4:1–4:19

Streaming Hardness of Unique Games
Venkatesan Guruswami and Runzhou Tao 5:1–5:12

On Strong Diameter Padded Decompositions
Arnold Filtser 6:1–6:21

Max-Min Greedy Matching
Alon Eden, Uriel Feige, and Michal Feldman 7:1–7:23

Hardy-Muckenhoupt Bounds for Laplacian Eigenvalues
Gary L. Miller, Noel J. Walkington, and Alex L. Wang 8:1–8:19

Improved 3LIN Hardness via Linear Label Cover
Prahladh Harsha, Subhash Khot, Euiwoong Lee, and Devanathan Thiruvenkatachari 9:1–9:16

Dynamic Pricing of Servers on Trees
Ilan Reuven Cohen, Alon Eden, Amos Fiat, and Łukasz Jeż 10:1–10:22

Approximating the Norms of Graph Spanners
Eden Chlamtáč, Michael Dinitz, and Thomas Robinson 11:1–11:22

Conditional Hardness of Earth Mover Distance
Dhruv Rohatgi 12:1–12:17

Single-Elimination Brackets Fail to Approximate Copeland Winner
Reyna Hulett 13:1–13:20

Routing Symmetric Demands in Directed Minor-Free Graphs with Constant Congestion
Timothy Carpenter, Ario Salmasi, and Anastasios Sidiropoulos 14:1–14:15

Rainbow Coloring Hardness via Low Sensitivity Polymorphisms
Venkatesan Guruswami and Sai Sandeep 15:1–15:17

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Syntactic Separation of Subset Satisfiability Problems
Eric Allender, Martín Farach-Colton, and Meng-Tsung Tsai 16:1–16:23

Malleable Scheduling Beyond Identical Machines
Dimitris Fotakis, Jannik Matuschke, and Orestis Papadigenopoulos 17:1–17:14

On the Cost of Essentially Fair Clusterings
Ioana O. Bercea, Martin Groß, Samir Khuller, Aounon Kumar, Clemens Rösner, Daniel R. Schmidt, and Melanie Schmidt 18:1–18:22

The Maximum Exposure Problem
Neeraj Kumar, Stavros Sintos, and Subhash Suri 19:1–19:20

Small Space Stream Summary for Matroid Center
Sagar Kale 20:1–20:22

Improved Bounds for Open Online Dial-a-Ride on the Line
Alexander Birx, Yann Disser, and Kevin Schewior 21:1–21:22

Improved Online Algorithms for Knapsack and GAP in the Random Order Model
Susanne Albers, Arindam Khan, and Leon Ladewig 22:1–22:23

Fast and Deterministic Approximations for k-Cut
Kent Quanrud 23:1–23:20

Global Cardinality Constraints Make Approximating Some Max-2-CSPs Harder
Per Austrin and Aleksa Stanković 24:1–24:17

Robust Appointment Scheduling with Heterogeneous Costs
Andreas S. Schulz and Rajan Udwani 25:1–25:17

Collapsing Superstring Conjecture
Alexander Golovnev, Alexander S. Kulikov, Alexander Logunov, Ivan Mihajlin, and Maksim Nikolaev 26:1–26:23

Improved Algorithms for Time Decay Streams
Vladimir Braverman, Harry Lang, Enayat Ullah, and Samson Zhou 27:1–27:17

Approximation Algorithms for Partially Colorable Graphs
Suprovat Ghoshal, Anand Louis, and Rahul Raychaudhury 28:1–28:20

Towards Optimal Moment Estimation in Streaming and Distributed Models
Rajesh Jayaram and David P. Woodruff 29:1–29:21

The Complexity of Partial Function Extension for Coverage Functions
Umang Bhaskar and Gunjan Kumar 30:1–30:21

Almost Optimal Classical Approximation Algorithms for a Quantum Generalization of Max-Cut
Sevag Gharibian and Ojas Parekh 31:1–31:17

Maximizing Covered Area in the Euclidean Plane with Connectivity Constraint
Chien-Chung Huang, Mathieu Mari, Claire Mathieu, Joseph S. B. Mitchell, and Nabil H. Mustafa 32:1–32:21

Robust Correlation Clustering
Devvrit, Ravishankar Krishnaswamy, and Nived Rajaraman 33:1–33:18

RANDOM

Counting Independent Sets and Colorings on Random Regular Bipartite Graphs
Chao Liao, Jiabao Lin, Pinyan Lu, and Zhenyu Mao 34:1–34:12

The Expected Number of Maximal Points of the Convolution of Two 2-D Distributions
Josep Diaz and Mordecai Golin 35:1–35:14

On a Connectivity Threshold for Colorings of Random Graphs and Hypergraphs
Michael Anastos and Alan Frieze 36:1–36:10

Slow Mixing of Glauber Dynamics for the Six-Vertex Model in the Ordered Phases
Matthew Fahrbach and Dana Randall 37:1–37:20

Lifted Multiplicity Codes and the Disjoint Repair Group Property
Ray Li and Mary Wootters 38:1–38:18

Think Globally, Act Locally: On the Optimal Seeding for Nonsubmodular Influence Maximization
Grant Schoenebeck, Biaoshuai Tao, and Fang-Yi Yu 39:1–39:20

Direct Sum Testing: The General Case
Irit Dinur and Konstantin Golubev 40:1–40:11

Fast Algorithms at Low Temperatures via Markov Chains
Zongchen Chen, Andreas Galanis, Leslie Ann Goldberg, Will Perkins, James Stewart, and Eric Vigoda 41:1–41:14

Deterministic Approximation of Random Walks in Small Space
Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan 42:1–42:22

Two-Source Condensers with Low Error and Small Entropy Gap via Entropy-Resilient Functions
Avraham Ben-Aroya, Gil Cohen, Dean Doron, and Amnon Ta-Shma 43:1–43:20

Efficient Average-Case Population Recovery in the Presence of Insertions and Deletions
Frank Ban, Xi Chen, Rocco A. Servedio, and Sandip Sinha 44:1–44:18

Improved Pseudorandom Generators from Pseudorandom Multi-Switching Lemmas
Rocco A. Servedio and Li-Yang Tan 45:1–45:23

Unconstraining Graph-Constrained Group Testing
Bruce Spang and Mary Wootters 46:1–46:20

Near-Neighbor Preserving Dimension Reduction for Doubling Subsets of ℓ_1
Ioannis Z. Emiris, Vasilis Margonis, and Ioannis Psarros 47:1–47:13

Improved Strong Spatial Mixing for Colorings on Trees
Charilaos Efthymiou, Andreas Galanis, Thomas P. Hayes, Daniel Štefankovič, and Eric Vigoda 48:1–48:16

(Near) Optimal Adaptivity Gaps for Stochastic Multi-Value Probing
Domagoj Bradac, Sahil Singla, and Goran Zuzic 49:1–49:21

Testing Odd Direct Sums Using High Dimensional Expanders
Roy Gotlib and Tali Kaufman 50:1–50:20

A Lower Bound for Sampling Disjoint Sets
Mika Göös and Thomas Watson 51:1–51:13

Approximating the Noise Sensitivity of a Monotone Boolean Function
Ronitt Rubinfeld and Arsen Vasilyan 52:1–52:17

Connectivity of Random Annulus Graphs and the Geometric Block Model
Sainyam Galhotra, Arya Mazumdar, Soumyabrata Pal, and Barna Saha 53:1–53:23

A Local Stochastic Algorithm for Separation in Heterogeneous Self-Organizing Particle Systems
Sarah Cannon, Joshua J. Daymude, Cem Gökmen, Dana Randall, and Andréa W. Richa 54:1–54:22

The Large-Error Approximate Degree of AC^0
Mark Bun and Justin Thaler 55:1–55:16

String Matching: Communication, Circuits, and Learning
Alexander Golovnev, Mika Göös, Daniel Reichman, and Igor Shinkar 56:1–56:20

Efficient Black-Box Identity Testing for Free Group Algebras
V. Arvind, Abhranil Chatterjee, Rajit Datta, and Partha Mukhopadhyay 57:1–57:16

The Maximum Label Propagation Algorithm on Sparse Random Graphs
Charlotte Knierim, Johannes Lengler, Pascal Pfister, Ulysse Schaller, and Angelika Steger 58:1–58:15

Samplers and Extractors for Unbounded Functions
Rohit Agrawal 59:1–59:21

Successive Minimum Spanning Trees
Svante Janson and Gregory B. Sorkin 60:1–60:16

Simple Analysis of Sparse, Sign-Consistent JL
Meena Jagadeesan 61:1–61:20

Streaming Coreset Constructions for M-Estimators
Vladimir Braverman, Dan Feldman, Harry Lang, and Daniela Rus 62:1–62:15

Pairwise Independent Random Walks Can Be Slightly Unbounded
Shyam Narayanan 63:1–63:19

Optimal Convergence Rate of Hamiltonian Monte Carlo for Strongly Logconcave Distributions
Zongchen Chen and Santosh S. Vempala 64:1–64:12

Exploring Differential Obliviousness
Amos Beimel, Kobbi Nissim, and Mohammad Zaheri 65:1–65:20

Thresholds in Random Motif Graphs
Michael Anastos, Peleg Michaeli, and Samantha Petti 66:1–66:19

Random-Cluster Dynamics in $\mathbb{Z}^2$: Rapid Mixing with General Boundary Conditions
Antonio Blanca, Reza Gheissari, and Eric Vigoda 67:1–67:19

On List Recovery of High-Rate Tensor Codes
Swastik Kopparty, Nicolas Resch, Noga Ron-Zewi, Shubhangi Saraf, and Shashwat Silas 68:1–68:22

Approximate $\mathbb{F}_2$-Sketching of Valuation Functions
Grigory Yaroslavtsev and Samson Zhou 69:1–69:21

Streaming Verification of Graph Computations via Graph Structure
Amit Chakrabarti and Prantar Ghosh 70:1–70:20

Approximate Degree, Secret Sharing, and Concentration Phenomena
Andrej Bogdanov, Nikhil S. Mande, Justin Thaler, and Christopher Williamson .. 71:1–71:21

Improved Extractors for Recognizable and Algebraic Sources
Fu Li and David Zuckerman 72:1–72:22

Preface

This volume contains the papers presented at the 22nd International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'2019) and the 23rd International Conference on Randomization and Computation (RANDOM'2019), which took place concurrently at the at Massachusetts Institute of Technology, USA during September 20–22, 2019.

APPROX focuses on algorithmic and complexity issues surrounding the development of efficient approximate solutions to computationally difficult problems, and was the 22nd in the series. RANDOM is concerned with applications of randomness to computational and combinatorial problems, and was the 23rd in the series. Prior to 2003, APPROX took place in Aalborg (1998), Berkeley (1999), Saarbrücken (2000), Berkeley (2001), and Rome (2002), while RANDOM took place in Bologna (1997), Barcelona (1998), Berkeley (1999), Geneva (2000), Berkeley (2001), and Harvard (2002). Since 2003, APPROX and RANDOM have been collocated, taking place in Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), Princeton (2011), Boston (2012), Berkeley (2013), Barcelona (2014), Princeton (2015), Paris (2016), Berkeley (2017), and Princeton (2018).

Topics of interest for APPROX and RANDOM are: approximation algorithms, hardness of approximation, small space, sub-linear time and streaming algorithms, online algorithms, approaches that go beyond worst case analysis, distributed and parallel approximation, embeddings and metric space methods, mathematical programming methods, spectral methods, combinatorial optimization, algorithmic game theory, mechanism design and economics, computational geometric problems, approximate learning, design and analysis of randomized algorithms, randomized complexity theory, pseudorandomness and derandomization, random combinatorial structures, random walks/Markov chains, expander graphs and randomness extractors, probabilistic proof systems, random projections and embeddings, error-correcting codes, average-case analysis, smoothed analysis, property testing, and computational learning theory.

The volume contains 33 contributed papers, selected by the APPROX Program Committee out of 66 submissions, and 39 contributed papers, selected by the RANDOM Program Committee also out of 66 submissions. We would like to thank all of the authors who submitted papers, the invited speakers, the members of the Program Committees, and the external reviewers. We are grateful for the guidance of the steering committees: Klaus Jansen, Samir Khuller, and Monaldo Mastrolili for APPROX, and Oded Goldreich, Cris Moore, Anup Rao, Omer Reingold, Dana Ron, Ronitt Rubinfeld, Amit Sahai, Ronen Shaltiel, Alistair Sinclair, and Paul Spirakis for RANDOM.

September 2019

Dimitris Achlioptas and László A. Végh

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Program Committees

APPROX

Nima Anari	Stanford University
Kristóf Bérczi	Eötvös University, Budapest
Deeparnab Chakrabarty	Dartmouth College
Karthekeyan Chandrasekaran	University of Illinois, Urbana-Champaign
Michael Dinitz	Johns Hopkins University
Leah Epstein	University of Haifa
Samuel Fiorini	Université libre de Bruxelles
Swati Gupta	Georgia Institute of Technology
Bundit Laekhanukit	Shanghai University of Finance and Economics
Joseph Seffi Naor	Technion
Huy Lê Nguyễn	Northeastern University
Kanstantsin Pashkovich	University of Ottawa
Barna Saha	University of Massachusetts Amherst
Bruce Shepherd	University of British Columbia
David B. Shmoys	Cornell University
He Sun	University of Edinburgh
László A. Végh (chair)	London School of Economics and Political Science

RANDOM

Dimitris Achlioptas (chair)	University of California Santa Cruz/Google
Nikhil Bansal	Eindhoven/Centrum Wiskunde & Informatica
Paul Beame	University of Washington
Ivona Bezakova	Rochester Institute of Technology
Klim Efremenko	Ben Gurion University
Uri Feige	Weizmann Institute of Science
Anna Gilbert	University of Michigan
Subhash Khot	New York University
Antonina Kolokova	Memorial University of Newfoundland
Ravi Kumar	Google
Or Meir	University of Haifa
Prasad Raghavendra	University of California Berkeley
Noga Ron-Zewi	University of Haifa
Sofya Raskhodnikova	Boston University
C. Seshadhri	University of California Santa Cruz
Devavrat Shah	Massachusetts Institute of Technology
Christian Sohler	Technical University of Dortmund/Google
Kunal Talwar	Google
Thomas Vidick	California Institute of Technology
Jan Vondrak	Stanford University
David Woodruff	Carnegie Mellon University

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Subreviewers

Raghavendra Addanki
David Adjiashvili
Saba Ahmadi
Matthew Aldridge
Zeyuan Allen-Zhu
Itai Arad
Sepehr Assadi
Ainesh Bakshi
Coulter Beeson
Amos Beimel
Erika Bérczi-Kovács
Amey Bhangale
Marcin Bienkowski
Jaroslaw Blasiok
Andrej Bogdanov
Jakub Bulín
Mark Bun
Gruia Calinescu
Bastien Cazaux
Deepayan Chakrabarti
Parinya Chalermsook
Hsien-Chih Chang
Arkadev Chattopadhyay
Eshan Chattopadhyay
Lin Chen
Eden Chlamtáč
Gil Cohen
Daniel Dadush
Anindya De
Ilias Diakonikolas
Kashyap Dixit
Benjamin Doerr
Carola Doerr
Dean Doron
Talya Eden
Ahmed El Alaoui
Hossein Esfandiari
Yaron Fairstein
Vitaly Feldman
Larkin Flodin
Kyle Fox
Tom Friedetzky
Alan Frieze
Mehrdad Ghadiri
Prantar Ghosh
Dion Gijswijt
Sasha Golovnev
David Gosset
Vineet Goyal
Catherine Greenhill
Benoit Groz
Heng Guo
Guru Guruganesh
Nick Harvey
Jan Hladky
Pavel Hrubes
Zhiyi Huang
Sungjin Im
Nikita Ivkin
Klaus Jansen
Rajesh Jayaram
T.S. Jayram
Matthew Jenssen
Shaofeng Jiang
John Kallaugher
Pritish Kamath
Michael Kapralov
Thomas Kesselheim
Sanjeev Khanna
Guy Kindler
Tamás Király
Ilan Komargodski
Swastik Kopparty
Guy Kortsarz
Michal Koucky
Grigorios Koumoutsos
Lukasz Kowalik
Andrei Krokhin
Janardhan Kulkarni
Marvin Künnemann
Adam Kurpisz
Hung Le
Euiwoong Lee
Reut Levi
Xin Li
Jason Li
Andre Linhares
Michael Litvak
Tianyu Liu
Kuikui Liu

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Zephyr Lucas
Vivek Madan
Péter Madarasi
Konstantin Makarychev
Yury Makarychev
Pasin Manurangsi
Jannik Matuschke
Colin McDiarmid
Andrew McGregor
Raghu Meka
Dor Minzer
Matthias Mnich
Sidhanth Mohanty
Michael Molloy
Dana Moshkovitz
Marcin Mucha
Wolfgang Mulzer
Cameron Musco
Christopher Musco
Viswanah Nagarajan
Yasamin Nazari
Maryam Negahbani
Alantha Newman
André Nusser
Zeev Nutov
Izhar Oppenheim
Ramesh Krishnan S. Pallavoor
Katarzyna Paluch
Dömötör Pálvölgyi
Denis Pankratov
Pan Peng
Will Perkins
Manish Purohit
Kent Quanrud
Shijin Rajakrishnan
Oded Regev
Robert Robere
Dana Ron
Atri Rudra
Sushant Sachdeva
Rishi Saket
Mohammad Salavatipour
Laura Sanita
Rahul Santhanam
Richard Santiago
Jonathan Scarlett
Frans Schalekamp
Aaron Schild
Melanie Schmidt
Grant Schoenebeck
Sahil Singla
D Sivakumar
Noah Stephens-Davidowitz
Sebastian Stiller
Warut Suksompong
Ohad Talmon
Li-Yang Tan
Justin Thaler
Vera Traub
Luca Trevisan
Madhur Tulsiani
Frank Vallentin
Kasturi Varadarajan
Nithin Varma
Sergei Vassilvitskii
Ben Lee Volk
Hoa Vu
David Wajc
Yipu Wang
Justin Ward
Omri Weinstein
Andreas Wiese
Christopher Williamson
David Williamson
Peter Winkler
Anthony Wirth
Eitan Yaakobi
Hiroki Yanagisawa
Kostas Zampetakis
Luca Zanetti
Samson Zhou
Stanislav Živný

List of Authors

Rohit Agrawal (59)
John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138, USA

Susanne Albers (22)
Technical University of Munich, Germany

Eric Allender (16)
Rutgers University, Piscataway, NJ 08854, USA

Michael Anastos (36, 66)
Carnegie Mellon University, Pittsburgh PA 15213, USA

V. Arvind (57)
Institute of Mathematical Sciences (HBNI), Chennai, India

Per Austrin (24)
KTH Royal Institute of Technology, Stockholm, Sweden

Frank Ban (44)
UC Berkeley, Berkeley, CA, USA

Amos Beimel (65)
Dept. of Computer Science, Ben-Gurion University, Israel

Avraham Ben-Aroya (43)
The Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv, Israel

Ioana O. Bercea (18)
School of Electrical Engineering, Tel-Aviv University, Israel

Umang Bhaskar (30)
Tata Institute of Fundamental Research, Mumbai, India

Alexander Birx (21)
Institute of Mathematics and Graduate School CE, TU Darmstadt, Germany

Antonio Blanca (67)
Department of Computer Science and Engineering, Pennsylvania State University, USA

Andrej Bogdanov (71)
Department of Computer Science and Engineering, Chinese University of Hong Kong; Institute for Theoretical Computer Science and Communications, Hong Kong

Domagoj Bradac (49)
Department of Mathematics, Faculty of Science, University of Zagreb, Croatia

Vladimir Braverman (27, 62)
Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA

Mark Bun (55)
Boston University, Boston, MA, USA

Sarah Cannon (54)
Claremont McKenna College, Claremont, CA, USA

Timothy Carpenter (14)
Dept. of Computer Science & Engineering, The Ohio State University, Columbus, OH, USA

Amit Chakrabarti (70)
Dartmouth College, Hanover, NH, USA

Abhranil Chatterjee (57)
Institute of Mathematical Sciences (HBNI), Chennai, India

Xi Chen (44)
Columbia University, New York, NY, USA

Zongchen Chen (41, 64)
School of Computer Science, Georgia Institute of Technology, Atlanta, USA

Eden Chlamtáč (11)
Ben Gurion University of the Negev, Beersheva, Israel

Chi-Ning Chou (2)
School of Engineering and Applied Sciences, Harvard University, Cambridge, Massachusetts, USA

Gil Cohen (43)
The Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv, Israel

Ilan Reuven Cohen (10)
TU Eindhoven, The Netherlands; CWI, Amsterdam, The Netherlands

Rajit Datta (57)
Chennai Mathematical Institute, Chennai, India

Joshua J. Daymude (54)
Computer Science, CIDSE, Arizona State University, Tempe, AZ, USA

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Devvrit (33)
BITS Pilani, Goa Campus, Goa, India

Josep Diaz (35)
Department of CS, UPC, Barcelona, Spain

Michael Dinitz (11)
Johns Hopkins University, Baltimore, MD, USA

Irit Dinur (40)
The Weizmann Institute of Science,
Rehovot, Israel

Yann Disser (21)
Institute of Mathematics,
TU Darmstadt, Germany

Dean Doron (43)
Department of Computer Science,
University of Texas at Austin, USA

Alon Eden (7, 10)
Tel-Aviv University, Israel

Charilaos Efthymiou (48)
Department of Computer Science,
University of Warwick, UK

Ioannis Z. Emiris (47)
Department of Informatics &
Telecommunications, National & Kapodistrian
University of Athens, Greece;
ATHENA Research & Innovation Center, Greece

Matthew Fahrbach (37)
School of Computer Science,
Georgia Institute of Technology, Atlanta,
Georgia, USA

Martín Farach-Colton (16)
Rutgers University, Piscataway, NJ 08854, USA

Uriel Feige (7)
Weizmann Institute of Science, Rehovot, Israel

Dan Feldman (62)
Department of Computer Science,
University of Haifa, Israel

Michal Feldman (7)
Tel Aviv University, Israel;
Microsoft Research, Herzlyia, Israel

Manuel Fernández V (1)
Computer Science Department,
Carnegie Mellon University,
Pittsburgh, Pennsylvania, USA

Amos Fiat (10)
Tel Aviv University, Israel

Arnold Filtser (6)
Ben Gurion University of the Negev,
Beersheva, Israel

Dimitris Fotakis (17)
School of Electrical and Computer Engineering,
National Technical University of Athens, Greece

Alan Frieze (36)
Carnegie Mellon University,
Pittsburgh PA 15213, USA

Andreas Galanis (41, 48)
Department of Computer Science,
University of Oxford, Oxford, UK

Sainyam Galhotra (53)
University of Massachusetts Amherst, USA

Sevag Gharibian (31)
University of Paderborn, Germany;
Virginia Commonwealth University,
Richmond, VA, USA

Reza Gheissari (67)
Courant Institute of Mathematical Sciences,
New York University, USA

Prantar Ghosh (70)
Dartmouth College, Hanover, NH, USA

Suprovat Ghoshal (28)
Indian Institute of Science, Bangalore, India

Leslie Ann Goldberg (41)
Department of Computer Science,
University of Oxford, Oxford, UK

Mordecai Golin (35)
CSE Department, Hong Kong UST

Alexander Golovnev (26, 56)
Harvard University, Cambridge, MA, USA

Konstantin Golubev (40)
D-MATH, ETH Zurich, Switzerland

Roy Gotlib (50)
Bar-Ilan University, Ramat Gan, Israel

Martin Groß (18)
School of Business and Economics,
RWTH Aachen, Germany

Venkatesan Guruswami (5, 15)
Computer Science Department,
Carnegie Mellon University,
5000 Forbes Ave, Pittsburgh, PA, USA, 15213

Cem Gökmen (54)
Georgia Institute of Technology,
Atlanta, GA, USA

Mika Göös (51, 56)
Institute for Advanced Study,
Princeton, NJ, USA

Prahladh Harsha (9)
School of Technology and Computer Science,
Tata Institute of Fundamental Research,
Mumbai, India

Thomas P. Hayes (48)
Department of Computer Science,
University of New Mexico,
Albuquerque, NM, USA

David Ellis Hershkowitz (4)
Carnegie Mellon University,
Pittsburgh, PA, USA

Chien-Chung Huang (32)
DI ENS, École Normale supérieure,
Université PSL, Paris, France;
CNRS, Paris, France

Reyna Hulett (13)
Department of Computer Science,
Stanford University, CA, USA

Meena Jagadeesan (61)
Harvard University,
Cambridge, Massachusetts, USA

Svante Janson (60)
Department of Mathematics, Uppsala University,
PO Box 480, SE-751 06 Uppsala, Sweden

Rajesh Jayaram (29)
Carnegie Mellon University,
Pittsburgh, PA, USA

Łukasz Jeż (10)
University of Wrocław, Poland

Sagar Kale (20)
EPFL, Lausanne, Switzerland

Tali Kaufman (50)
Bar-Ilan University, Ramat Gan, Israel

Arindam Khan (22)
Indian Institute of Science, Bangalore, India

Subhash Khot (9)
Department of Computer Science,
Courant Institute of Mathematical Sciences,
New York University, USA

Samir Khuller (18)
Department of Computer Science,
Northwestern University, Evanston, USA

Charlotte Knierim (58)
ETH Zurich, Switzerland

Swastik Kopparty (68)
Department of Mathematics and
Department of Computer Science,
Rutgers University, NJ, USA

Ravishankar Krishnaswamy (33)
Microsoft Research, Bengaluru, India

Alexander S. Kulikov (26)
Steklov Institute of Mathematics at
St. Petersburg,
Russian Academy of Sciences, Russia

Aounon Kumar (18)
Department of Computer Science,
University of Maryland, College Park, USA

Gunjan Kumar (30)
Tata Institute of Fundamental Research,
Mumbai, India

Neeraj Kumar (19)
Department of Computer Science,
University of California, Santa Barbara, USA

Leon Ladewig (22)
Technical University of Munich, Germany

Harry Lang (27, 62)
MIT CSAIL, Cambridge, MA, USA

Euiwoong Lee (9)
Department of Computer Science,
Courant Institute of Mathematical Sciences,
New York University, USA

Zhixian Lei (2)
School of Engineering and Applied Sciences,
Harvard University,
Cambridge, Massachusetts, USA

Johannes Lengler (58)
ETH Zurich, Switzerland

Fu Li (72)
Department of Computer Science,
University of Texas at Austin, USA

Ray Li (38)
Department of Computer Science,
Stanford University, CA, USA

Chao Liao (34)
Shanghai Jiao Tong University, China

Jiabao Lin (34)
Shanghai University of Finance and Economics,
China

Alexander Logunov (26)
St. Petersburg State University, Russia

Anand Louis (28)
Indian Institute of Science, Bangalore, India

Pinyan Lu (34)
Shanghai University of Finance and Economics, China

Nikhil S. Mande (71)
Department of Computer Science, Georgetown University, USA

Zhenyu Mao (34)
Shanghai University of Finance and Economics, China

Vasilis Margonis (47)
Department of Informatics & Telecommunications, National & Kapodistrian University of Athens, Greece

Mathieu Mari (32)
DI ENS, École Normale supérieure, Université PSL, Paris, France

Claire Mathieu (32)
CNRS, Paris, France

Jannik Matuschke (17)
Research Center for Operations Management, KU Leuven, Belgium

Arya Mazumdar (53)
University of Massachusetts Amherst, USA

Peleg Michaeli (66)
School of Mathematical Sciences, Tel Aviv University, Israel

Ivan Mihajlin (26)
University of California, San Diego, CA, USA

Gary L. Miller (8)
Carnegie Mellon University, Pittsburgh, PA, USA

Joseph S. B. Mitchell (32)
Stony Brook University, Stony Brook, NY 11794, USA

Benjamin Moseley (3)
Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA; Relational AI, Berkeley CA, USA

Partha Mukhopadhyay (57)
Chennai Mathematical Institute, Chennai, India

Jack Murtagh (42)
School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA

Nabil H. Mustafa (32)
Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge, ESIEE Paris, France

Preetum Nakkiran (2)
School of Engineering and Applied Sciences, Harvard University, Cambridge, Massachusetts, USA

Shyam Narayanan (63)
Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

Maksim Nikolaev (26)
St. Petersburg State University, Russia

Kobbi Nissim (65)
Dept. of Computer Science, Georgetown University, Washington, D.C., USA

Soumyabrata Pal (53)
University of Massachusetts Amherst, USA

Orestis Papadigenopoulos (17)
Department of Computer Science, The University of Texas at Austin, USA

Ojas Parekh (31)
Sandia National Laboratories, Albuquerque, New Mexico, USA

Will Perkins (41)
Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, Chicago, USA

Samantha Petti (66)
School of Mathematics, Georgia Institute of Technology, Atlanta, Georgia, USA

Pascal Pfister (58)
ETH Zurich, Switzerland

Ioannis Psarros (47)
Institute of Computer Science, University of Bonn, Germany

Kent Quanrud (23)
Department of Computer Science, University of Illinois at Urbana-Champaign, USA

Nived Rajaraman (33)
IIT Madras, Chennai, India

Dana Randall (37, 54)
School of Computer Science, Georgia Institute of Technology, Atlanta, Georgia, USA

R. Ravi (4)
Carnegie Mellon University, Pittsburgh, PA, USA

Rahul Raychaudhury (28)
Indian Institute of Science, Bangalore, India

Daniel Reichman (56)
Department of Computer Science,
Princeton University, NJ, USA

Omer Reingold (42)
Computer Science Department,
Stanford University, Stanford, CA USA

Nicolas Resch (68)
Department of Computer Science,
Carnegie Mellon University,
Pittsburgh, PA, USA

Andréa W. Richa (54)
Computer Science, CIDSE,
Arizona State University, Tempe, AZ, USA

Thomas Robinson (11)
Ben Gurion University of the Negev,
Beersheva, Israel

Dhruv Rohatgi (12)
MIT, Cambridge, Massachusetts, USA

Noga Ron-Zewi (68)
Department of Computer Science,
University of Haifa, Israel

Ronitt Rubinfeld (52)
CSAIL at MIT, Cambridge, MA, USA;
Blavatnik School of Computer Science at
Tel Aviv University, Israel

Daniela Rus (62)
MIT CSAIL, Cambridge, MA, USA

Clemens Rösner (18)
Institute of Computer Science,
University of Bonn, Germany

Barna Saha (53)
University of California, Berkeley, USA

Ario Salmasi (14)
Dept. of Computer Science & Engineering,
The Ohio State University, Columbus, OH, USA

Sai Sandeep (15)
Carnegie Mellon University,
Pittsburgh, PA, USA

Shubhangi Saraf (68)
Department of Mathematics and Department of
Computer Science, Rutgers University, NJ, USA

Ulysse Schaller (58)
ETH Zurich, Switzerland

Kevin Schewior (21)
Institut für Informatik,
Technische Universität München,
Garching, Germany

Daniel R. Schmidt (18)
Institute of Computer Science,
University of Cologne, Germany

Melanie Schmidt (18)
Institute of Computer Science,
University of Bonn, Germany

Grant Schoenebeck (39)
University of Michigan, Ann Arbor, USA

Andreas S. Schulz (25)
Technische Universität München, Germany

Rocco A. Servedio (44, 45)
Columbia University, New York, NY, USA

Igor Shinkar (56)
School of Computing Science,
Simon Fraser University, Burnaby, BC, Canada

Aaron Sidford (42)
Management Science & Engineering,
Stanford University, Stanford, CA USA

Anastasios Sidiropoulos (14)
Dept. of Computer Science,
University of Illinois at Chicago, USA

Shashwat Silas (68)
Department of Computer Science,
Stanford University, CA, USA

Sahil Singla (4, 49)
Princeton University, Princeton, NJ, USA;
Institute for Advanced Study,
Princeton, NJ, USA

Sandip Sinha (44)
Columbia University, New York, NY, USA

Stavros Sintos (19)
Duke University, Durham, NC, USA

Gregory B. Sorkin (60)
Department of Mathematics, The London
School of Economics and Political Science,
Houghton Street, London WC2A 2AE, England

Bruce Spang (46)
Stanford University, CA, USA

Aleksa Stanković (24)
KTH Royal Institute of Technology,
Stockholm, Sweden

Angelika Steger (58)
ETH Zurich, Switzerland

James Stewart (41)
Department of Computer Science,
University of Oxford, Oxford, UK

Subhash Suri (19)
Department of Computer Science,
University of California, Santa Barbara, USA

Maxim Sviridenko (3)
Yahoo Research, New York, NY, USA

Amnon Ta-Shma (43)
The Blavatnik School of Computer Science,
Tel-Aviv University, Tel Aviv, Israel

Li-Yang Tan (45)
Department of Computer Science,
Stanford University, Palo Alto, CA, USA

Biaoshuai Tao (39)
University of Michigan, Ann Arbor, USA

Runzhou Tao (5)
Institute for Interdisciplinary
Information Sciences,
Tsinghua University, Beijing, China 100084

Justin Thaler (55, 71)
Georgetown University, Washington, DC, USA

Devanathan Thiruvenkatachari (9)
Department of Computer Science,
Courant Institute of Mathematical Sciences,
New York University, USA

Meng-Tsung Tsai (16)
National Chiao Tung University,
Hsinchu, Taiwan

Rajan Udwani (25)
Columbia University, New York, NY, USA

Enayat Ullah (27)
Department of Computer Science,
Johns Hopkins University, Baltimore, MD, USA

Salil Vadhan (42)
School of Engineering and Applied Sciences,
Harvard University, Cambridge, MA, USA

Arsen Vasilyan (52)
CSAIL at MIT, Cambridge, MA, USA

Santosh S. Vempala (64)
School of Computer Science,
Georgia Institute of Technology, USA

Eric Vigoda (41, 48, 67)
School of Computer Science,
Georgia Institute of Technology, Atlanta, USA

Noel J. Walkington (8)
Carnegie Mellon University,
Pittsburgh, PA, USA

Alex L. Wang (8)
Carnegie Mellon University,
Pittsburgh, PA, USA

Thomas Watson (51)
University of Memphis, TN, USA

Christopher Williamson (71)
Department of Computer Science
and Engineering,
Chinese University of Hong Kong, Hong Kong

David P. Woodruff (1, 29)
Computer Science Department,
Carnegie Mellon University, Pittsburgh,
Pennsylvania, USA

Mary Wootters (38, 46)
Departments of Computer Science and Electrical
Engineering, Stanford University, CA, USA

Grigory Yaroslavtsev (69)
Indiana University, Bloomington, IN, USA;
The Alan Turing Institute, London, UK

Taisuke Yasuda (1)
Department of Mathematics,
Carnegie Mellon University,
Pittsburgh, Pennsylvania, USA

Fang-Yi Yu (39)
University of Michigan, Ann Arbor, USA

Mohammad Zaheri (65)
Dept. of Computer Science,
Georgetown University, Washington, D.C., USA

Samson Zhou (27, 69)
School of Informatics, Computing, and
Engineering, Indiana University,
Bloomington, IN, USA

David Zuckerman (72)
Department of Computer Science,
University of Texas at Austin, USA

Goran Zuzic (49)
Department of Computer Science,
Carnegie Mellon University,
Pittsburgh, PA, USA

Daniel Štefankovič (48)
Department of Computer Science,
University of Rochester, NY, USA

The Query Complexity of Mastermind with ℓ_p Distances

Manuel Fernández V
Computer Science Department, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
manuelf@andrew.cmu.edu

David P. Woodruff
Computer Science Department, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
dwoodruf@cs.cmu.edu

Taisuke Yasuda
Department of Mathematics, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
yasuda.taisuke1@gmail.com

Abstract

Consider a variant of the Mastermind game in which queries are ℓ_p distances, rather than the usual Hamming distance. That is, a codemaker chooses a hidden vector $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ and answers to queries of the form $\|\mathbf{y} - \mathbf{x}\|_p$ where $\mathbf{x} \in \{-k, -k+1, \ldots, k-1, k\}^n$. The goal is to minimize the number of queries made in order to correctly guess $\mathbf{y}$.

In this work, we show an upper bound of $O\left(\min\left\{n, \frac{n \log k}{\log n}\right\}\right)$ queries for any real $1 \leq p < \infty$ and $O(n)$ queries for $p = \infty$. To prove this result, we in fact develop a nonadaptive polynomial time algorithm that works for a natural class of separable distance measures, i.e., coordinate-wise sums of functions of the absolute value. We also show matching lower bounds up to constant factors, even for adaptive algorithms for the approximation version of the problem, in which the problem is to output $\mathbf{y}'$ such that $\|\mathbf{y}' - \mathbf{y}\|_p \leq R$ for any $R \leq k^{1-\varepsilon} n^{1/p}$ for constant $\varepsilon > 0$. Thus, essentially any approximation of this problem is as hard as finding the hidden vector exactly, up to constant factors. Finally, we show that for the noisy version of the problem, i.e., the setting when the codemaker answers queries with any $q = (1 \pm \varepsilon)\|\mathbf{y} - \mathbf{x}\|_p$, there is no query efficient algorithm.

2012 ACM Subject Classification Mathematics of computing → Combinatorics; Theory of computation → Design and analysis of algorithms

Keywords and phrases Mastermind, Query Complexity, ℓ_p Distance

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.1

Category APPROX

Funding *David P. Woodruff*: Part of this work was done while visiting Google as well as the Simons Institute for the Theory of Computing. D. Woodruff would also like to thank partial support from the Office of Naval Research (ONR) grant N00014-18-1-2562.

Acknowledgements We thank Flavio Chierichetti and Ravi Kumar for helpful discussions, as well as the anonymous reviewers for helpful feedback.

1 Introduction

Mastermind is a game played between two players, the *codemaker* and the *codebreaker*. In the 1970 original 4-position 6-color version of the game, the codemaker chooses 4 colored pegs, each taking one of 6 colors, and the codebreaker tries to guess the codemaker's 4 pegs by making queries to the codemaker by taking a guess at the sequence of the codemaker's 4 colored pegs. These guesses are answered by two numbers, the number of pegs guessed that are in the right position and the right color, indicated by black pegs, and the additional number of pegs of the right color but in the wrong position, indicated by white pegs.


© Manuel Fernández V, David P. Woodruff, and Taisuke Yasuda;
licensed under Creative Commons License CC-BY



Ever since, this game and its generalizations and variants have been studied by many computer scientists. The original version was completely characterized by [7], who showed upper and lower bounds of 5 queries for deterministic strategies. The n-position k-color generalization of the game was studied in [4], which sparked a line of research that lead to progressive improvement in upper and lower bounds for this problem, both in the original version of the game as well as in related variants of the game [2]. As these variants are not the focus of this work, we refer the reader to the expositions of [5, 2] for more details on this literature.

Note that in the variant that the codebreaker only receives the black peg answers, the problem can be phrased as guessing a hidden vector based on Hamming distance queries. One can then consider many variants of the Mastermind game in which the codebreaker guesses the codemaker's hidden vector based on other distance queries. For instance, motivated by the theory of black-box complexity, [1] recently studied the variant where the distance is the length of the longest common prefix with respect to an unknown permutation. In recreational mathematics, the ℓ_1 distance case has been studied under the name of "digit-distance" [6]. In this work, we study the case of ℓ_p distance queries. That is, the codemaker chooses a hidden vector $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ and answers to queries of the form $\|\mathbf{y} - \mathbf{x}\|_p$ where $\mathbf{x} \in \{-k, -k+1, \ldots, k-1, k\}^n$.

1.1 Our contributions

On the algorithmic side, we present Theorem 9, in which we develop a very general nonadaptive algorithm that works for any separable distance measure, i.e., the distance between $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is given by $f(\mathbf{x}-\mathbf{y})$ where $f(\mathbf{x}) = \sum_{i=1}^n g_i(|x_i|)$. When we apply this to the case of $g_i(x) = x^p$ *for any constant real* $1 \le p < \infty$, i.e., when f is the (p-th power of the) ℓ_p norm, we obtain a polynomial time algorithm making $O\left(\min\left\{n, \frac{n \log k}{\log n}\right\}\right)$ queries. For $p = \infty$, we give a simple algorithm achieving $O(n)$ queries. We also give lower bounds for any adaptive algorithm that match our upper bounds up to constant factors, *for any constant integer* $1 \le p < \infty$ (Theorem 11) and for $p = \infty$ (Theorem 12). In fact, our lower bounds are for a weaker problem, the problem of outputting an approximation $\mathbf{y}'$ such that its distance from the true hidden vector $\mathbf{y}$ is at most $\|\mathbf{y}' - \mathbf{y}\|_p \le R$, whenever the approximation radius satisfies $R \le k^{1-\varepsilon} n^{1/p}$ (where we think of $n^{1/p} = 1$ when $p = \infty$) for constant $\varepsilon > 0$. Thus, approximation for this problem is hard, in the sense that finding the point exactly is optimal up to constant factors, even when the approximation radius is as large as $k^{1-\varepsilon} n^{1/p}$.

Our main algorithmic technique for obtaining Theorem 9 is a judicious application of a generalization of the Fourier-based detecting matrix construction of [3]. Our lower bounds are simply obtained by counting the number of lattice points in an ℓ_p ball.

Finally, we consider a noisy version of the above problem, where the codemaker is allowed to answer queries with any answer that is within $(1 \pm \varepsilon)\|\mathbf{y} - \mathbf{x}\|_p$. For this variant, we show that any algorithm must make $\Omega(\exp(\varepsilon^2 \Theta(k^p n)))$ queries in Theorem 13. That is, there is no query efficient algorithm for this problem.

2 Preliminaries

2.1 Notation

▶ **Definition 1** (ℓ_p norm). *Let $1 \le p \le \infty$. Then, we endow $\mathbb{R}^n$ with the ℓ_p norm $\|\cdot\|_p$, given by*

$$\|\mathbf{x}\|_p \coloneqq \left(\sum_{i=1}^n |x_i|^p\right)^{1/p} \tag{1}$$

if $p < \infty$ *and*

$$\|\mathbf{x}\|_\infty \coloneqq \max_{i=1}^{n} |x_i| \tag{2}$$

if $p = \infty$.

▶ **Definition 2** (Weight of binary vector). *Let* $a \in \{0,1\}^\nu$. *Then,* $\mathrm{wt}(a)$ *is the number of* 1*s in* a.

▶ **Definition 3** (Even-odd decomposition). *Let* $h : \mathbb{R} \to \mathbb{R}$ *be any function. Then, the* even-odd decomposition of h *is given by*

$$\begin{aligned} h_{\mathrm{even}}(x) &\coloneqq \frac{h(x) + h(-x)}{2} \\ h_{\mathrm{odd}}(x) &\coloneqq \frac{h(x) - h(-x)}{2}. \end{aligned} \tag{3}$$

It is easy to see that $h = h_{\mathrm{even}} + h_{\mathrm{odd}}$ *and that* $h_{\mathrm{even}}(-x) = h_{\mathrm{even}}(x)$ *and* $h_{\mathrm{odd}}(-x) = -h_{\mathrm{odd}}(x)$ *for all* $x \in \mathbb{R}$.

2.2 Bshouty's detecting matrix

We very briefly review the construction of the detecting matrix of [3], as we build off of this result for our algorithms.

▶ **Definition 4** (Detecting matrix [3]). *A* $(d_1, d_2, \ldots, d_n)$*-detecting matrix is a* $\{0,1\}$*-matrix such that for every* $\mathbf{u}, \mathbf{v} \in \prod_{i=1}^{n} \{0, 1, \ldots, d_i - 1\}$ *with* $\mathbf{u} \neq \mathbf{v}$, *we have* $M\mathbf{u} \neq M\mathbf{v}$.

The theorem we use is the following:

▶ **Theorem 5** (Bshouty detecting matrix, Theorem 4/Corollary 5 of [3]). *Let* $1 < d_1 \leq d_2 \leq \cdots \leq d_n$ *where* $d_1 + d_2 + \cdots + d_n = d$. *There is a* $(d_1, d_2, \ldots, d_n)$*-detecting matrix* M *of size* $s \times n$ *where*

$$s(\log s - 4) \leq 2n \log \frac{d}{n}. \tag{4}$$

Furthermore, for $\mathbf{u} \in \prod_{i=1}^{n} \{0, 1, \ldots, d_i - 1\}$, *there is a polynomial time algorithm for recovering* $\mathbf{u}$ *given* $M\mathbf{u}$.

We will only sketch the main idea behind the construction of the matrix and the decoding algorithm, and refer the reader to [3] for the proof of the bounds and the correctness.

2.2.1 Fourier representation [3]

We consider the Fourier basis on real-valued functions defined on the Boolean hypercube $\{-1, +1\}^\nu$, i.e., the basis

$$\mathcal{B} \coloneqq \left\{ \chi_a(x) \coloneqq \prod_{a_i = 1} x_i \,\middle|\, a \in \{0,1\}^\nu \right\} \subseteq \{f : \{-1,+1\}^\nu \to \mathbb{R}\}. \tag{5}$$

It is known that $\mathcal{B}$ is an orthonormal basis, and thus any $f : \{-1,+1\}^\nu \to \mathbb{R}$ can be uniquely represented as

$$f(x) = \sum_{a \in \{0,1\}^s} \hat{f}(a) \chi_a(x) \tag{6}$$

where $\hat{f}(a)$ is the Fourier coefficient of χ_a given by

$$\hat{f}(a) = \frac{1}{2^\nu} \sum_{x \in \{-1,+1\}^\nu} f(x)\chi_a(x). \tag{7}$$

Using the fast Fourier transform, all the coefficients $\hat{f}(a)$ can be found from the values of $f(x), x \in \{-1,+1\}^\nu$, and ordered according to lexicographic order of $a \in \{0,1\}^\nu$ in time $O(\nu 2^\nu)$.

2.2.2 Detecting matrix construction

The overall idea is as follows. We choose s as in equation (4) and $\nu \coloneqq \log_2 s$. Then, we view column vectors in $\mathbb{R}^s$ with $s = 2^\nu$ rows as enumerations of the values of functions $f : \{-1,+1\}^\nu \to \mathbb{R}$. That is, for $x \in \{-1,+1\}^\nu$, the xth row of the column vector representing f is $f(x)$. We then view our detecting matrix $M \in \{0,1\}^{s \times n}$ as a family of n $\{0,1\}$-valued functions defined on $\{-1,+1\}^\nu$ and Mu as a linear combination of functions from this family, where the coefficients of the linear combination are specified by the unknown vector $\mathbf{u} \in \prod_{i=1}^n \{0,1,\ldots,d_i - 1\}$. The n functions of M have a special structure in the Fourier basis, so that there is an efficient iterative algorithm for recovering the coordinates of u in batches from the Fourier coefficients of the function Mu.

We iteratively construct columns of M as follows. For each $a \in \{0,1\}^\nu$, we will choose ℓ_a more columns to construct, so that in the end, we have $\sum_{a \in \{0,1\}^\nu} \ell_a = n$ columns.

Suppose that columns 1 through r have already been constructed. Let $a \in \{0,1\}^\nu$ and choose an integer ℓ_a such that

$$\begin{aligned} d_{r+1}d_{r+2}\ldots d_{r+\ell_a} &\le 2^{\mathrm{wt}(a)} \\ d_{r+1}d_{r+2}\ldots d_{r+\ell_a}d_{r+\ell_a+1} &> 2^{\mathrm{wt}(a)-1}. \end{aligned} \tag{8}$$

We then construct ℓ_a more columns of M so that the ith new function $g_{a,i}$ has Fourier coefficient of χ_a as

$$\hat{g}_{a,i}(a) = d_{r+1}d_{r+2}\ldots d_{r+i}/2^{\mathrm{wt}(a)} \tag{9}$$

and the Fourier coefficient of χ_b for any $b > a$ (in the usual ordering on the Boolean hypercube) as

$$\hat{g}_{a,i}(b) = 0. \tag{10}$$

The way we choose the column functions $g_{a,i}$ to have these properties is described in [3].

2.2.3 Decoding algorithm

We now show how to efficiently decode $M\mathbf{u}$. Essentially, we will decode ℓ_a of the entries of $\mathbf{u}$ at a time, subtract them off, and recurse.

Note that column vector $M\mathbf{u}$ is the enumeration of the values of a linear combination f of the $g_{a,i}$ functions from above, where the row corresponding to $x \in \{-1,+1\}^\nu$ is $f(x)$. Then, using the fast Fourier transform, we find all the Fourier coefficients $\hat{f}(z)$ for $z \in \{0,1\}^\nu$ and search for a maximal $a \in \{0,1\}^\nu$ such that $\hat{f}(a) \neq 0$. For such an a, one can prove that its Fourier coefficient in f is

$$\hat{f}(a) = \frac{1}{2^{\mathrm{wt}(a)}}(\lambda_{r+1} + \lambda_{r+2}d_{r+1} + \lambda_{r+3}d_{r+1}d_{r+2} + \cdots + \lambda_{r+\ell_a+1}d_{r+1}d_{r+2}\ldots d_{r+\ell_a}) \tag{11}$$

where r is the number of columns in M before the columns corresponding to a, and $\lambda_{r+i} = u_{r+i}$ (for sake of matching the notation in [3]). Since $\lambda_{r+i} \in \{0, 1, \ldots, d_{r+i} - 1\}$ for all $i \in [\ell_a]$, we can recover all of the λ_{r+i}. Then, these coefficients can be subtracted off and we can recurse on the remaining entries of $\mathbf{u}$.

In our Theorem 9, we will modify the above algorithm to allow for non-integer values for the λ_{r+i}, as long as they are bounded and well-separated (to be made precise later).

3 Algorithms

We now describe our upper bounds. As a warm up, we start with algorithms for ℓ_1, ℓ_2, and ℓ_∞. These will introduce some tricks that we exploit in our coordinate-wise sums algorithm. Then, we combine these tricks along with a modification of the Bshouty detecting matrix algorithm described above to obtain Theorem 9.

3.1 Algorithms for ℓ_1, ℓ_2, and ℓ_∞

Our algorithms will be based around the idea of applying the Bshouty detecting matrix M to the hidden vector $\mathbf{y}$. This can be most straightforwardly applied in the case of ℓ_2, by expanding squared distances (equation (12)).

▶ **Theorem 6** (Algorithm for ℓ_2 queries). *Let $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ be an unknown vector, and suppose that we receive answers to s queries of the form $\|\mathbf{x} - \mathbf{y}\|_2$. Then, there is a polynomial time algorithm that recovers $\mathbf{y}$ in $s = O\left(\min\left\{n, \frac{n \log k}{\log n}\right\}\right)$ queries.*

Proof. By first making the query with the $\mathbf{0}$ vector, we may find the norm $\|\mathbf{y}\|_2$ of the unknown vector. Now suppose we query for $\|\mathbf{x} - \mathbf{y}\|_2$. Note then that

$$\langle \mathbf{x}, \mathbf{y} \rangle = \frac{\|\mathbf{x}\|_2^2 + \|\mathbf{y}\|_2^2 - \|\mathbf{x} - \mathbf{y}\|_2^2}{2} \tag{12}$$

so we can compute the inner product between $\mathbf{x}$ and $\mathbf{y}$. Thus by taking n queries to be the n standard basis vectors $\mathbf{x} = \mathbf{e}_i$ for $i \in [n]$, we can always recover $\mathbf{y}$ in $n+1$ queries. To obtain $s = O\left(\frac{n \log k}{\log n}\right)$ for $k \leq n$, we can take our query vectors $\mathbf{x}$ to be the rows of the detecting matrix of Theorem 4/Corollary 5 of [3] and recover $\mathbf{y}$ by using the decoding algorithm as described in the proof. We thus conclude as desired. ◀

As shown above, if we can simulate computing inner products with binary vectors in $O(1)$ queries each, then we get an $O(n)$ algorithm by querying with the standard basis vectors or $O\left(\frac{n \log k}{\log n}\right)$ by using [3]. For ℓ_1, we take a similar approach. This time, the way we extract the inner product is quite different from the case of ℓ_2. This technique turns out to be much more flexible, and will allow us to generalize the result to coordinate-wise sums.

▶ **Theorem 7** (Algorithm for ℓ_1 queries). *Let $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ be an unknown vector, and suppose that we receive answers to s queries of the form $\|\mathbf{x} - \mathbf{y}\|_1$. Then, there is a polynomial time algorithm that recovers $\mathbf{y}$ in $s = O\left(\min\left\{n, \frac{n \log k}{\log n}\right\}\right)$ queries.*

Proof. We will just show how to compute inner products in $O(1)$ queries, since the rest follows as in the ℓ_2 case. Let $\boldsymbol{\tau} \in \{0,1\}^n$ be any binary vector and consider the sign vector $\boldsymbol{\sigma} \in \{\pm 1\}^n$ with $\sigma_i = (-1)^{\tau_i + 1}$. Then for $\sigma_i \in \{\pm 1\}$ and $-k \leq y_i \leq k$, we have that

$$|k\sigma_i - y_i| = \left|k\sigma_i - \sigma_i^2 y_i\right| = |k - \sigma_i y_i| = k - \sigma_i y_i. \tag{13}$$

Thus,

$$\|k\boldsymbol{\sigma} - \mathbf{y}\|_1 = \sum_{i=1}^{n} |k\sigma_i - y_i| = \sum_{i=1}^{n} k - \sigma_i y_i = kn - \boldsymbol{\sigma} \cdot \mathbf{y} \tag{14}$$

so we may compute the quantity $\boldsymbol{\sigma} \cdot \mathbf{y} = kn - \|k\boldsymbol{\sigma} - \mathbf{y}\|_1$. We may then compute the desired inner product with binary vectors as $\boldsymbol{\tau} \cdot \mathbf{y} = (\boldsymbol{\sigma} \cdot \mathbf{y} + \mathbf{1}_n \cdot \mathbf{y})/2$. ◀

To conclude the section, we show an $O(n)$ algorithm for ℓ_∞ queries. This turns out to be optimal, as we show later.

▶ **Theorem 8** (Algorithm for ℓ_∞ queries). *Let $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ be an unknown vector, and suppose that we receive answers to s queries of the form $\|\mathbf{x} - \mathbf{y}\|_\infty$. Then, there is a polynomial time algorithm that recovers $\mathbf{y}$ in $s = O(n)$ queries.*

Proof. For each $i \in [n]$, we make the query $q_i^+ = \|k\mathbf{e}_i - \mathbf{y}\|_\infty$ and $q_i^- = \|-k\mathbf{e}_i - \mathbf{y}\|_\infty$. Note that $y_i = 0$ if and only if these two are both equal to k. If $y_i > 0$, then $q_i^- = k + y_i > k$ and if $y_i < 0$, then $q_i^+ = k - y_i > k$. Thus, with these two queries, we can determine y_i. Thus, we recover $\mathbf{y}$ in $O(n)$ queries. ◀

3.2 Algorithm for coordinate-wise sums

In the previous section, we obtained polynomial time algorithms with tight query complexity for ℓ_1 and ℓ_2 by simulating inner product computations between $\mathbf{y}$ and binary vectors. We now generalize these ideas to an algorithm for any query given by sums along the coordinates. This in particular includes all (p-th powers of) ℓ_p norms, even for p not an integer.

▶ **Theorem 9** (Algorithm for coordinate-wise sums). *Let $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ be an unknown vector, and suppose that we receive answers to s queries of the form $f(\mathbf{y} - \mathbf{x})$, where $f(\mathbf{x}) = \sum_{i=1}^n g_i(|x_i|)$. For each $i \in [n]$, define the function $h_i(x) = g_i(k - x)$ and consider the even-odd decomposition $h_i = (h_i)_{\text{even}} + (h_i)_{\text{odd}}$ (see Definition 3). Also consider the following quantities:*

$$\begin{aligned}
M_i^{\min} &\coloneqq \min_{x \in \{-k, -k+1, \ldots, k-1, k\}} (h_i)_{\text{odd}}(x) \\
M_i^{\max} &\coloneqq \max_{x \in \{-k, -k+1, \ldots, k-1, k\}} (h_i)_{\text{odd}}(x) \\
\Delta_i &\coloneqq \min_{\substack{x_1, x_2 \in \{-k, -k+1, \ldots, k-1, k\} \\ x_1 \neq x_2}} |(h_i)_{\text{odd}}(x_1) - (h_i)_{\text{odd}}(x_2)| \\
\Delta &\coloneqq \min_{i=1}^{n} \Delta_i \\
d_i &\coloneqq \left\lceil \frac{M_i^{\max} - M_i^{\min}}{\Delta} \right\rceil + 1
\end{aligned} \tag{15}$$

If $\Delta > 0$, then there is a polynomial time algorithm that recovers $\mathbf{y}$ with $s = O\left(\min\left\{n, \frac{\log \prod_{i=1}^n d_i}{\log n}\right\}\right)$ queries.

Proof. Let $\mathbf{h}_{\text{even}}$ and $\mathbf{h}_{\text{odd}}$ be the functions that apply $(h_i)_{\text{even}}$ and $(h_i)_{\text{odd}}$ on the ith coordinate, respectively. We will show that we can recover $\mathbf{h}_{\text{odd}}(\mathbf{y})$ in $O\left(\min\left\{n, \frac{\log \prod_{i=1}^n d_i}{\log n}\right\}\right)$ queries. Note that since $\min_{i=1}^n \Delta_i > 0$, $(h_i)_{\text{odd}}$ is injective for each i and thus we can recover $\mathbf{y}$ from $\mathbf{h}_{\text{odd}}(\mathbf{y})$ in polynomial time using a lookup table for the values of $(h_i)_{\text{odd}}$.

3.2.1 Inner products with binary vectors

We first show that we can compute the inner product between $\mathbf{h}_{\text{odd}}(\mathbf{y})$ and any binary vector $\boldsymbol{\tau} \in \{0,1\}^n$. To do this, consider the sign vector $\boldsymbol{\sigma} \in \{\pm 1\}^n$ with $\sigma_i = (-1)^{\tau_i+1}$. Note that for $\sigma_i \in \{\pm 1\}$ and $-k \le y_i \le k$, we have $|k\sigma_i - y_i| = |k - \sigma_i y_i| = k - \sigma_i y_i$. Then, by querying vectors of the form $\mathbf{x} = k\boldsymbol{\sigma}$, we obtain

$$f(k\boldsymbol{\sigma} - \mathbf{y}) = \sum_{i=1}^{n} g_i(k - \sigma_i y_i) = \sum_{i=1}^{n} h_i(\sigma_i y_i). \tag{16}$$

Then using the even/oddness of $(h_i)_{\text{even}}/(h_i)_{\text{odd}}$, we have

$$\sum_{i=1}^{n} h_i(\sigma_i y_i) = \left(\sum_{i=1}^{n} (h_i)_{\text{even}}(y_i)\right) + \left(\sum_{i=1}^{n} \sigma_i (h_i)_{\text{odd}}(y_i)\right) = \mathbf{1}_n \cdot \mathbf{h}_{\text{even}}(\mathbf{y}) + \boldsymbol{\sigma} \cdot \mathbf{h}_{\text{odd}}(\mathbf{y}). \tag{17}$$

Note also that by querying for $k\mathbf{1}_n$ and $-k\mathbf{1}_n$, we also obtain

$$\begin{aligned} \frac{f(k\mathbf{1}_n - \mathbf{y}) + f(-k\mathbf{1}_n - \mathbf{y})}{2} &= \sum_{i=1}^{n} (h_i)_{\text{even}}(y_i) = \mathbf{1}_n \cdot \mathbf{h}_{\text{even}}(\mathbf{y}) \\ \frac{f(k\mathbf{1}_n - \mathbf{y}) - f(-k\mathbf{1}_n - \mathbf{y})}{2} &= \sum_{i=1}^{n} (h_i)_{\text{odd}}(y_i) = \mathbf{1}_n \cdot \mathbf{h}_{\text{odd}}(\mathbf{y}). \end{aligned} \tag{18}$$

Using these, we may compute $\boldsymbol{\tau} \cdot \mathbf{h}_{\text{odd}}(\mathbf{y}) = \frac{1}{2}(\boldsymbol{\sigma} + \mathbf{1}_n) \cdot \mathbf{h}_{\text{odd}}(\mathbf{y})$ and thus we are able to compute dot products of arbitrary binary vectors with $\mathbf{h}_{\text{odd}}(\mathbf{y})$. At this point, we can obtain $O(n)$ queries just by taking the binary vectors to be the standard basis vectors, so we focus on obtaining an algorithm making at most $O\left(\frac{\log \prod_{i=1}^{n} d_i}{\log n}\right)$ queries.

3.2.2 Modification of the Bshouty detecting matrix decoding [3]

Recall the detecting matrix of [3] for integer vectors in $\prod_{i=1}^{n} \{0, 1, \ldots, d_i - 1\}$ for $d_i \in \mathbb{N}$ for $i \in [n]$. If $\mathbf{h}_{\text{odd}}(\mathbf{y})$ took integer values, then we could just directly use this theorem to conclude with the desired query complexity. However, this is not true of $\mathbf{h}_{\text{odd}}(\mathbf{y})$, and so we need to show how to modify the [3] construction to handle our setting.

We first shift and scale our vector $\mathbf{h}_{\text{odd}}(\mathbf{y})$. Let $\mathbf{M}^{\min}$ be the vector with $M_i^{\min}$ in the ith coordinate. Note that we can easily compute $\boldsymbol{\tau} \cdot \mathbf{M}^{\min}$. Thus, we are able to compute dot products of arbitrary binary vectors with the vector $(\mathbf{h}_{\text{odd}}(\mathbf{y}) - \mathbf{M}^{\min})$. By dividing by Δ, we have dot products of arbitrary binary vectors with $\frac{1}{\Delta}\left(\mathbf{h}_{\text{odd}}(\mathbf{y}) - \mathbf{M}^{\min}\right)$. We now define this as

$$\begin{aligned} \varphi_i(y) &\coloneqq \frac{1}{\Delta}\left((h_i)_{\text{odd}}(y) - M_i^{\min}\right) \\ \boldsymbol{\varphi}(\mathbf{y}) &\coloneqq \frac{1}{\Delta}\left(\mathbf{h}_{\text{odd}}(\mathbf{y}) - \mathbf{M}^{\min}\right) \end{aligned} \tag{19}$$

Note then that $0 \le \varphi_i \le d_i - 1$ (see equation (15)) and that $y_1 \ne y_2 \implies |\varphi(y_1) - \varphi(y_2)| \ge 1$.

Now consider the detecting matrix construction of Theorem 4 in [3]. Recall that we may extract the Fourier coefficient of χ_a for some maximal a in our unknown vector $\boldsymbol{\varphi}(\mathbf{y})$ viewed as a function, which gives us

$$\lambda_{r+1} + \lambda_{r+2} d_{r+1} + \lambda_{r+3} d_{r+1} d_{r+2} + \cdots + \lambda_{r+\ell_a+1} d_{r+1} d_{r+2} \ldots d_{r+\ell_a} \tag{20}$$

which in our case we set $\lambda_j = \varphi_j(y_j)$. Now let $\mathcal{X} \coloneqq \prod_{j=r+1}^{r+\ell_a+1} \varphi_j(\{-k, -k+1, \ldots, k-1, k\})$ be the image of our original points in a subset of $\ell_a + 1$ coordinates starting at $r+1$ under the corresponding φ_j. Consider the function $\psi : \mathcal{X} \to \mathbb{R}^+$ defined via

$$\psi(\mathbf{z}) = \sum_{i=0}^{\ell_a} z_{i+1} \prod_{j=1}^{i} d_{r+j}. \tag{21}$$

It is easy to see that when we endow $\mathcal{X}$ with the lexicographical ordering, then ψ is increasing. Thus, given the Fourier coefficient as in equation (20), we can do binary search on the at most k^n values in $\mathcal{X}$ to extract the values λ_{r+i} in time $O(n \log k)$. Given this step of recovering ℓ_a of the coordinates, we can proceed as in the rest of [3] by subtracting these coordinates of the unknown vector and recursing. Hence, we conclude that we may recover $\boldsymbol{\varphi}(\mathbf{y})$ efficiently and thus $\mathbf{h}(\mathbf{y})$, as claimed. ◀

3.2.3 Reconstruction with ℓ_p queries

As a corollary of the above result, we obtain an algorithm for recovering $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ from $s = O\left(\min\left\{n, \frac{n \log k}{\log n}\right\}\right)$ distance queries in ℓ_p.

▶ **Corollary 10** (Algorithm for ℓ_p queries)**.** *Let $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ be an unknown vector, and suppose that we receive answers to s queries of the form $\|\mathbf{x} - \mathbf{y}\|_p$ for p a constant. Then, there is a polynomial time algorithm that recovers $\mathbf{y}$ in $s = O\left(\min\left\{n, \frac{n \log k}{\log n}\right\}\right)$ queries.*

Proof. We are in the setting to use Theorem 9, with $g_i(x) = x^p$ for all $i \in [n]$ and $h_{\text{odd}}(x) = \frac{1}{2}((k-x)^p - (k+x)^p)$. Recall that we have efficient algorithms with the desired guarantees when $p \in \{1, 2\}$ so we dismiss these cases. In the remaining range of p, we just need to compute $\prod_{i=1}^{n} d_i$.

Note that

$$h'_{\text{odd}}(x) = -\frac{p}{2}\left((k-x)^{p-1} + (k+x)^{p-1}\right) < 0 \tag{22}$$

on $x \in [-k, k]$ so h_{odd} is decreasing on this interval. Then, $(2k)^p/2 = h_{\text{odd}}(-k) \geq h_{\text{odd}}(x) \geq h_{\text{odd}}(k) = -(2k)^p/2$. Furthermore, note that

$$h''_{\text{odd}}(x) = \frac{p(p-1)}{2}\left((k-x)^{p-2} - (k+x)^{p-2}\right). \tag{23}$$

If $p > 2$, then this is negative on $x \geq 0$, so $|h'_{\text{odd}}(x)|$ is smallest at $x = 0$ and thus $|h'_{\text{odd}}(x)| \geq |h'_{\text{odd}}(0)| = pk^{p-1}$ for all x. If $1 < p < 2$, then this is positive on $x \geq 0$, so $|h'_{\text{odd}}(x)|$ is smallest at $x = k$ and thus $|h'_{\text{odd}}(x)| \geq |h'_{\text{odd}}(k)| = (p/2)(2k)^{p-1}$ for all x. In either case, we have that $\Delta = \Omega(pk^{p-1})$ and the range is $M^{\max} - M^{\min} = O(k^p)$ and thus $d_i = O((M^{\max} - M^{\min})/\Delta) = O(k)$. Thus, the query complexity is

$$O\left(\min\left\{n, \frac{\log \prod_{i=1}^{n} d_i}{\log n}\right\}\right) = O\left(\min\left\{n, \frac{n \log k}{\log n}\right\}\right) \tag{24}$$

as desired. ◀

4 Lower Bounds

In this section, we complement our algorithms with matching lower bounds, for integer p. Our lower bounds work even for the problem of approximating the hidden vector and for adaptive randomized algorithms with constant success probability.

▶ **Theorem 11** (Lower bound for integer ℓ_p). *Let $1 \le p < \infty$ be a constant integer and let $R \in (0, kn^{1/p}]$ be an approximation radius. Suppose there exists an algorithm $\mathcal{A}$ such that for all unknown vectors $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$, $\mathcal{A}$ outputs a vector $\mathbf{y}' \in \{-k, -k+1, \ldots, k-1, k\}^n$ such that*

$$\|\mathbf{y}' - \mathbf{y}\|_p \le R \tag{25}$$

in s possibly adaptive ℓ_p queries with probability at least $2/3$ over the algorithm's random coin tosses. Then

$$s = \Omega\left(\frac{n \log(kn^{1/p}/R)}{\log k + \log n}\right). \tag{26}$$

In particular, if $R \le k^{1-\varepsilon} n^{1/p}$ for some constant $\varepsilon > 0$, then

$$s = \Omega\left(\frac{n \log k}{\log k + \log n}\right), \tag{27}$$

which is $\Omega\left(\frac{n \log k}{\log n}\right)$ if $k < n$ and $\Omega(n)$ if $k \ge n$.

Proof. By Yao's minimax principle [9], it suffices to show the lower bound for all deterministic algorithms $\mathcal{A}$ that correctly approximates a uniformly random $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ with probability at least $2/3$.

Note that each query $\|\mathbf{x} - \mathbf{y}\|_p^p$ results in a nonnegative integer that is at most $(2k)^p n$. Thus, there are at most $((2k)^p n + 1)^s$ possible sequences of answers. Now let Q be the set of all sequence of answers that $\mathcal{A}$ can observe, and for each sequence of answers $\mathbf{q} \in Q$, let $S_{\mathbf{q}}$ denote the set of vectors $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$ such that the deterministic algorithm $\mathcal{A}$ observes $\mathbf{q}$ on input $\mathbf{y}$. Then, $S_{\mathbf{q}}$ partitions the unknown vectors $\mathbf{y}$ into $|Q|$ disjoint sets. Then, the probability that $|S_{\mathbf{q}}|$ has size at most $\frac{1}{100}\frac{(2k+1)^n}{|Q|}$ is

$$\begin{aligned}
\Pr_{\mathbf{y}}\left(|S_{\mathbf{q}}| \le \frac{1}{100}\frac{(2k+1)^n}{|Q|}\right) &= \sum_{\substack{\mathbf{q}\in Q \\ |S_{\mathbf{q}}| \le \frac{1}{100}\frac{(2k+1)^n}{|Q|}}} \Pr_{\mathbf{y}}(\mathcal{A} \text{ queries the sequence } \mathbf{q}) \\
&= \sum_{\substack{\mathbf{q}\in Q \\ |S_{\mathbf{q}}| \le \frac{1}{100}\frac{(2k+1)^n}{|Q|}}} \frac{|S_{\mathbf{q}}|}{(2k+1)^n} \\
&\le \sum_{\substack{\mathbf{q}\in Q \\ |S_{\mathbf{q}}| \le \frac{1}{100}\frac{(2k+1)^n}{|Q|}}} \frac{1}{100}\frac{(2k+1)^n}{|Q|}\frac{1}{(2k+1)^n} \\
&\le \sum_{\mathbf{q}\in Q} \frac{1}{100|Q|} = \frac{1}{100}.
\end{aligned} \tag{28}$$

Thus with probability at least $99/100$, $|S_{\mathbf{q}}|$ has size at least $\frac{1}{100}\frac{(2k+1)^n}{|Q|}$.

Note that by [8], the volume of a unit ℓ_p ball is $2^n \Gamma(1+1/p)^n / \Gamma(1+n/p)$, so the volume of a ball of radius R in ℓ_p is

$$V := R^n 2^n \frac{\Gamma(1+1/p)^n}{\Gamma(1+n/p)} = \left(\Theta\left(\frac{R}{n^{1/p}}\right)\right)^n. \tag{29}$$

Now suppose that $\mathbf{q}$ is a sequence of queries such that $|S_{\mathbf{q}}| > 2V$ and let $\mathbf{z}$ be the output of the deterministic algorithm $\mathcal{A}$ on the sequence of queries $\mathbf{q}$. Then, at most V of the points

in S can be in the ℓ_p ball of radius R centered at $\mathbf{z}$. Thus, with probability at least $1/2$ over the random hidden vector $\mathbf{y}$, we output a point $\mathbf{z}$ such that $\|\mathbf{z}-\mathbf{y}\|_p \geq R$. Thus, if

$$\frac{1}{100}\frac{(2k+1)^n}{|Q|} > 2V, \tag{30}$$

then our probability of success is at most $1/2 + 1/100$ and thus we do not have a correct algorithm. Thus, it must be that

$$\frac{1}{100}\frac{(2k+1)^n}{|Q|} \leq 2V \implies \frac{(2k+1)^n}{200V} \leq |Q| \leq ((2k)^p n+1)^s. \tag{31}$$

Rearranging, we have that

$$s \geq \frac{\log \frac{(2k+1)^n}{200V}}{\log((2k)^p n+1)} = \Omega\left(\frac{n\log(kn^{1/p}/R)}{\log k+\log n}\right), \tag{32}$$

as claimed. ◀

For $p=\infty$, we have a lower bound of $\Omega(n)$ regardless of k.

▶ **Theorem 12** (Lower bound for ℓ_∞). *Let $R \in (0,k]$ be an approximation radius. Suppose there exists an algorithm $\mathcal{A}$ such that for all unknown vectors $\mathbf{y} \in \{-k,-k+1,\ldots,k-1,k\}^n$, $\mathcal{A}$ outputs a vector $\mathbf{y}' \in \{-k,-k+1,\ldots,k-1,k\}^n$ such that*

$$\|\mathbf{y}'-\mathbf{y}\|_\infty \leq R \tag{33}$$

in s possibly adaptive ℓ_∞ queries with probability at least $2/3$ over the algorithm's random coin tosses. Then

$$s = \Omega\left(\frac{n\log(k/R)}{\log k}\right). \tag{34}$$

In particular, if $R \leq k^{1-\varepsilon}$ for a constant $\varepsilon > 0$, then $s = \Omega(n)$.

Proof. By the same argument as the finite ℓ_p case, we use Yao's minimax principle to reduce the argument to a lower bound for all deterministic algorithms $\mathcal{A}$ on uniformly random inputs $\mathbf{y}$ succeeding with probability at least $2/3$. Furthermore, by the same partition argument as before, we have that $|S_{\mathbf{q}}|$ is at least $\frac{1}{100}\frac{(2k+1)^n}{|Q|}$ with probability at least $99/100$.

The volume of an ℓ_∞ ball of radius R is $(2R)^n$, so as before, we must have

$$\frac{1}{100}\frac{(2k+1)^n}{|Q|} \leq 2(2R)^n. \tag{35}$$

When $p=\infty$, there are only $(2k+1)^s$ possible sequences of answers, so we instead have the bound

$$\frac{(2k+1)^n}{(2R)^n} \leq 200(2k+1)^s \tag{36}$$

By rearranging, we obtain the bound $s = \Omega\left(\frac{n\log(k/R)}{\log k}\right)$ as desired. ◀

4.1 Lower bound for the noisy problem

Finally, we show that in the noisy version of the problem, i.e., the setting where the codemaker is allowed to answer the queries $\mathbf{x}$ with any $q = (1 \pm \varepsilon)\|\mathbf{y} - \mathbf{x}\|_p$, there is no good algorithm.

▶ **Theorem 13** (Lower bound for the noisy problem). *Let $1 \leq p < \infty$ be a constant and let $0 < R < kn^{1/p}$ be an approximation radius. Suppose there exists an algorithm $\mathcal{A}$ such that for all unknown vectors $\mathbf{y} \in \{-k, -k+1, \ldots, k-1, k\}^n$, $\mathcal{A}$ outputs a vector $\mathbf{y}' \in \{-k, -k+1, \ldots, k-1, k\}^n$ such that*

$$\|\mathbf{y}' - \mathbf{y}\|_p \leq R \tag{37}$$

in s possibly adaptive $(1 \pm \varepsilon)$-noisy ℓ_p queries, i.e., answers with adversarially chosen $q_{\mathbf{x}} = (1 \pm \varepsilon)\|\mathbf{y} - \mathbf{x}\|_p$, with probability at least $2/3$ over the algorithm's random coin tosses. Then

$$s = \Omega\big(\exp\big(\varepsilon^2 \Theta(k^p n)\big)\big). \tag{38}$$

Proof. By Yao's minimax principle, we can take the algorithm to be deterministic by taking our hidden vector $\mathbf{y}$ to be drawn uniformly from $\{-k, -k+1, \ldots, k-1, k\}^n$. Now fix any query $\mathbf{x} \in \{-k, -k+1, \ldots, k-1, k\}^n$ and let $\mu = \mathbf{E}_{\mathbf{z}}\Big(\|\mathbf{x} - \mathbf{z}\|_p^p\Big) = \Theta(k^p n)$. Then by Chernoff bounds,

$$\mathop{\mathbf{Pr}}_{\mathbf{y}}\Big(\Big|\|\mathbf{x} - \mathbf{y}\|_p^p - \mu\Big| \geq \varepsilon\mu\Big) \leq 2\exp\big(-\varepsilon^2\mu\big). \tag{39}$$

Thus, if the number of queries s is less than $\exp\big(\varepsilon^2\Theta(k^p n)\big)/200$, then by the union bound over the s queries, with probability at least $99/100$ over the choice of $\mathbf{y}$, the codemaker can just return $\mathbf{E}_{\mathbf{z}}\Big(\|\mathbf{x} - \mathbf{z}\|_p^p\Big)$ for any query $\mathbf{x}$. Thus, the deterministic codebreaker algorithm sees the same sequence of answers with probability at least $99/100$ and so the algorithm cannot be correct. Hence, we conclude that $s = \Omega(\exp\big(\varepsilon^2\Theta(k^p n)\big))$. ◀

References

1 Peyman Afshani, Manindra Agrawal, Benjamin Doerr, Carola Doerr, Kasper Green Larsen, and Kurt Mehlhorn. The query complexity of a permutation-based variant of Mastermind. *Discrete Applied Mathematics*, 2019.

2 Aaron Berger, Christopher Chute, and Matthew Stone. Query complexity of mastermind variants. *Discrete Mathematics*, 341(3):665–671, 2018.

3 Nader H. Bshouty. Optimal Algorithms for the Coin Weighing Problem with a Spring Scale. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*, 2009. URL: `http://www.cs.mcgill.ca/%7Ecolt2009/papers/004.pdf#page=1`.

4 Vasek Chvátal. Mastermind. *Combinatorica*, 3(3):325–329, 1983. `doi:10.1007/BF02579188`.

5 Benjamin Doerr, Carola Doerr, Reto Spöhel, and Henning Thomas. Playing Mastermind With Many Colors. *J. ACM*, 63(5):42:1–42:23, 2016. `doi:10.1145/2987372`.

6 David Ginat. Digit-distance Mastermind. *The Mathematical Gazette*, 86(507):437–442, 2002.

7 Donald E. Knuth. The computer as a master mind. *Journal of Recreational Mathematics*, 9:1–6, 1977.

8 Xianfu Wang. Volumes of generalized unit balls. *Mathematics Magazine*, 78(5):390–395, 2005.

9 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Annual IEEE Symposium on Foundations of Computer Science*, pages 222–227. IEEE, 1977.

Tracking the ℓ_2 Norm with Constant Update Time

Chi-Ning Chou
School of Engineering and Applied Sciences, Harvard University, Cambridge, Massachusetts, USA
http://cnchou.github.io
chiningchou@g.harvard.edu

Zhixian Lei
School of Engineering and Applied Sciences, Harvard University, Cambridge, Massachusetts, USA
zhixianlei@seas.harvard.edu

Preetum Nakkiran
School of Engineering and Applied Sciences, Harvard University, Cambridge, Massachusetts, USA
http://preetum.nakkiran.org
preetum@cs.harvard.edu

Abstract

The ℓ_2 *tracking problem* is the task of obtaining a streaming algorithm that, given access to a stream of items $a_1, a_2, a_3, \ldots$ from a universe $[n]$, outputs at each time t an estimate to the ℓ_2 norm of the *frequency vector* $f^{(t)} \in \mathbb{R}^n$ (where $f_i^{(t)}$ is the number of occurrences of item i in the stream up to time t). The previous work [Braverman-Chestnut-Ivkin-Nelson-Wang-Woodruff, PODS 2017] gave a streaming algorithm with (the optimal) space using $O(\epsilon^{-2}\log(1/\delta))$ words and $O(\epsilon^{-2}\log(1/\delta))$ update time to obtain an ϵ-accurate estimate with probability at least $1-\delta$. We give the first algorithm that achieves update time of $O(\log 1/\delta)$ which is independent of the accuracy parameter ϵ, together with the nearly optimal space using $O(\epsilon^{-2}\log(1/\delta))$ words. Our algorithm is obtained using the Count Sketch of [Charilkar-Chen-Farach-Colton, ICALP 2002].

2012 ACM Subject Classification Theory of computation → Sketching and sampling

Keywords and phrases Streaming algorithms, Sketching algorithms, Tracking, CountSketch

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.2

Category APPROX

Related Version A full version of the paper is available at https://arxiv.org/abs/1807.06479.

Funding *Chi-Ning Chou*: Supported by NSF awards CCF 1565264 and CNS 1618026.
Zhixian Lei: Supported by NSF awards CCF 1565264 and CNS 1618026.
Preetum Nakkiran: Work supported in part by a Simons Investigator Award, NSF Awards CCF 1565641 and CCF 1715187, and the NSF Graduate Research Fellowship Grant No. DGE1144152.

Acknowledgements The authors wish to thank Jelani Nelson for invaluable advice throughout the course of this research. We also thank Mitali Bafna and Jarosław Błasiok for useful discussion and thank Boaz Barak for many helpful comments on an earlier draft of this article. We are also grateful to reviewers' comments.

1 Introduction

The *streaming model* considers the following setting. One is given a list $a_1, a_2, \ldots, a_m \in [n]$ as input where we think of n as extremely large. The algorithm is only allowed to read the input once in a stream and the goal is to answer some predetermined queries using space of size logarithmic in n. For each $i \in [n]$ and time $t \in [m]$, define $f_i^{(t)} = |\{1 \le j \le t : \ a_j = i\}|$ as the frequency of i at time t. Many classical streaming problems are concerned with approximating statistics of $f^{(m)}$ such as the distinct element problem (*i.e.*, $\|f^{(m)}\|_0$). One of


© Chi-Ning Chou, Zhixian Lei, and Preetum Nakkiran;
licensed under Creative Commons License CC-BY



the most well-studied problems is the one-shot ℓ_2 estimation problem where the goal is to estimate $\|f^{(m)}\|_2^2$ within multiplicative error $(1 \pm \epsilon)$ and had been achieved by the seminal AMS sketch by Alon et al. [1].

We consider a streaming algorithm A that maintains some logarithmic space and outputs an estimation σ_t at the t^{th} step of the computation. A achieves ℓ_2 (ϵ, δ)-tracking if for every input stream $a_1, a_2, \ldots, a_m \in [n]$

$$\Pr\left[\exists_{t\in[m]} |\sigma_t - \|f^{(t)}\|_2^2| > \epsilon\Delta_t\right] \leq \delta$$

where the "normalization factor" Δ_t differs between *strong* tracking and *weak* tracking. For (ϵ, δ)-*strong tracking*, $\Delta_t = \|f^{(t)}\|_2^2$ is the norm squared of the frequency vector up to the time t, while for (ϵ, δ)-weak tracking, $\Delta_t = \|f^{(m)}\|_2^2$ is the norm squared of the overall frequency vector. Note that strong tracking implies weak tracking and weak tracking implies one-shot approximation. In this work, we focus on ℓ_2 tracking via linear sketching, where we specify a distribution D on matrices $\Pi \in \mathbb{R}^{k\times n}$, and maintain a sketch vector at time t as $\tilde{f}^{(t)} \triangleq \Pi f^{(t)}$. Then the estimate σ_t is defined as $\|\tilde{f}^{(t)}\|_2^2$. The space complexity of A is the number of machine words[1] required by A. The update time complexity of A is the time to update σ_t, in terms of number of arithmetic operations.

Both weak tracking and strong tracking have been studied in different context [11, 5, 4] and the focus of this paper is on the *update time complexity*. Specifically, we are interested in the dependency of update time on the approximation factor ϵ. The state-of-the-art result prior to our work is by Braverman et al. [4] showing that AMS provides weak tracking with $O(\epsilon^{-2}\log(1/\delta))$ update time and $O(\epsilon^{-2}\log(1/\delta))$ words of space.

Apart from tracking, there have been several sketching algorithms for one-shot approximation that have faster update time. Dasgupta et al. [8] and Kane and Nelson [16] showed that sparse JL achieves $O_\delta(\epsilon^{-1})$ [2] update time for ℓ_2 one-shot approximation. Charikar, Chen, and Farach-Colton [6] designed the CountSketch algorithm for the heavy hitter problem and Thorup and Zhang [23] showed that it achieve $O_\delta(1)$ update time for ℓ_2 one-shot approximation.

Update time

Unlike the space complexity in streaming model, there have been less studies in the update time complexity though it is of great importance in applications. For example, the *packet passing problem* [21] requires the ℓ_2 estimation in the streaming model with input arrival rate as high as 7.75×10^6 packets [3] per second. Thorup and Zhang [24] improved the update time from 182 nanoseconds to 50 nanoseconds and made the algorithm more practical.

While some streaming problems have algorithms with constant update time (*e.g.,* distinct elements [19] and ℓ_2 estimation [24]), some other important problems do not (ℓ_p estimation for $p \neq 2$ [17], heavy hitters problems[4] [6, 7], and tracking problems [4]). Larsen et al. [22] systematically studies the update time complexity and showed lower bounds against heavy hitters, point query, entropy estimation, and moment estimation in the non-adaptive turnstile streaming model. In particular, they show that $O(\epsilon^{-2})$-space algorithms for ℓ_2 estimation of vectors over $\mathbb{R}^n$, with failure probability δ, must have update time roughly $\Omega(\log(1/\delta)/\sqrt{\log n})$. Note that their lower bound does not depend on ϵ.

---

[1] Following convention, we assume the size of a machine word is at least $\Omega(\max(\log n, \log m))$ bits.
[2] $O_\delta(\cdot)$ is the same as the usual big O notation except treating δ as a constant.
[3] Each packet has 40 bytes (320 bits).
[4] There is a memory and update time tradeoff for heavy hitter from space $O(\epsilon^{-2}\log(n/\delta))$ to $O(\epsilon^{-2}(n/\delta))$ to get constant update time. However, achieving constant update time and logarithmic space simultaneously is unknown.

Space lower bounds

For one-shot estimation of the ℓ_2 norm, Kane et al. [20] showed that $\Theta(\epsilon^{-2}\log m + \log\log n)$ bits of space are required, for any streaming algorithm. This space lower bound is tight due to the AMS sketch. However, this only applies in the constant failure probability regime.

In the regime of sub-constant failure probability δ, known tight lower-bounds on Distributional JL [15, 14] imply that $\Omega(\epsilon^{-2}\log(1/\delta))$ rows are necessary for the special case of linear sketching algorithms. [5] For linear sketches, this lower bound on number of rows is equivalent to a lower bound on the words of space.

For the regime of faster update time, Kane and Nelson [16] shows that CountSketch-type of constructions (with the optimal $\Omega(\epsilon^{-2}\log(1/\delta))$ rows) require sparsity i.e. number of non-zero elements $\widetilde{\Omega}(\epsilon^{-1}\log(1/\delta))$ [6] per column to achieve distortion ϵ and failure probability δ. But, this does not preclude a sketch with suboptimal dependency on δ in the number of rows from having constant sparsity, for example a sketch with $\Omega_\delta(\epsilon^{-2})$ rows and constant sparsity – indeed, this is what CountSketch achieves. Note that in our setting, we can boost constant-failure probability to arbitrarily small failure probability by taking medians of estimators.[7] Thus, we may be able to bypass the lower-bounds for linear sketches.

To summarize the situation: for constant failure probability, it is only known that linear sketches require dimension $\Omega(\epsilon^{-2})$, and it is not known if super-constant sparsity is required for tracking with this optimal dimension. In particular, it was not known how to achieve say $(\epsilon, O(1))$-weak tracking for ℓ_2, with $O(\epsilon^{-2})$ words of space and constant update time.

Our contributions

In this paper, we show that there is a streaming algorithm with $O(\log(1/\delta))$ update time and space using $O(\epsilon^{-2}\log(1/\delta))$ words that achieves ℓ_2 (ϵ,δ)-weak tracking.

▶ **Theorem 1** (informal). *For any $\epsilon > 0$, $\delta \in (0,1)$, and $n \in \mathbb{N}$. For any insertion-only stream over $[n]$ with frequencies $f^{(1)}, f^{(2)}, \ldots, f^{(m)}$, there exists a streaming algorithm providing ℓ_2 (ϵ,δ)-weak tracking with space using $O(\epsilon^{-2}\log(1/\delta))$ words and $O(\log(1/\delta))$ update time.*

Further, by applying a standard union bound argument in Lemma 13, the same algorithm can achieve ℓ_2 strong tracking as well.

▶ **Corollary 2.** *For any $\epsilon > 0$, $\delta \in (0,1)$, and $n \in \mathbb{N}$. For any insertion-only stream over $[n]$ with frequencies $f^{(1)}, f^{(2)}, \ldots, f^{(m)}$, there exists a streaming algorithm providing ℓ_2 (ϵ,δ)-strong tracking with $O(\epsilon^{-2}\log(1/\delta)\log\log m)$ words and $O(\log(1/\delta)\log\log m)$ update time.*

The algorithm in the main theorem is obtained by running $O(\log(1/\delta))$ many copies of CountSketch and taking the median.

The main techniques used in the proof are the chaining argument and Hansen-Wright inequality which are also used in [4] to show the tracking properties of AMS. However, direct applications of these tools on the CountSketch algorithm would not give the desired bounds due to the sparse structure of the sketching matrix. To overcome this issue, we have to dig into the structure of sketching matrix of CountSketch. We will compare the difference between our techniques and that in [4] after presenting the proof of Theorem 1 (see Remark 12).

[5] Note that an (ϵ,δ)-weak tracking via linear sketch defines a distribution over matrices that satisfies the Distributional JL guarantee, with distortion $(1\pm\epsilon)$ and failure probability δ.
[6] $\widetilde{\Omega}(\cdot)$ is the same as the $\Omega(\cdot)$ notation by ignoring extra logarithmic factor.
[7] This is not immediate for weak tracking.

The rest of the paper is organized as follows. Some preliminaries are provided in Section 2. In Section 3, we prove our main theorem showing that CountSketch with $O(\epsilon^{-2})$ rows achieves ℓ_2 $(\epsilon, O(1))$-weak tracking with constant update time. As for the ℓ_2 strong tracking, we discuss some upper and lower bounds in Section 4. In Section 5, we discuss some future directions and open problems.

2 Preliminaries

In the following, $n \in \mathbb{N}$ denotes the size of the universe, k denotes the number of rows of the sketching matrix, t denotes the time, and m denote the final time. We let $[n] = \{1, 2, \ldots, n\}$ and use $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ to denote the usual $O(\cdot)$ and $\Omega(\cdot)$ with some extra poly-logarithmic factor.

The input of the streaming algorithm is a list $a_1, a_2, \ldots, a_m \in [n]$. For each $i \in [n]$ and time $t \in [m]$, define $f_i^{(t)} = |\{1 \le j \le t : \ a_j = i\}|$ as the frequency of i at time t. The one-shot ℓ_2 approximation problem is to produce an estimate for $\|f^{(m)}\|_2^2$ with $(1 \pm \epsilon)$ multiplicative error and success probability at least $1 - \delta$ for $\epsilon > 0$ and $\delta \in (0, 1)$.

2.1 ℓ_2 tracking

Here, we give the formal definition of ℓ_2 tracking for sketching algorithm.

▶ **Definition 3** (ℓ_2 tracking). *For any $\epsilon > 0, \delta \in (0, 1)$, and $n, m \in \mathbb{N}$. Let $f^{(1)}, f^{(2)}, \ldots, f^{(m)}$ be the frequency of an insertion-only stream over $[n]$ and $\tilde{f}^{(1)}, \tilde{f}^{(2)}, \ldots, \tilde{f}^{(m)}$ be its (randomized) approximation produced by a sketching algorithm. We say the algorithm provides ℓ_2 (ϵ, δ)-strong tracking if*

$$\Pr\left[\exists_{t\in[m]}, \ \left|\|\tilde{f}^{(t)}\|_2^2 - \|f^{(t)}\|_2^2\right| > \epsilon\|f^{(t)}\|_2^2\right] \le \delta.$$

We say the algorithm provides ℓ_2 (ϵ, δ)-weak tracking if

$$\Pr\left[\exists_{t\in[m]}, \ \left|\|\tilde{f}^{(t)}\|_2^2 - \|f^{(t)}\|_2^2\right| > \epsilon\|f^{(m)}\|_2^2\right] \le \delta.$$

Note that the difference between the two tracking guarantee is that in strong tracking we bound the deviation of the estimate from the true norm squared by $\epsilon\|f^{(t)}\|_2^2$ while in the weak tracking we bound this deviation by $\epsilon\|f^{(m)}\|_2^2$.

2.2 AMS sketch and CountSketch

Alon *et al.* [1] proposed the seminal AMS sketch for ℓ_2 approximation in the streaming model. In AMS sketch, consider $\Pi \in \mathbb{R}^{k\times n}$ where $\Pi_{j,i} = \sigma_{j,i}/\sqrt{k}$ and $\sigma_{j,i}$ is i.i.d. Rademacher for each $j \in [m], i \in [n]$. When $k = O(\epsilon^{-2})$, AMS sketch approximates ℓ_2 norm within $(1 \pm \epsilon)$ multiplicative error. Note that the update time of AMS sketch is k since the matrix Π is dense.

Charikar, Chen, and Farach-Colton [6] proposed the following CountSketch algorithm for the heavy hitter problem and Thorup and Zhang [23] showed that CountSketch is also able to solve the ℓ_2 approximation. Here, consider $\Pi \in \mathbb{R}^{k\times n}$ where we denote the i^{th} column of Π as Π_i for each $i \in [n]$. Π_i is defined as follows. First, pick $j \in [k]$ uniformly and set $\Pi_{j,i}$ to be an independent Rademacher. Next, set the other entries in Π_i to be 0. Note that unlike AMS sketch, the normalization term in CountSketch is 1 since there is exactly one non-zero entry in each column. [6] showed that CountSketch provides one-shot ℓ_2 approximation with $O(\epsilon^{-2})$ rows.

▶ **Lemma 4** ([6, 23]). *Let $\epsilon > 0$, $\delta \in (0,1)$, and $n \in \mathbb{N}$. Pick $k = \Omega(\epsilon^{-2}\delta^{-1})$, we have for any $x \in \mathbb{R}^n$,*

$$\Pr_{\Pi}\left[|\|\Pi \boldsymbol{x}\|_2^2 - \|\boldsymbol{x}\|_2^2| > \epsilon\|\boldsymbol{x}\|_2^2\right] \leq \delta.$$

Implement CountSketch in logarithmic space

Previously, we defined CountSketch using uniformly independent randomness, which requires space $\Omega(nk)$. However, one could see that in the proof of Theorem 8 we actually only need 8-wise independence. Thus, the space required can be reduced to $O(\log n)$ for each row. It is well known that CountSketch with k rows can be implemented with 8-wise independent hash family using $O(k)$ words. We describe the whole implementation in Appendix A for completeness.

2.3 ϵ-net for insertion-only stream

In our analysis, we will use the following existence of a small ϵ-net for insertion-only streams.

▶ **Definition 5** (ϵ-net). *Let $S \subseteq \mathbb{R}^n$ be a set of vectors. For any $\epsilon > 0$, we say $E \subseteq \mathbb{R}^n$ is an ϵ-net for S with respect to ℓ_2 norm if for any $x \in S$, there exists $y \in E$ such that $\|x - y\|_2 \leq \epsilon$.*

▶ **Lemma 6** ([5]). *Let $\{x^{(t)}\}_{t\in[m]}$ be an insertion-only stream. For any $\epsilon > 0$, there exists a size $\left(1 + \epsilon^{-2} \cdot \|x^{(m)}\|_2\right)$ ϵ-net for $\{x^{(t)}\}_{t\in[m]}$ with respect to ℓ_2 norm. Moreover, the elements in the net are all from $\{x^{(t)}\}_{t\in[m]}$.*

Proof Sketch. The idea is to use a greedy algorithm, by scanning through the stream from the beginning and adding an element $x^{(t)}$ into the net if there does not already exist an element in the net that is ϵ-close to $x^{(t)}$. ◀

2.4 Concentration inequalities

Our analysis crucially relies on the following Hanson-Wright inequality [10].

▶ **Lemma 7** (Hanson-Wright inequality [10]). *For any symmetric $B \in \mathbb{R}^{n\times n}$, $\sigma \in \{\pm 1\}^n$ being independent Rademacher vector, and integer $p \geq 1$, we have*

$$\|\sigma^\top B\sigma - \mathbb{E}_\sigma[\sigma^\top B\sigma]\|_p \leq O\left(\sqrt{p}\|B\|_F + p\|B\|\right) = O(p\|B\|_F),$$

where $\|X\|_p$ is defined as $\mathbb{E}[|X|^p]^{1/p}$ and $\|\cdot\|_F$ is the Frobenius norm.

Note that the only randomness in $\sigma^\top B\sigma - \mathbb{E}_\sigma[\sigma^\top B\sigma]$ is the Rademacher vector σ.

3 CountSketch with $O(\epsilon^{-2})$ rows provides ℓ_2 weak tracking

In this section we will show that CountSketch with $O(\epsilon^{-2})$ rows provides $(\epsilon, O(1))$-weak tracking.

▶ **Theorem 8** (CountSketch with $O(\epsilon^{-2})$ rows provides ℓ_2 weak tracking). *For any $\epsilon > 0$, $\delta \in (0,1)$, and $n \in \mathbb{N}$. Pick $k = \Omega(\epsilon^{-2}\delta^{-1})$. For any insertion-only stream over $[n]$ with frequency $f^{(1)}, f^{(2)}, \ldots, f^{(m)}$, the CountSketch algorithm with k rows provides ℓ_2 (ϵ, δ)-weak tracking.*

▶ Remark. Note that for linear sketches, the dependency of number of rows on ϵ is tight in Theorem 8. This is implied by known lower-bounds on Distributional JL [15, 14], which imply lower-bounds on one-shot ℓ_2 approximation.

▶ Remark. Recall that the number of rows in linear sketches is proportional to the number of words needed in the algorithm.

Using the standard median trick, we can run $O(\log(1/\delta))$ copies of CountSketch with $k = O(\epsilon^{-2})$ in parallel and output the median. With this, Theorem 8 immediately gives the following corollary with better dependency on δ.

▶ **Corollary 9.** *For any $\epsilon > 0$, $\delta \in (0,1)$, and $n \in \mathbb{N}$. For any insertion-only stream over $[n]$ with frequency $f^{(1)}, f^{(2)}, \ldots, f^{(m)}$, there exists a streaming algorithm providing ℓ_2 (ϵ,δ)-weak tracking with $k = O(\epsilon^{-2}\log(1/\delta))$ rows and update time $O(\log(1/\delta))$.*

The proof of Theorem 8 uses the Dudley-like chaining technique similar to other tracking proofs [4]. However, direct application of the chaining argument would not suffice and we have to utilize the structure of the sketching matrix of CountSketch (see Remark 12 for comparison). We will prove Theorem 8 in Subsection 3.1.

3.1 Proof of Theorem 8

In this subsection, we give a formal proof for our main theorem. Let us start with some notations for CountSketch. Recall that for any $i \in [n]$, the i^{th} column of Π is defined by (i) picking $j \in [k]$ uniformly and set $\Pi_{j,i}$ to be a Rademacher random variable and (ii) set the other entries in Π_i to be 0. Denote $\Pi_{j,i} = \sigma_{j,i}\eta_{j,i}$, where $\sigma_{j,i}$ is a Rademacher random variable, and $\eta_{j,i}$ is the indicator for choosing the j^{th} row in the i^{th} column. Note that there is exactly one non-zero entry in each column and the probability distribution is uniform. The approximation error of Π for a vector $\mathbf{x} \in \mathbb{R}^n$ is denoted as $\gamma(\mathbf{x}) := \left|\|\Pi\mathbf{x}\|_2^2 - \|\mathbf{x}\|_2^2\right|$. To show weak tracking, it suffices to upper bound the supremum of $\gamma(f^{(t)})$.

$$\mathbb{E}_\Pi \sup_{t\in[m]} \gamma(f^{(t)}) = \mathbb{E}_\Pi \sup_{t\in[m]} \left|\|\Pi f^{(t)}\|_2^2 - \|f^{(t)}\|_2^2\right|. \tag{1}$$

The first observation[8] is that one can rewrite the error $\gamma(\mathbf{x})$ as follows.

$$\gamma(\mathbf{x}) = \left|\mathbf{x}^\top\Pi^\top\Pi\mathbf{x} - \mathbf{x}^\top\mathbf{x}\right| = \left|\sigma^\top B_{\eta,\mathbf{x}}\sigma - \mathbf{x}^\top\mathbf{x}\right| = \left|\sigma^\top \tilde{B}_{\eta,\mathbf{x}}\sigma\right|,$$

where $\sigma \in \{-1,1\}^n$ is an independent Rademacher random vector and for any $i, i' \in [n]$,

$$(\tilde{B}_{\eta,\mathbf{x}})_{i,i'} = \begin{cases} \mathbf{x}_i\mathbf{x}_{i'}, & i \neq i' \text{ and } \exists j \in [k],\ \eta_{j,i} = \eta_{j,i'} = 1 \\ 0, & \text{else.} \end{cases}$$

Note that the diagonals of $\tilde{B}_{\eta,\mathbf{x}}$ are all zero as follow.

$$\tilde{B}_{\eta,\mathbf{x}} = \begin{pmatrix} 0 & \mathbf{x}_1\mathbf{x}_2\langle\Pi_1,\Pi_2\rangle & \cdots & \mathbf{x}_1\mathbf{x}_n\langle\Pi_1,\Pi_n\rangle \\ \mathbf{x}_2\mathbf{x}_1\langle\Pi_2,\Pi_1\rangle & 0 & \cdots & \mathbf{x}_2\mathbf{x}_n\langle\Pi_2,\Pi_n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_n\mathbf{x}_1\langle\Pi_n,\Pi_1\rangle & \mathbf{x}_n\mathbf{x}_2\langle\Pi_n,\Pi_2\rangle & \cdots & 0 \end{pmatrix}.$$

[8] Note that the matrix $\tilde{B}_{\mathbf{x}}$ we are using is different from the matrix used in the previous analysis of [4]. This difference is crucial since the matrix of [4] does not work for CountSketch.

For convenience, for any matrix $B \in \mathbb{R}^{n\times n}$, we overload the notation γ by denoting $\gamma(B) = \sigma^\top B\sigma$. That is, $\gamma(\tilde{B}_{\eta,\mathbf{x}}) = \gamma(\mathbf{x})$. One benefit of writing ℓ_2 weak tracking error into the above quadratic form is that Hanson-Wright inequality (see Lemma 7) is now applicable.

The lemma below shows that the expectation of the weak tracking error is upper bounded by the Frobenius norm of $\tilde{B}_{\eta,f^{(m)}}$.

▶ **Lemma 10.** *Let $\{f^{(t)}\}_{t\in[m]}$ be the frequencies of an insertion-only stream. We have*

$$\mathbb{E}\left[\sup_{t\in[m]} \gamma(f^{(t)}) \mid \eta\right] = O(\|\tilde{B}_{\eta,f^{(m)}}\|_F).$$

The proof of Lemma 10 uses the Dudley-like chaining argument. For the smooth of presentation, we postpone the details to Subsection 3.2. Next, the following lemma shows that for any vector $x \in \mathbb{R}^n$, with high probability, $\|\tilde{B}_{\eta,x}\|_F = O(\|x\|_2^2/\sqrt{k})$.

▶ **Lemma 11.** *For any $\delta \in (0,1)$ and $x \in \mathbb{R}^n$,*

$$\Pr\left[\|\tilde{B}_{\eta,x}\|_F > \frac{\sqrt{2}\|x\|_2^2}{\sqrt{\delta\cdot k}}\right] \le \frac{\delta}{2}.$$

Lemma 11 has similar flavor as Lemma 4. The proof can be found in Subsection 3.2. Finally, Theorem 8 is an immediate corollary of Lemma 10 and Lemma 11. Here we provide a proof for completeness.

Proof of Theorem 8. Recall that to prove Theorem 8, it suffices to show that with probability at least $1-\delta$ over η, $\sup_{t\in[m]}\gamma(f^{(t)}) \le \epsilon$. From Lemma 10, for a fixed η, we have $\Pr\left[\sup_{t\in[m]}\gamma(f^{(t)}) > C_1\|\tilde{B}_{\eta,f^{(m)}}\|_F\right] \le \delta/2$ for some constant $C_1 > 0$. Next, from Lemma 11, we have $\|\tilde{B}_{\eta,f^{(m)}}\|_F \le \|f^{(m)}\|_2^2\cdot k^{-1/2}\cdot\delta^{-1/2}$ with probability at least $1-\delta/2$ over the randomness in η for some constant $C_2 > 0$. Pick $m \ge C_1C_2\cdot\epsilon^{-2}\cdot\delta^{-1}$, we have $\Pr\left[\sup_{t\in[m]}\gamma(f^{(t)}) > \epsilon\|f^{(m)}\|_2^2\right] \le \delta$ and complete the proof. ◀

3.2 Proof of the two key lemmas

In this subsection, we provide the proofs for Lemma 10 and Lemma 11. Let us start with Lemma 10 which shows that the tracking error can be upper bounded by the Frobenius norm of $\tilde{B}_{\eta,f^{(m)}}$.

Proof of Lemma 10. Recall that we define $\tilde{B}_{\eta,x}$ such that $\gamma(x) = \sigma^\top\tilde{B}_{\eta,x}\sigma$ where σ is 8-wise independent Rademacher random vector. An important trick here is that we think of *fixing*[9] η in the following.

The starting point of chaining argument is constructing a sequence of ϵ-nets with exponentially decreasing error for $\{\tilde{B}_{\eta,f^{(t)}}\}_{t\in[m]}$. Note that here $\{\tilde{B}_{\eta,f^{(t)}}\}_{t\in[m]}$ are matrices but one can view it as a vector and apply Lemma 6 where ℓ_2 norm for a vector becomes Frobenius norm for a matrix. Namely, for any non-negative integer ℓ, let $T_{\eta,\ell}$ be the $(\|\tilde{B}_{\eta,f^{(m)}}\|_F/2^\ell)$-net for $\{\tilde{B}_{\eta,f^{(t)}}\}_{t\in[m]}$ under Frobenius norm where $|T_{\eta,\ell}| \le 1+2^{2\ell}$. Note that here we fixed η first and then constructed the nets. Thus, for each $t\in[m]$, one can rewrite $\tilde{B}_{\eta,f^{(t)}}$ into a *chain* as follows.

$$\tilde{B}_{\eta,f^{(t)}} = B^{(t)}_{\eta,0} + \sum_{\ell=1}^{\infty} B^{(t)}_{\eta,\ell} - B^{(t)}_{\eta,\ell-1}, \tag{2}$$

[9] We do this by conditioning on η.

where $B^{(t)}_{\eta,\ell} \in T_{\eta,\ell}$ and $\|\tilde{B}_{\eta,f^{(t)}} - B^{(t)}_{\eta,\ell}\|_F \le 2^{-\ell} \cdot \|\tilde{B}_{\eta,f^{(m)}}\|_F$. Moreover, from Equation 2 we have

$$\mathbb{E} \sup_{t\in[m]} \gamma(f^{(t)}) \le \mathbb{E} \sup_{t\in[m]} \gamma(B^{(t)}_{\eta,0}) + \sum_{\ell=1}^{\infty} \mathbb{E} \sup_{t\in[m]} \gamma(B^{(t)}_{\eta,\ell} - B^{(t)}_{\eta,\ell-1}). \tag{3}$$

To bound the first term of Equation 3, observe that $T_{\eta,0} = \{\tilde{B}_{\eta,f^{(1)}}\}$ where $\tilde{B}_{\eta,f^{(1)}}$ is the all zero matrix. Namely, the first term of Equation 3 is zero. As for the second term of Equation 3, we apply the chaining argument as follows. For any positive integer ℓ, denote $\mathcal{A}_\ell = \{B^{(t)}_{\eta,\ell} - B^{(t)}_{\eta,\ell-1}\}_{t\in[m]}$. Note that from the construction of ϵ-net in Lemma 6, we have $|\mathcal{A}_\ell| \le 2|T_{\eta,\ell}| \le 2^{2\ell+2}$ by triangle inequality.

$$\begin{aligned}\mathbb{E}\left[\sup_{t\in[m]} \gamma(B^{(t)}_{\eta,\ell} - B^{(t)}_{\eta,\ell-1})\right] &= \int_0^\infty \Pr\left[\sup_{A\in\mathcal{A}_\ell} \gamma(A) > u\right] du \\ &\le u^*_\ell + \int_{u^*_\ell}^\infty \Pr\left[\sup_{A\in\mathcal{A}_\ell} \gamma(A) > u\right] du, \end{aligned} \tag{4}$$

where $u^*_\ell > 0$ will be chosen later. For any $A \in \mathcal{A}_\ell$ and integer $p \ge 2$, by Markov's inequality and Hanson-Wright inequality, we have

$$\Pr[\gamma(A) > u] \le \frac{\mathbb{E}[\gamma(A)^p]}{u^p} = \frac{\|\sigma^\top A\sigma\|_p^p}{u^p} \le \frac{\left(C\cdot\sqrt{p}\|A\|_F + C\cdot p\|A\|\right)^p}{u^p}$$

for some constant $C > 0$. Note that the randomness here is only in σ and thus we can apply the Hanson-Wright inequality. Let $R_\ell = \sup_{A\in\mathcal{A}_\ell}\left(C\cdot\sqrt{p}\|A\|_F + C\cdot p\|A\|\right) \le C'p\cdot\|\tilde{B}_{\eta,f^{(m)}}\|_F\cdot 2^{-\ell}$ for some $C' > 0$. The last inequality holds because of $\|\cdot\| \le \|\cdot\|_F$ and the choice of ϵ-net. Now, choose $u^*_\ell = 2S_\ell \cdot R_\ell$ where S_ℓ will be decided later, Equation 4 becomes

$$\begin{aligned}\mathbb{E}\left[\sup_{t\in[m]} \gamma(B^{(t)}_{\eta,\ell} - B^{(t)}_{\eta,\ell-1})\right] &\le u^*_\ell + \int_{u^*_\ell}^\infty |\mathcal{A}_\ell| \cdot \frac{{R_\ell}^p}{u^p} du \\ &\le 2S_\ell R_\ell + |\mathcal{A}_\ell| \cdot \frac{{R_\ell}^p}{(2S_\ell R_\ell)^{p-1}} \\ &\le 2S_\ell C'p \cdot \|\tilde{B}_{\eta,f^{(m)}}\|_F\cdot 2^{-\ell} + |\mathcal{A}_\ell| \cdot \frac{C'p\cdot\|\tilde{B}_{\eta,f^{(m)}}\|_F}{S_\ell^{p-1}}\cdot 2^{-\ell}\end{aligned} \tag{5}$$

where the second term of Equation 5 is due to union bound. Now, Equation 3 becomes

$$\begin{aligned}\mathbb{E} \sup_{t\in[m]} \gamma(f^{(t)}) &\le \sum_{\ell=1}^\infty 2S_\ell C'p \cdot \|\tilde{B}_{\eta,f^{(m)}}\|_F\cdot 2^{-\ell} + |\mathcal{A}_\ell| \cdot \frac{C'p\cdot\|\tilde{B}_{\eta,f^{(m)}}\|_F}{S_\ell^{p-1}}\cdot 2^{-\ell} \\ &\le \|\tilde{B}_{\eta,f^{(m)}}\|_F \cdot \left(\sum_{\ell=1}^\infty 2C'pS_\ell\cdot 2^{-\ell} + \frac{2^\ell C'p}{S_\ell^{p-1}}\right).\end{aligned} \tag{6}$$

Choose $S_\ell = 2^{3\ell/4}$ and $p \ge 4$, the summation term in Equation 6 can thus be upper bounded by a constant. We conclude that

$$\mathbb{E} \sup_{t\in[m]} \gamma(f^{(t)}) = O(\|\tilde{B}_{\eta,f^{(m)}}\|_F).$$

Note that this also means that 8-wise independence suffices and thus the sketching matrix can be efficiently stored (see Appendix A for more details). ◀

Next, we prove Lemma 11 which upper bounds the expectation of $\|\tilde{B}_{\eta,\mathbf{x}}\|$ for any $\mathbf{x} \in \mathbb{R}^n$.

Proof of Lemma 11. We first show that $\mathbb{E}_\eta \|\tilde{B}_{\eta,x}\|_F^2 \leq \frac{\|x\|_2^4}{k}$ and the lemma immediately holds due to Markov's inequality.

Let $\mathbf{1}_{ii'}$ be the indicator for whether there exists $j \in [k]$ such that $\eta_{ij} = \eta_{i'j} = 1$. Note that for $i \neq i'$, $\mathbb{E}[\mathbf{1}_{ii'}] = 1/k$ and the only randomness here is in η.

$$\begin{aligned}\mathbb{E}\|\tilde{B}_{\eta,x}\|_F^2 &= \mathbb{E} \sum_{i,i' \in [n]} (\tilde{B}_{\eta,x})_{i,i'}^2 = \mathbb{E} \sum_{(i,i') \in [n]^2,\ i \neq i'} x_i^2 x_{i'}^2 \mathbf{1}_{ii'} \\ &= \frac{1}{k} \sum_{(i,i') \in [n]^2,\ i \neq i'} x_i^2 x_{i'}^2 \leq \frac{\|x\|_2^4}{k},\end{aligned}$$

where the last inequality is by Cauchy-Schwarz. Note that 8-wise independence is sufficient in the above argument. ◀

▶ Remark 12. Here, let us briefly compare the difference between our techniques and that in [4]. There are two key observations on the structure of the sketching matrix of CountSketch. First, we observe that the Frobenius norm of $\Pi^\top \Pi$ is dominated by its diagonal and thus *removing* the diagonal would give us a more accurate analysis on the contribution from the off-diagonal term. However, removing the diagonal of $\Pi^\top \Pi$ destroys the symmetric structure and thus the standard ϵ-net argument (e.g., in [4]) would not work. To overcome this, we observe that one can directly construct ϵ-net for the matrix obtained by removing the diagonal from $\Pi^\top \Pi$. Combining these two observations and standard chaining argument, we are able to show that CountSketch provides ℓ_2 weak tracking.

4 Strong tracking of AMS sketch and CountSketch

In this section, we are going to discuss the strong tracking of AMS sketch and CountSketch. We start with a standard reduction from weak tracking to strong tracking via union bound. This gives us an $O(\log m)$ blow-up in the dependency on δ. Next, we show that this is essentially tight for both AMS sketch and CountSketch up to a logarithmic factor.

▶ **Lemma 13** (folklore). *For any $\epsilon > 0$, $\delta \in (0,1)$, and $n, m \in \mathbb{N}$. If a linear sketch provides (ϵ, δ) weak tracking for length m inputs having value from $[n]$, then it also provides $(2\epsilon, \delta')$ strong tracking where $\delta' = \min\{1, (\log m) \cdot \delta\}$.*

Proof. See Subsection B.1 for details. ◀

From Lemma 13, we immediate have the following corollaries.

▶ **Corollary 14.** *For any $\epsilon > 0$ and $\delta \in (0,1)$, AMS sketch with $O\left(\epsilon^{-2}(\log\log m + \log(1/\delta))\right)$ rows provides ℓ_2 (ϵ, δ)-strong tracking.*

▶ **Corollary 15.** *For any $\epsilon > 0$ and $\delta \in (0,1)$, CountSketch with $O\left(\epsilon^{-2}\delta^{-1}\log m\right)$ rows provides ℓ_2 (ϵ, δ)-strong tracking.*

▶ Remark. After applying median trick on CountSketch, the dependency of the number of rows on δ becomes $O(\log(1/\delta))$ and thus $O\left(\epsilon^{-2}(\log\log m + \log(1/\delta))\right)$ rows suffices to achieve ℓ_2 (ϵ, δ)-strong tracking.

In the following, we are going to show that the above two upper bounds are essentially tight for these two algorithms.

▶ **Theorem 16.** *There exists constants $C > 0$ such that for any $\epsilon \in (0, 0.1)$ and $\delta \in (0, 1)$, there exists $N_0 \in \mathbb{N}$ such that if $k < C \cdot \left(\log \frac{\log m}{\log(1/\epsilon)} + \log(1/\delta)\right)$ and $N_0 \leq n \leq m$, then fully independent AMS sketch with k rows does not provide ℓ_2 (ϵ, δ)-strong tracking.*

That is, AMS sketch requires $\tilde{\Omega}\left(\epsilon^{-2}(\log\log m + \log(1/\delta))\right)$ rows to achieve ℓ_2 (ϵ, δ)-strong tracking. Interestingly, the hard instance for AMS sketch to achieve strong tracking is simply the stream consisting all distinct elements. See Subsection B.2 for details.

▶ **Theorem 17.** *There exists a constant $C > 0$ such that for any $\epsilon \in (0, 0.5)$, and $\delta \in (0, 1)$, there exists $N_0 \in \mathbb{N}$ such that if $k \leq C \cdot \epsilon^{-2}\delta^{-1}\frac{\log m}{\log(1/\epsilon)}$ and $N_0 \leq n \leq O(\log m)$, then CountSketch with k rows does not provide ℓ_2 (ϵ, δ)-strong tracking.*

That is, CountSketch requires $\tilde{\Omega}(\epsilon^{-2}\delta^{-1}\log m)$ rows to achieve ℓ_2 (ϵ, δ)-strong tracking. The hard instance for CountSketch is more complicated than that of AMS sketch. See Subsection B.3 for details.

5 Conclusion

In this work, we showed that CountSketch provides ℓ_2 weak tracking with update time having no dependence on the error parameter ϵ. We also give almost tight ℓ_2 strong tracking lower bounds for AMS sketch and CountSketch.

An immediate open problem after this work would be tracking ℓ_p with faster update time for $0 < p < 2$. The ℓ_p estimation problem had been solved by Indyk [12] via *p-stable sketch* and was proven to provide weak tracking by Błasiok et al. [3]. However, same as AMS sketch, the p-stable sketch is dense and has update time $\Omega(\epsilon^{-2})$. Nevertheless, Kane et al. [18] gave a space-optimal algorithm for ℓ_p estimation problem with update time $O(\log^2(1/\epsilon)\log\log(1/\epsilon))$. It would be interesting to see if their algorithm also provides ℓ_p weak tracking.

References

1. Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 20–29. ACM, 1996.
2. Andrew C Berry. The accuracy of the Gaussian approximation to the sum of independent variates. *Transactions of the american mathematical society*, 49(1):122–136, 1941.
3. Jaroslaw Blasiok, Jian Ding, and Jelani Nelson. Continuous Monitoring of l_p Norms in Data Streams. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 81. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
4. Vladimir Braverman, Stephen R Chestnut, Nikita Ivkin, Jelani Nelson, Zhengyu Wang, and David P Woodruff. BPTree: An ℓ_2 heavy hitters algorithm using constant memory. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 361–376. ACM, 2017.
5. Vladimir Braverman, Stephen R Chestnut, Nikita Ivkin, and David P Woodruff. Beating CountSketch for heavy hitters in insertion streams. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 740–753. ACM, 2016.
6. Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming*, pages 693–703. Springer, 2002.
7. Graham Cormode and Shan Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005.

8 Anirban Dasgupta, Ravi Kumar, and Tamás Sarlós. A sparse Johnson-Lindenstrauss transform. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 341–350. ACM, 2010.

9 Carl-Gustaf Esseen. *On the Liapounoff limit of error in the theory of probability.* Almqvist & Wiksell Stockholm, 1942.

10 David Lee Hanson and Farroll Tim Wright. A bound on tail probabilities for quadratic forms in independent random variables. *The Annals of Mathematical Statistics*, 42(3):1079–1083, 1971.

11 Zengfeng Huang, Wai Ming Tai, and Ke Yi. Tracking the Frequency Moments at All Times. *arXiv preprint*, 2014. `arXiv:1412.1763`.

12 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *Journal of the ACM (JACM)*, 53(3):307–323, 2006.

13 Tadeusz Inglot and Teresa Ledwina. Asymptotic optimality of new adaptive test in regression model. *Annales de l'Institut Henri Poincare (B) Probability and Statistics*, 42(5):579–590, 2006.

14 T. S. Jayram and David P. Woodruff. Optimal Bounds for Johnson-Lindenstrauss Transforms and Streaming Problems with Subconstant Error. *ACM Trans. Algorithms*, 9(3):26:1–26:17, June 2013. `doi:10.1145/2483699.2483706`.

15 Daniel Kane, Raghu Meka, and Jelani Nelson. Almost optimal explicit Johnson-Lindenstrauss families. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 628–639. Springer, 2011.

16 Daniel M Kane and Jelani Nelson. Sparser Johnson-Lindenstrauss transforms. *Journal of the ACM (JACM)*, 61(1):4, 2014.

17 Daniel M Kane, Jelani Nelson, Ely Porat, and David P Woodruff. Fast moment estimation in data streams in optimal space. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 745–754. ACM, 2011.

18 Daniel M Kane, Jelani Nelson, Ely Porat, and David P Woodruff. Fast moment estimation in data streams in optimal space. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 745–754. ACM, 2011.

19 Daniel M Kane, Jelani Nelson, and David P Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 41–52. ACM, 2010.

20 Daniel M Kane, Jelani Nelson, and David P Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1161–1178. Society for Industrial and Applied Mathematics, 2010.

21 Balachander Krishnamurthy, Subhabrata Sen, Yin Zhang, and Yan Chen. Sketch-based change detection: methods, evaluation, and applications. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 234–247. ACM, 2003.

22 Kasper Green Larsen, Jelani Nelson, and Huy L Nguyên. Time lower bounds for nonadaptive turnstile streaming algorithms. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 803–812. ACM, 2015.

23 Mikkel Thorup and Yin Zhang. Tabulation based 4-universal hashing with applications to second moment estimation. In *SODA*, volume 4, pages 615–624, 2004.

24 Mikkel Thorup and Yin Zhang. Tabulation-based 5-independent hashing with applications to linear probing and second moment estimation. *SIAM Journal on Computing*, 41(2):293–331, 2012.

A Implementation of CountSketch

Here, we present the implementation of CountSketch for the completeness. Note that the construction is standard and not new.

Algorithm 1 Constructing CountSketch.

$k \leftarrow \lceil \frac{c}{\epsilon^2} \rceil$ for some constant $c > 0$.
$\tilde{f} \in \mathbb{Z}^k$ vector with initial value 0.
Sample $h : [n] \to [k]$ from a 8-wise independent hash family.
Sample $g : [n] \to \{\pm 1\}$ from a 8-wise independent hash family.
for $t = 1, 2, \ldots, m$ **do**
    On input $a_t = i$, set $\tilde{f}_{h(i)} = \tilde{f}_{h(i)} + g(i)$.

Note that both h and g can be stored in space $O(\log n + \log(1/\epsilon))$ and be evaluated in $O(1)$ many arithmetic operations. $\tilde{f}$ can be stored in space $O(\epsilon^{-2} \log m)$ bits. For the convenience of analysis, we define the sketching matrix $\Pi \in \{0, \pm 1\}^{k \times n}$ of CountSketch by $\Pi_{h(i),i} = g(i)$ for all $i \in [n]$.

B Proofs for strong tracking

B.1 From weak tracking to strong tracking

After applying union bound on all points $t = 1, 2, \ldots, m$, a streaming algorithm provides ℓ_2 (ϵ, δ)-approximation also provides ℓ_2 (ϵ, δ')-strong tracking where $\delta' = \min\{1, m\delta\}$. However, the blow-up in δ is m, which is undesirable. The following lemma shows that with a more delicate union bound argument, the reduction from weak tracking to strong tracking only has $O(\log m)$ blow-up in δ. Note that the lemma is a folklore and we provide a proof for completeness.

Proof. Let $\{f^{(t)}\}_{t\in[m]}$ be the frequency of an insertion-only stream and let $\{\tilde{f}^{(t)}\}_{t\in[m]}$ be its (randomized) approximations produced by the linear sketch. Let $w = \lfloor \log m \rfloor + 1$ and $t_i = 2^i - 1$ for each $i \in [w]$. Note that for each $i \in [w]$ and $t_{i-1} < t \le t_i$, $\frac{1}{2}\|f^{(t_i)}\|_2^2 \le \|f^{(t)}\|_2^2 \le \|f^{(t_i)}\|_2^2$. Define the event

$$E_i := \left\{ \|\tilde{f}^{(t_i)}\|_2^2 - \|f^{(t_i)}\|_2^2| > \epsilon \|f^{(t_i)}\|_2^2 \right\}.$$

Observe that for each $t_{i-1} < t \le t_i$, $|\|\tilde{f}^{(t)}\|_2^2 - \|f^{(t_i)}\|_2^2| > 2\epsilon \cdot \|f^{(t)}\|_2^2$ would imply $\neg E_i$. Namely, $\neg \cup_{i\in[w]} E_i$ implies strong tracking.

By the ℓ_2 (ϵ, δ)-weak tracking property of the streaming algorithm, for each $i \in [w]$, we have $\Pr[E_i] \le \delta$ and thus $\Pr[\cup_{i\in[w]} E_i] \le w\delta$. We conclude that the streaming algorithm provides ℓ_2 $(2\epsilon, w\delta)$-strong tracking. ◀

B.2 Strong tracking lower bound for AMS sketch

The hard instance is simply the stream of all distinct elements, *i.e.*, $i_t = t$ for all $t \in [m]$.

Proof of Theorem 16. Consider the stream of all distinct elements as the hard instance, *i.e.*, $i_t = t$ for all $t \in [m]$. Thus, $\|f^{(t)}\|_2^2 = t$ and $\|\Pi f^{(t)}\|_2^2 = \sum_{i\in[k]} \left(\sum_{j\in[t]} \Pi_{i,j}\right)^2$ for all $t \in [m]$.

Define a sequence of time $\{t_j\}$ as follows. $t_0 = 0$ and $t_j = \sum_{i\in[j]} \Delta_i$ where $\Delta_i = \lceil 10/\epsilon \rceil^i$. Pick ℓ and m properly such that $t_\ell \le m$. Some quick facts about the choice of parameters here: (i) $|t_j - \Delta_j| \le \frac{\epsilon}{5} \cdot t_j$. (ii) $\ell = \Theta(\frac{\log m}{\log(1/\epsilon)})$.

To show AMS sketch does not provide (ϵ, δ)-strong tracking for $\epsilon \in (0, 0.1)$ and $\delta \in (0, 1)$, it suffices to show that with probability at least δ there exists $j \in [\ell]$ such that $\|\Pi f^{(t_j)}\|_2^2 - t_j > (1+\epsilon) \cdot t_j$.

For the convenience of the analysis, for any $i \in [k]$ and $j \in [\ell]$, let $X_i^{(t_j)} = \sum_{s=t_{j-1}+1}^{t_j} \Pi_{i,s}$ which is the sum of Δ_j independent Rademacher random variables divided by $\sqrt{k}$. Also let $Z_j = \sum_{i\in[k]} (X_i^{(t_j)})^2$. Note that $\mathbb{E}[Z_j] = \Delta_j/\sqrt{k}$ and

$$\begin{aligned}\|\Pi f^{(t_j)}\|_2^2 &= \sum_{i\in[k]} \left(\sum_{j'\in[j]} X_i^{(t_{j'})} \right)^2 \\ &= Z_j + \sum_{i\in[k]} \left(\sum_{j'\in[j-1]} X_i^{(t_{j'})} \right)^2 + 2 \sum_{i\in[k]} \langle X_i^{(t_j)}, \sum_{j'\in[j-1]} X_i^{(t_{j'})} \rangle. \end{aligned} \tag{7}$$

Define an event $E_j := \{Z_j \ge (1+2\epsilon) \cdot \mathbb{E}[Z_j]\}$ for each $j \in [\ell]$. Observe that when conditioning on $\cap_{j'\in[j-1]} \neg E_{j'}$, the second term of Equation 7 is bounded by $O(t_{j-1})$ and the third term is bounded by $O(\sqrt{t_{j-1} Z_j})$ due to Cauchy-Schwarz. By the choice of parameters, both term can be bounded by $0.1 t_j$. Furthermore, E_j implies $\|\Pi f^{(t_j)}\|_2^2 - t_j > (1+\epsilon) \cdot t_j$. Note that E_j is independent to $E_1, \ldots, E_{j-1}$. The following lemma lower bound the probability of E_j to happen.

▶ **Lemma 18.** *There exists a constant $c > 0$ such that $\Pr[E_j] \ge e^{-c\epsilon^2 k}$ for any $j = \Omega(\log\log k)$.*

Proof of Lemma 18. From the seminal *Berry-Esseen theorem* [2, 9], we know that when $t_j = e^{\Omega(k)} = \Omega(\frac{\log m}{\delta})$ then $X^{(t_j)}$ is point-wisely $e^{-\Omega(k)}$-close to a normal distribution with zero mean and variance Δ_j. That is, $\frac{kZ_j}{\Delta_j}$ is also point-wisely $e^{-\Omega(k)}$-close to a *chi-square* distribution $\chi^2_{\Delta_j}$ with mean Δ_j and Δ_j degree of freedom[10].

Inglot and Ledwina [13] showed that the tail of chi-square random distribution can be lower bounded as $\Pr[\chi_k^2 \ge (1+2\epsilon) \cdot k] \ge \frac{1}{2} e^{-\epsilon^2 k/10}$ when k large enough. Combine with the Berry-Esseen theorem, we have $\Pr[E_j] \ge e^{-c\epsilon^2 k}$ for some constant $c > 0$. ◀

Note that as $\{Z_j\}_{j\in[\ell]}$ are mutually independent, the events $\{E_j\}_{j\in[\ell]}$ are also mutually independent. That is,

$$\begin{aligned}\Pr\left[\exists t \in [m],\ \left|\|\Pi f^{(t)}\|_2^2 - \|f^{(t)}\|_2^2\right| > 2\epsilon \|f^{(t)}\|_2^2\right] &\ge \Pr\left[\cup_{j\in[\ell]} E_j\right] \\ &\ge 1 - \prod_{j\in[\ell]} \Pr\left[\neg E_j \mid \neg E_{j'},\ \forall j' \in [j-1]\right] \\ &\ge 1 - \left(1 - e^{-c\epsilon^2 k}\right)^\ell \ge \ell e^{-c\epsilon^2 k}.\end{aligned}$$

◀

Namely, there exists another constant $C > 0$ such that if $k < C\epsilon^{-2}\left(\log \frac{\log m}{\log(1/\epsilon)} + \log(1/\delta)\right) \le \frac{1}{c}\epsilon^{-2} \log \frac{\ell}{\delta}$. Thus, AMS sketch does not provide (ϵ, δ)-strong tracking for all $\epsilon \in (0, 0.1)$.

[10] Recall that a *chi-square random variable* of d degree of freedom is equivalent to the sum of d squares of the standard normal random variable.

B.3 Strong tracking lower bound for CountSketch

To prove Theorem 17, we are going to construct a stream such that any CountSketch does not provide strong tracking. Let's start from some observation. For any $i \neq i' \in [n]$ and $a > 0$, let $\mathbf{x} = a(\mathbf{e}_i + \mathbf{e}_{i'})$ such that $\|\mathbf{x}\|_2^2 = 2a^2$. Now, observe that If $\Pi_i = \Pi_{i'}$, then we have $\|\Pi\mathbf{x}\|_2^2 = 4a^2$. If $\Pi_i = -\Pi_{i'}$, then we have $\|\Pi\mathbf{x}\|_2^2 = 0$. Note that in both cases, the approximation $\|\Pi\mathbf{x}\|_2^2$ and the correct answer $\|\mathbf{x}\|_2^2$ has a huge gap $2a^2$, *i.e.*, $\left|\|\Pi\mathbf{x}\|_2^2 - \|\mathbf{x}\|_2^2\right| \geq \|\mathbf{x}\|_2^2$.

With the above observation, one can see that a collision (either $\Pi_i = \Pi_{i'}$ or $\Pi_i = -\Pi_{i'}$) is a sufficient condition for an estimation error. As a result, to show CountSketch does not provide strong tracking, it suffices to show the following two things: (i) there will be some collision with constant probability and (ii) construct a stream such that once a collision happens, the estimation error is large.

Note that (ii) is very specific to tracking since unlike ℓ_2 estimation which only cares about the final estimation, we need to keep track of the estimation at any time. Thus, to show the impossibility of tracking, we have to show that the estimation fails at least once at some point.

Proof of Theorem 17. Let n be the number of elements and k be the number of rows of CountSketch. Let $\Delta = \lceil 100/\epsilon \rceil$ and $w = \lceil 1/\epsilon \rceil$. For any $j \in [\ell]$, define $t_j = \sum_{j' \in [j]} \Delta^{j'+1} = \frac{\Delta^{j+1} - \Delta^1}{\Delta - 1}$ and the stream at time t_j as follows.

$$f^{(t_j)} = \left(\underbrace{\Delta, \ldots, \Delta}_{w}, \underbrace{\Delta^2, \ldots, \Delta^2}_{w}, \underbrace{\Delta^j, \ldots, \Delta^j}_{w}, 0, \ldots, 0 \right).$$

We have $\|f^{(t_j)}\|_2^2 = \sum_{j' \in [j]} w \cdot \Delta^{2j'+1} = \frac{w \cdot \Delta^{2j+2} - w \cdot \Delta^2}{\Delta^2 - 1}$. Note that one can easily complete rest of the stream $\{f^{(t)}\}_{t \in [m]}$ for any $m \geq t_\ell$. Note that here we can pick $\ell = \Theta(\frac{\log m}{\log(1/\epsilon)})$.

Define the event $E_j := \{\|\Pi f^{(t_j)}\|_2^2 - \|f^{(t_j)}\|_2^2 > \epsilon \cdot \|f^{(t_j)}\|_2^2\}$. To show that CountSketch does not provide w_2 (ϵ, δ)-strong tracking, it suffices to prove $\Pr[\cup_{j \in [\ell]} E_j] > \delta$. The following lemma lower bounds the probability of single E_j.

▶ **Lemma 19.** *For each $j \in \ell$, we have* $\Pr[E_j \mid \neg \cup_{j' \in [j]} E_{j'}] \geq \frac{1}{10k\epsilon^2}$.

Proof. First, let $\bar{f}^{(t_j)} = f^{(t_j)} - f^{(t_{j-1})}$ for each $j \in \ell$ where we define $f^{(0)} = \mathbf{0}$. Observe that

$$\begin{aligned}
\|\Pi f^{(t_j)}\|_2^2 - \|f^{(t_j)}\|_2^2 &= \|\Pi \bar{f}^{(t_j)} + \Pi f^{(t_{j-1})}\|_2^2 - \|\bar{f}^{(t_j)} + f^{(t_{j-1})}\|_2^2 \\
&= \|\Pi \bar{f}^{(t_j)}\|_2^2 - \|\bar{f}^{(t_j)}\|_2^2 + \|\Pi f^{(t_{j-1})}\|_2^2 - \|f^{(t_{j-1})}\|_2^2 \\
&\quad + 2\langle \Pi \bar{f}^{(t_j)}, \Pi f^{(t_{j-1})} \rangle - 2\langle \bar{f}^{(t_j)}, f^{(t_{j-1})} \rangle.
\end{aligned}$$

Further, condition on $\neg \cup_{j' \in [j-1]} E_{j'}$, we have $\|f^{(t_{j-1})}\|_2^2$, $\|\Pi f^{(t_{j-1})}\|_2^2$, $|\langle \Pi \bar{f}^{(t_j)}, \Pi f^{(t_{j-1})} \rangle|$, and $|\langle \bar{f}^{(t_j)}, f^{(t_{j-1})} \rangle|$ are all at most $(\epsilon/10) \cdot \|f^{(t_j)}\|_2^2$ by the choice of Δ. Namely,

$$\|\Pi f^{(t_j)}\|_2^2 - \|f^{(t_j)}\|_2^2 \geq \|\Pi \bar{f}^{(t_j)}\|_2^2 - \|\bar{f}^{(t_j)}\|_2^2 - \frac{\epsilon}{2} \cdot \|f^{(t_j)}\|_2^2. \tag{8}$$

▶ **Lemma 20.** $\Pr\left[\|\Pi \bar{f}^{(t_j)}\|_2^2 - \|\bar{f}^{(t_j)}\|_2^2 > 3\epsilon \cdot \|f^{(t_j)}\|_2^2\right] > \frac{1}{10k\epsilon^2}$.

Proof. Let us consider the columns of Π that correspond to the non-zero entries of $\bar{f}^{(t_j)}$. That is, column $\Delta\cdot(j-1)+1$ to $\Delta\cdot j$. Note that once there are exactly one collision happens among these columns and the both the value are the same, then $\|\Pi\bar{f}^{(t_j)}\|_2^2 - \|\bar{f}^{(t_j)}\|_2^2 > 3\epsilon\cdot\|f^{(t_j)}\|_2^2$. The probability of the above to happen is at least the following.

$$\frac{1}{2}\cdot\frac{k\cdot\binom{w}{2}\cdot(k-1)\cdot(k-2)\cdots(k-w+2)}{k^w} \geq \frac{w^2}{5k} > \frac{1}{10k\epsilon^2}. \qquad \blacktriangleleft$$

Now, Lemma 19 immediately follows from Equation 8 and Lemma 20. ◀

Let us wrap up the proof of Theorem 17 as follows.

$$\begin{aligned}\Pr\left[\exists t\in[m],\ \left|\|\Pi f^{(t)}\|_2^2 - \|f^{(t)}\|_2^2 > \epsilon\|f^{(t)}\|_2^2\right|\right] &\geq \Pr\left[\cup_{j\in[\ell]}E_j\right]\\ &= \prod_{j\in[\ell]}\Pr\left[E_j \mid \neg\cup_{j'\in[j-1]}E_{j'}\right]\\ &\geq \left(1-\frac{1}{10k\epsilon^2}\right)^{\ell} \geq 1-\frac{\ell}{k\epsilon^2}.\end{aligned}$$

By the choice of parameters, the last quantity would be greater than δ and thus CountSketch with $k \leq C\cdot\epsilon^{-2}\delta^{-1}\frac{\log(m)}{\log(1/\epsilon)}$ rows does not provide ℓ_2 (ϵ,δ)-strong tracking. ◀

Submodular Optimization with Contention Resolution Extensions



Benjamin Moseley
Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA
Relational AI, Berkeley CA, USA
moseleyb@andrew.cmu.edu

Maxim Sviridenko
Yahoo Research, New York, NY, USA
sviri@oath.com

Abstract

This paper considers optimizing a submodular function subject to a set of downward closed constraints. Previous literature on this problem has often constructed solutions by (1) discovering a fractional solution to the multi-linear extension and (2) rounding this solution to an integral solution via a contention resolution scheme. This line of research has improved results by either optimizing (1) or (2).

Diverging from previous work, this paper introduces a principled method called contention resolution extensions of submodular functions. A contention resolution extension combines the contention resolution scheme into a continuous extension of a discrete submodular function. The contention resolution extension can be defined from effectively any contention resolution scheme. In the case where there is a loss in both (1) and (2), by optimizing them together, the losses can be combined resulting in an overall improvement. This paper showcases the concept by demonstrating that for the problem of optimizing a non-monotone submodular subject to the elements forming an independent set in an interval graph, the algorithm gives a .188-approximation. This improves upon the best known $\frac{1}{2e} \simeq .1839$ approximation.

2012 ACM Subject Classification Mathematics of computing → Combinatorial algorithms

Keywords and phrases Submodular, Optimization, Approximation Algorithm, Interval Scheduling

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.3

Category APPROX

Funding *Benjamin Moseley*: Supported in part by a Google Research Award, a Yahoo Research Award and NSF Grants CCF-1830711, CCF-1824303, and CCF-1733873.

1 Introduction

Submodular function maximization has numerous applications and there has been a rich theory developed on the topic. See [9] for pointers to relevant work. In this problem, the input consists of a universe of n elements U and a submodular set function $f : 2^U \to \mathbb{R}^+$. A function is submodular if for all sets $A, B \subseteq U$ where $A \subseteq B$ and any element $e \in U \setminus B$ it is the case that $f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) + f(B)$.[1] Submodular functions are a general class of functions that capture the concept of diminishing returns. Natural occurrences of submodular functions include the cut function [8] and the coverage function [3]. Due to their generality, submodular functions capture many common objective functions. For example, submodular functions are frequently used in machine learning for problems such as document summarization [18], exemplar clustering [12], influence in social networks [13] and other problems [15].

[1] Equivalently, a function is submodular if for all sets $A, B \subseteq U$ it is the case that $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$.


© Benjamin Moseley and Maxim Sviridenko;
licensed under Creative Commons License CC-BY

The submodular maximization problem is to select a set S maximizing $f(S)$ such that $S \in \mathcal{I}$ where $\mathcal{I}$ is a family of sets of feasible solutions. The set $\mathcal{I}$ is usually assumed to be downward closed.[2] The set of feasible solutions $\mathcal{I}$ is defined based on the constraints of the given problem. Prior work has focused on two cases. In one, the function f is additionally assumed to be *monotone* and in the other the function f is *non-monotone*. A submodular function is monotone if $f(S \cup \{e\}) \geq f(S)$ for all $S \subseteq U$ and $e \in U$. The function f is said to be non-monotone if there is no monotonicity restriction.

Optimizing a submodular function subject to classes of downward closed constraints has been extensively studied [7, 11, 4, 1, 10]. The most widely considered classes of constraints are a cardinality constraint [3], matroid constraints [17], knapsack constraints [16], and interval constraints [9]. Through this line of research, a general algorithmic method has emerged. The method consists of two parts. (1) Find a fractional solution to the multilinear extension, and then (2) use a contention resolution scheme or techniques like pipage rounding [5] to round the fractional solution to a feasible integral solution. The *multilinear extension* is an extension of a discrete submodular set function f to the fractional continuous setting. This algorithmic method is general enough to give strong results for numerous problems, including the best known results for monotone and non-monotone submodular maximization under a single matroid constraint [3, 7, 1].

Several past works have focused on optimizing either steps (1) or (2) to improve state-of-the-art methods. Generally, past work has focused on improving (1), the procedure to construct a fractional solution. This is because [6] gave general methods for converting fractional solutions to the multilinear extension to an integral solution. The algorithm typically used in (1) is the *continuous greedy* algorithm and its variants [2, 4, 19, 11, 7].

The Multi-Linear Extension, Continuous Greedy, and Contention Resolution Schemes. Let F be the multilinear extension of f. The **multilinear extension** F is a continuos function that extends f to the fractional domain $[0,1]^{|U|}$. The input to F is a vector $\mathbf{x}$ where $0 \leq x_i \leq 1$ for all i. Let S contain each element U with probability x_i. The value of $F(\mathbf{x})$ is $\mathbb{E}[f(S)]$. It is important to note that S may not be in $\mathcal{I}$. Past work uses the continuous greedy framework to discover a vector $\mathbf{x}$ such that $F(\mathbf{x})$ is close to the optimal solution. Then, this is rounded to an integral solution using a contention resolution scheme $\mathcal{C}(\mathbf{x})$. The idea is to first construct the set S at random, as is done in the computation of $F(\mathbf{x})$. Then some elements from S are dropped to find a set $S' \subseteq S$ such that $S' \in \mathcal{I}$. Key is showing that $\mathbb{E}[f(S')]$ is close to $F(\mathbf{x})$, and thereby bounding $\mathbb{E}[f(S')]$ by the optimal solution.

The **continuous greedy algorithm** iteratively builds a fractional solution $\mathbf{x}$. The algorithm adds a small fractional amount $\mathbf{x}^*$ of some elements to $\mathbf{x}$ such that it greedily increases $F(\mathbf{x} + \mathbf{x}^*)$. Past work has focused on the optimizing the greedy choice of $\mathbf{x}^*$.

This line of work has mostly focused on optimizing (1). This is due to (1) is being the core part of the algorithm where there is loss in the approximation factor. In many cases though, there is additionally loss when performing (2) as well [6, 9].

Contention Resolution Extensions. As mentioned, past work has focused on optimizing (1) and (2) in isolation. This paper for the first time considers optimizing (1) and (2) together to combine the losses in the two procedures and show overall improved results. Our main results are enabled by a principled algorithmic method called *contention resolution extensions*, going beyond optimizing the multi-linear extension.

[2] A set $\mathcal{I}$ is said to be downward closed if $S \in \mathcal{I}$ implies $S' \in \mathcal{I}$ for all $S' \subseteq S$.

The framework takes as input a (randomized) contention resolution scheme $\mathcal{C}(\mathbf{x})$. The contention resolution scheme takes as input a fractional solution and returns a feasible integral solution. Past work constructs $\mathbf{x}$ and then produces the final solution using $\mathcal{C}$ only in the last step. Instead, this paper uses $\mathcal{C}$ to construct $\mathbf{x}$. At each step the new method greedily selects a small fractional amount of each element $\mathbf{x}^*$ to maximize the expected value of $\mathcal{C}(\mathbf{x}+\mathbf{x}^*)$. When the algorithm terminates, it simply returns $\mathcal{C}(\mathbf{x})$ for the final vector $\mathbf{x}$ computed. In this way, the algorithm's greedy choices at each step are closely connected to the final solution that the algorithm will return.

Improved results can be shown using this framework because the loss in step (1) and (2) can be combined in the analysis. Further, the loss in the contention resolution scheme is optimized over in each step, allowing the algorithm to converge to a fractional solution that is chosen directly to optimize the final solution.

1.1 Applications of the Contention Resolution Extension Framework

This paper shows how contention resolution extensions can be used to improve state-of-the-art results for optimizing submodular functions.

The paper considers the problem of optimizing a submodular function over independent sets in an interval graph. In this problem, each element is associated with an interval. The goal is to select a set of intervals that do not intersect to maximize a non-monotone submodular function. The best known previous result is a $\frac{1}{2e} \simeq .1839$-approximation [9].

▶ **Theorem 1.** *For any non-monotone submodular function where $f(\emptyset) = 0$ there is a .188-approximation algorithm for maximizing the function subject to an interval constraint.*

Overview of the Improved Analysis. To describe how our analysis improves over previous work, first consider the unified continuous greedy algorithm of [11]. Let $\mathcal{C}$ be a contention resolution scheme and OPT denote the value of the optimal solution. As discussed, the algorithm greedily builds a fractional solution $\mathbf{x}$. At each step, an amount $\mathbf{x}^*$ is added to $\mathbf{x}$ where $\mathbf{x}^*$ contains a small amount of some of the elements. Past analysis of the continuous greedy framework proves that in each step $F(\mathbf{x})$ increases by an amount proportional to $(1-||\mathbf{x}||_\infty)\text{OPT}$. That is, the incremental improvement of $F(\mathbf{x})$ at each step is proportional to OPT multiplied by an amount that depends on the most any element is fractionally selected in $\mathbf{x}$. The analysis crucially relies on a bound on $||\mathbf{x}||_\infty$ at each step. The algorithm arrives at the final solution using $\mathcal{C}$ on the vector $\mathbf{x}$ at the end of the continuous greedy procedure. For many contention resolution schemes, the expected value of the solution returned is bounded by $F(\mathbf{x})$ multiplied by the minimum probability an element is not discarded by the contention resolution scheme.

Following the above, notice that improving the bound on $||\mathbf{x}||_\infty$ in each step will improve the overall analysis. Our algorithmic framework will allow us to achieve better bounds on $||\mathbf{x}||_\infty$. In particular, we know that the final solution returned is obtained by running $\mathcal{C}$, which increases the probability that an element is not included in the final solution. If somehow the probability an element is discarded by $\mathcal{C}$ could be incorporated into each step of the algorithm to ensure $||\mathbf{x}||_\infty$ is small, then this would improve the overall analysis.

Our algorithm uses $\mathcal{C}$ at each step in the continuous process of constructing $\mathbf{x}$. In particular, by using $\mathcal{C}$ there is less of a chance an element is selected. For this reason, the analysis effectively gets a tighter bound on $||\mathbf{x}||_\infty$, resulting in an overall improved analysis.

A challenge in this approach is that no prior analysis has considered optimizing $\mathcal{C}(\mathbf{x})$ and have always used $F(\mathbf{x})$. Consequently, our analysis introduces new techniques for optimizing over contention resolutions extensions.

2 Preliminaries

Let f be a non-monotone submodular function. The input to the problem is a universe of n elements S. The goal is to select a set of elements $S' \subseteq \mathcal{I}$ such that $f(S')$ is maximized where $\mathcal{I}$ is a set of feasible solution sets. Let $f_R(S') := f(R \cup S') - f(R)$ be the value of adding elements in the set S' to the set R. In this paper it is assumed that $f(\emptyset) = 0$.

The paper considers a hereditary set system defined by independent sets in interval graphs. In this problem, each element $i \in U$ is an interval $(s_i, d_i]$. A set S' is in $\mathcal{I}$ if no two intervals in S' intersect.

The analysis framework in this paper builds on previous submodular optimization work. The next lemma follows from the contention resolution framework of [6]. It is not proven explicitly, but follows from the proof in the paper. Consider a contention resolution scheme that takes as input a set S' and returns a set $D(S') \subseteq S'$. The scheme is said to be monotonic if the probability an element $i \in D(S'')$ is only greater than the probability $i \in D(S')$ for $S'' \subseteq S'$ and $\{i\} \in S''$.

▶ **Theorem 2** ([6]). *Let S' be a set constructed using a randomized procedure. Consider a deterministic monotonic contention resolution scheme that given a set S' of elements constructs a set $D(S') \subseteq S'$ such that $\mathbf{Pr}[i \in D(S') \mid i \in S'] \geq c$ for all S' and i. Further, there exists an ordering of elements $e_1, e_2, \ldots$ in $D(S')$ such that $f_{e_1,e_2,\ldots e_i}(\{e_{i+1}\}) > 0$ for all $0 \leq i < |D(S')|$. Then it is the case that $c\mathbb{E}[f(S')] \leq \mathbb{E}[f(D(S'))]$.*

The following lemma is implied by a well known relationship between the Lovasz extension and multilinear extension of submodular functions. See [9] and [20]. We prove this here for completeness.

▶ **Theorem 3.** *Let f be a non-negative submodular function with $f(\emptyset) = 0$. Fix any set O. Let R be a set of elements constructed at random where element i is in R with probability p_i. Say that $p_i \leq \alpha$ for all $i \notin O$. It is the case that $\mathbb{E}[f(R \cup O)] \geq (1 - \alpha)f(O)$.*

Proof. Let p_i be the probability that i is in R for $i \notin O$ and let $p_i = 1$ for $i \in O$. Consider ordering all of the intervals so that $p_1 \geq p_2 \geq \ldots \geq p_n$. For notational convienience, assume $p_{n+1} = 0$. Recall that for any sets S' and S'' we set $f_{S'}(S'') = f(S' \cup S'') - f(S')$. In the following $[k]$ is the set $\{1, 2, \ldots, k\}$. Let $R' = R \cup O$ in the following. We see the following.

$$\begin{aligned}
& \mathbb{E}[f(R')] = f(\emptyset) + \sum_{k=1}^{n} \mathbb{E}[f(R' \cap [k]) - f(R' \cap [k-1])] \\
= \; & \sum_{k=1}^{n} \mathbb{E}[f_{R' \cap [k-1]}(R' \cap \{k\})] \geq \sum_{k=1}^{n} \mathbb{E}[f_{[k-1]}(R' \cap \{k\})] \quad [f(\emptyset) = 0 \text{ and submodularity}] \\
= \; & \sum_{k=1}^{n} p_k f_{[k-1]}(k) = \sum_{k=1}^{n} p_k (f([k]) - f([k-1])) = \sum_{k=1}^{n} (p_k - p_{k+1}) f([k]) \\
\geq \; & (1 - \alpha) f(O) \quad [f \text{ is positive and } p_i \leq (1 - \alpha) \text{ for all } i \notin O \text{ by assumption}]
\end{aligned}$$

◀

3 Non-Monotone Function Subject to an Interval Constraint

In this section, we consider the problem of optimizing a non-monotone submodular function f subject to an interval scheduling constraint. In this problem, there is a set S of possible intervals $(s_i, d_i]$. We note that the intervals do not contain their starting point. This is

simply for notational purposes and is without loss of generality. A set S' of intervals is feasible (in $\mathcal{I}$) if no two intersect and the goal is to maximize $f(S')$. It is said that two intervals intersect if they both include a common point.

The algorithm maintains a vector y of size n. Let y^i denote the ith entry in the vector. Intuitively, one can think of the entry y^i as the probability of selecting interval i. The vector y will be chosen such that the following holds. Fix any point t. It is the case that $\sum_{i:t\in(s_i,d_i]} y^i \leq 1$. That is, the total weight of intervals intersecting point t is at most one.

The Function $F(y)$. The function F is defined as follows. A set R of intervals is selected by choosing each interval i with probability y^i. The function $F(y) = \mathbb{E}[f(R)]$. This function is the multi-linear extension. Notice that R may not be in the set of feasible solutions $\mathcal{I}$.

The Function $G(y)$. The function G is constructed similarly to F, but it removes additional intervals from R to get a set $D(R)$. The value of $G(y)$ is set to $\mathbb{E}[f(D(R))]$. Intervals are removed from R so that $D(R)$ forms a feasible solution. In this way, G acts as a contention resolution scheme. Each interval i in R is added to $D(R)$ if there is no other interval in R that intersects the start point s_i of i. This function is a *contention resolution extension*[3] of the set function $f(S)$. Notice that the set $D(R)$ is a feasible solution.

Formally, each interval i is in R with probability y^i. Given R let $D(R) = \{i \in R \mid \forall j \in R, s_i \notin (s_j, d_j]\}$. Set $G(y) = \mathbb{E}[f(D(R))]$.

The Algorithm. The algorithm works as follows. The algorithm continuously optimizes G. At time t a vector y_t has been constructed. Let δ be very small, $\frac{1}{\texttt{poly}(n)}$. The algorithm initializes $y_{t+\delta}$ to y_t and then increases some of the entires. Pseudocode can be found in Algorithm 1. In the following description, for any vector v let $v + 1_i$ denote the vector v except that the coordinate of i is fixed to 1.

Separately for each element i, the algorithm finds the value of $\gamma_i = \sum_{S'\subseteq S} \mathbf{Pr}[R = S']f(D(S' \cup \{i\}))$, equivalently the value of $G(y_t + 1_i)$. This can be estimated to high accuracy following sampling techniques used in previous work [6, 9, 7] and for ease of explanation we assume that it can be computed exactly. Let $\beta_i := \delta e^{-y_t^i}(1 - y_t^i)$ and $w_i = \beta_i(\gamma_i - G(y_t)) = \beta_i(G(y_t + 1_i) - G(y_t))$. The value of w_i is precisely the change in $G(y_t)$ if y_t^i is increased to 1 and then scaled by β_i.

The algorithm finds a maximum weight independent set I over all intervals where an interval i is given weight w_i. It is well known that such a solution can be found in polynomial time using dynamic programming [14]. For each interval $i \in I$, $y_{t+\delta}^i$ is increased by an additive β_i.

The procedure can stop at any time t where $0 \leq t \leq 1$.[4] When the procedure stops, the final solution is produced by constructing $D(R)$ as in the description of G. This set is returned as the solution. This is a feasible solution by construction and the expected value of the algorithm's solution will be $G(y_t)$.

[3] We note that this is not the only contention resolution extension and there are other natural contention resolution schemes that could be used.

[4] One could stop at $t > 1$ so long as the contention resolution scheme constructs a feasible solution. This did not result in improvement in our analysis.

Algorithm 1 Computing $y_{t+\delta}$ from y_t.

for $i \in U$ **do**
  $\gamma_i \leftarrow G(y_t + 1_i)$
  $\beta_i \leftarrow \delta e^{-y_t^i}(1 - y_t^i)$
  $w_i \leftarrow \beta_i(\gamma_i - G(y_t))$ // $= \beta_i(G(y_t + 1_i) - G(y_t))$
end for
Give each interval i a weight of w_i. Using these weights, find a maximum weight subset of intervals I that do not intersect.
for $i \in U$ **do**
  if $i \in I$ **then**
    $y_{t+\delta}^i = y_t^i + \beta_i$
  else
    $y_{t+\delta}^i = y_t^i$
  end if
end for
Output $y_{t+\delta}$

3.1 Analysis

Let O denote the intervals in a fixed optimal solution. For each interval i, let E_i be the set of intervals at or before s_i that intersect i and let i be in E_i. The analysis begins by showing that any single interval is selected with at most a small probability.

▶ **Lemma 4.** *The maximum value an entry in y_t can have is $\alpha(t) := \frac{100(e^{37t/100}-1)}{100e^{37t/100}-63} + 2\delta \leq 1 - e^{-t} + 2\delta t$ for any $0 \leq t \leq 1$.*

Proof. In each step, an interval i chosen to be in I has its probability of selection increased by the algorithm. This increase is at most $\delta(1 - y_t^i)e^{-y_t^i}$ at time t. In the worst case, y_t^i is increased at each time step t. The proof will assume that this is the case for element i. For all $y_t^i \leq 1$, from convexity of $e^{-y_t^i}$ we derive,

$$e^{-y_t^i} \leq 1 - (1 - e^{-1})y_t^i \leq 1 - 0.63y_t^i.$$

We now define a function $\rho^i(t)$ which is a piecewise linear version of y_t^i over times t. Define the function $\rho^i(t)$ for any integer $j \geq 2$ and $t \in [0, 1]$ as follows: for each $t \in [(j-1)\delta, j\delta]$ let $\rho^i(t) = \delta \sum_{\tau=0}^{j-2}(1 - y_{\tau\delta}^i)(1 - 0.63y_{\tau\delta}^i) + (t - (j-1)\delta)(1 - y_{(j-1)\delta}^i)(1 - 0.63y_{(j-1)\delta}^i)$. Set $\rho^i(0) = 0$. Obviously $y_{\tau\delta}^i \leq \rho^i(t)$ when $t \leq \tau\delta$ and $y_{(\tau+1)\delta}^i \leq y_{\tau\delta}^i + \delta$ for all τ. Moreover,

$$\begin{aligned}\frac{d\rho^i(t)}{dt} &= (1 - y_{(j-1)\delta}^i)(1 - 0.63y_{(j-1)\delta}^i) \leq (1 - y_{j\delta}^i + \delta)(1 - 0.63y_{j\delta}^i + \delta) \\ &\leq (1 - \rho^i(t))(1 - 0.63\rho^i(t)) + 4\delta\end{aligned}$$

Consider setting up a new function $\alpha(t)$ where $\alpha(0) = 0$ and $\frac{d\alpha}{dt} = (1-\alpha(t))(1-0.63\alpha(t))+4\delta$. Solving this differential equation gives that $\alpha(t) = \frac{100(e^{37t/100}-1)}{100e^{37t/100}-63} + 4\delta t$. We know that $\rho^i(0) = \alpha(0) = 0$. The function $\alpha(t)$ is continuous and the function $\rho(t)$ is piecewise linear. Further, for any $0 \leq t \leq 1$ whenever $\rho^i(t) = \alpha(t)$ the derivative of $\alpha(t)$ is larger than $\rho^i(t)$. This gives that $\rho^i(t) \leq \alpha(t)$ for all $0 \leq t \leq 1$.

Thus, we have that $y_t^i \leq \rho^i(t) \leq \alpha(t)$ for all $0 \leq t \leq 1$, proving the lemma. ◀

We will begin by relating the functions G and F. To do this, we will use Theorem 2. This theorem requires that we bound the probability an interval in R is in $D(R)$. We do this in the following lemma.

▶ **Lemma 5.** *For any time $0 \leq t \leq 1$ it is the case that $\mathbf{Pr}[i \in D(R) \mid i \in R] = \mathbf{Pr}[R \cap (E_i \setminus \{i\}) = \emptyset] \geq e^{-(t-y_t^i)} \geq e^{-t}$.*

Proof. Fix an interval $i = (s_i, d_i]$. If this interval is in R, then the only reason it is not in $D(R)$ is because there is another interval $j \in R$ such that j intersects the start point of i. That is if $j \in E_i \cap R$ and $j \neq i$ then in this case i will not be in $D(R)$; otherwise, if $R \cap (E_i \setminus \{i\}) = \emptyset$ then i is in $D(R)$ when $i \in R$. Thus, it suffices to bound the probability any interval is sampled to be in R which intersects s_i. The probability no interval in $E_i \setminus \{i\}$ is sampled is $\prod_{j \neq i, s_i \in (s_j, d_j]} (1 - y_t^j) \geq e^{-t+y_t^i}$. Where the inequality follows from the fact that $\sum_{j: s_i \in (s_j, d_j]} y_t^j \leq t$ for any step of the algorithm, i.e. any time t where $0 \leq t \leq 1$. ◀

Now we show two key lemmas. The first shows a relationship between G and F.

▶ **Lemma 6.** *$G(y) \geq \frac{1}{e^t} F(y)$ for all vectors y.*

Proof. We utilize Theorem 2. First notice that the procedure to construct $D(R)$ in the definition of G is a monotonic scheme. This is because the probability an interval is in $D(R)$ only decreases if intervals are added to R. Lemma 5 and Theorem 2 give the lemma. ◀

The next lemma is the key technical lemma that bounds the increase in the G at each step of the algorithm.

▶ **Lemma 7.** *It is the case that $G(y_{t+\delta}) \geq (1-\delta)G(y_t) + \frac{\delta}{e^t} \mathbb{E}\left[\sum_{i \in O} \left(f(R \cup \{i\}) - f(R)\right)\right] - O(n^2\delta^2) f(O)$ for all $t \leq \ln 2 - \delta$.*

We defer the proof of the lemma and first show how this can be used to construct our result. Using the previous two lemma, we can bound the total increase in the function by the optimal solution.

▶ **Lemma 8.** *It is the case that $G(y_{t+\delta}) \geq (1-\delta)G(y_t) + \frac{\delta}{e^t} \left(\left(1 - \alpha(t)\right) f(O) - e^t G(y_t)\right) - O(n^2\delta^2) f(O)$ for all $t \leq \ln 2 - \delta$.*

Proof. Lemma 7 says that $G(y_{t+\delta}) \geq (1-\delta)G(y_t) + \frac{\delta}{e^t} \mathbb{E}[\sum_{i \in O} f(R \cup \{i\}) - f(R)] - O(n^2\delta^2) f(O)$. By definition, $\mathbb{E}[f(R)] = F(y_t)$ and Lemma 6 states that $F(y_t) \leq e^t G(y_t)$. This gives the following. The first inequality follows from submodularity.

$$\begin{aligned}
& (1-\delta)G(y_t) + \frac{\delta}{e^t} \mathbb{E}[\sum_{i \in O} f(R \cup \{i\}) - f(R)] - O(n^2\delta^2) f(O) \\
\geq\ & (1-\delta)G(y_t) + \frac{\delta}{e^t} \mathbb{E}[f(R \cup O) - f(R)] - O(n^2\delta^2) f(O) \\
\geq\ & (1-\delta)G(y_t) + \frac{\delta}{e^t} \left(\mathbb{E}[f(R \cup O)] - e^t G(y_t)\right) - O(n^2\delta^2) f(O). \qquad (1)
\end{aligned}$$

Notice that $\mathbb{E}[f(R \cup O)] \geq (1 - \alpha(t)) f(O)$ by Theorem 3 because Lemma 4 gives that the maximum probability any interval is in R is bounded by $\alpha(t)$. Combining this with equation (1) gives the lemma. ◀

Using the two above lemmas, we can show our main result.

Proof of Theorem 1. Lemma 8 states that $G(y_{t+\delta}) \geq (1-\delta)G(y_t) + \frac{\delta}{e^t}((1-\alpha(t))f(O) - e^t G(y_t)) - O(n^2\delta^2)f(O)$ wherever $t \leq \ln 2 - \delta$. This implies that $G(y_{t+\delta}) - G(y_t) \geq -2\delta G(y_t) + \frac{\delta}{e^t}((1-\alpha(t))f(O)) - O(n^2\delta^2)f(O)$ for $t \leq \ln 2 - \delta$.

By choosing δ to be sufficiently small, $G(y_{t+\delta})$ can be bounded using a differential equation. Consider a function $g(t)$ where $g(0) = 0$ and for any $t \in [(j-1)\delta, j\delta]$ it is the case that

$$\begin{aligned} g(t) \quad &= \delta \sum_{\tau=0}^{j-2} \left(-2G(y_{\tau\delta}) + \frac{f(O)}{e^{\tau\delta}}(1-\alpha(\tau\delta)) \right) \\ &\quad + (t-(j-1)\delta)\left(-2G(y_{(j-1)\delta}) + \frac{f(O)}{e^{(j-1)\delta}}(1-\alpha((j-1)\delta)) \right). \end{aligned}$$

Inductively, notice that $G(y_t) + O(n^2\delta^2 \cdot \frac{t}{\delta})f(O) \geq g(t)$ for any t divisible by δ and t less than $\ln 2 - \delta$. Further, $\frac{dg}{dt} = -2G(y_{(j-1)\delta}) + \frac{f(O)}{e^{(j-1)\delta}}(1-\alpha((j-1)\delta)) \geq -2g(t) + \frac{f(O)}{e^t}(1-\alpha(t)) - 2\delta f(O)$. Consider a new function $h(t)$ where $h(0) = 0$ and $\frac{dh}{dt} = -2h(t) + \frac{f(O)}{e^t}(1-\alpha(t)) - 2\delta f(O)$. Solving this differential equation results in $h(.54) > .188 f(O)$[5]. Note that $.54 \leq \ln 2 - \delta$ for sufficiently small δ.

We know that $h(0) = g(0) = 0$. We also know that $h(t)$ is a continuous function and $g(t)$ is piecewise linear. Further, for any $0 \leq t \leq 1$ whenever $h(t) = g(t)$ the derivative of $g(t)$ is only larger than that of $h(t)$. Thus, we have that $h(t) \leq g(t)$ for all t. Knowing that $g(t) \leq G(y_t) + O(n^2\delta^2\frac{t}{\delta})f(O) \leq G(y_t) + O(n^2\delta)f(O)$ for $t \leq \ln n - \delta$, it is the case that $.188 f(O) < h(.54) \leq g(.54) \leq G(y_{.54}) + O(n^2\delta)f(O)$, proving the theorem for $\delta \leq \frac{1}{n^3}$. ◀

It only remains to prove Lemma 7. The proof can be found in Section 4.

4 Proof of Lemma 7

For this section, let y be the current solution computed by our algorithm at some fixed stage t. Throughout the section all lemmas and proofs will assume that $t \leq \ln 2 - \delta$, an assumption in the statement of Lemma 7. Let v be a vector equal to $y_{t+\delta} - y_t$. For simplicity, we drop the index t and throughout this section we only focus on stage t and drop the index t in y_t. We want to bound $G(y+v)$. Throughout this section, let I be the intervals in the support of v. These are the elements the algorithm chooses in the independent set and whose variables get increased. Let O be the intervals in the optimal solution.

Let S be the set of all intervals. Let R be the random set of intervals chosen according to y where every interval is sampled independently. Formally, for each interval i draw a number r_i uniformly at random from $[0,1]$ and let i be in R if $r_i < y^i$. Let $\mathcal{E}_i$ denote the event $y^i < r_i \leq y^i + \beta_i$. Intuitively, $\mathcal{E}_i$ is the event that i would not be in R if y^i is used for the sampling, but would have if y^i was increased by β_i. For $i \in I$ this is the event i was chosen in the computation of $G(y+v)$, but not $G(y)$.

For any set S', let $D(S')$ contain the intervals from S' chosen according to the algorithm that is used in G. That is $D(S')$ is constructed from S' by only adding an interval $j \in S'$ to be in $D(S')$ if there is no other interval in S' with earlier start point that also intersects j.

We would like to bound $G(y+v)$ by quantities involving O and $G(y)$. Let $\mathcal{E}(I')$ denote the event that $\mathcal{E}_i$ occurs for all $i \in I'$ and $\mathcal{E}_i$ does not occur for any $i \in I \setminus I'$ and recall that $\mathbf{Pr}[\mathcal{E}_i] = \beta_i = \delta(1-y^i)e^{y^i}$, the amount the algorithm would increase y^i if $i \in I$. It will

[5] This was verified using a differential equation solving software from Mathematica and independently verified using numerical evaluation.

be useful to first bound the probability that $R = S'$ for some S'. To do this, the following lemmas bound the probability of either an interval being in R or $R = S'$ depending on the events $\mathcal{E}_i$. The claim isn't difficult and the proof is deferred to the appendix.

▷ Claim 9. For any $i \in I$ it is the case that $\mathbf{Pr}[i \in R \mid \overline{\mathcal{E}_i}] = \frac{\mathbf{Pr}[i \in R]}{1-\beta_i} \geq \mathbf{Pr}[i \in R]$ and $\mathbf{Pr}[i \notin R \mid \overline{\mathcal{E}_i}] \geq (1-\beta_i)\mathbf{Pr}[i \notin R]$ when $t \leq \ln 2 - \delta$. Further, for any $i \in I$ and any set $S' \subseteq S$ it is the case that $\mathbf{Pr}[S' = R \mid \mathcal{E}(\{i\})] \geq \mathbf{Pr}[R = S' | \mathcal{E}_i] \prod_{j \in I, j \neq i}(1-\beta_j)$ when $t \leq \ln 2 - \delta$.

Intuitively, the next claim relates the probability R would be the same set if intervals are drawn randomly using y or $y + v$.

▷ Claim 10. Fix any set $S' \subseteq S$. It is the case that $\mathbf{Pr}[R = S' \text{ and } \mathcal{E}(\emptyset)] \geq (1 - \sum_{i \in I \setminus S'} \frac{\beta_i}{1-y^i})\mathbf{Pr}[R = S']$.

Proof. Notice that for any $i \in I$, it is the case that $\mathbf{Pr}[i \in R \text{ and } \mathcal{E}(\emptyset)] = \mathbf{Pr}[i \in R]$ and $\mathbf{Pr}[i \notin R \text{ and } \mathcal{E}(\emptyset)] = \mathbf{Pr}[i \notin R] - \beta_i = \mathbf{Pr}[i \notin R](1 - \frac{\beta_i}{1-y^i})$. The last equality follows from $\mathbf{Pr}[i \notin R] = 1 - y^i$ by definition. Knowing that elements are sampled independently, we have the following. The first equality follows since elements are sampled independently. The three terms break up the cases on if an elements is not in I, is in $I \cap S'$ or is in I and not S'.

$$\begin{aligned} & \mathbf{Pr}[R = S' \text{ and } \mathcal{E}(\emptyset)] \\ = \; & \mathbf{Pr}[R \setminus I = S' \setminus I] \prod_{i \in I \cap S'} \mathbf{Pr}[i \in R \text{ and } \mathcal{E}(\emptyset)] \prod_{i \in I \setminus S'} \mathbf{Pr}[i \notin R \text{ and } \mathcal{E}(\emptyset)] \\ = \; & \mathbf{Pr}[R = S'] \prod_{i \in I \setminus S'} (1 - \frac{\beta_i}{1-y^i}) \geq (1 - \sum_{i \in I \setminus S'} \frac{\beta_i}{1-y^i})\mathbf{Pr}[R = S']. \end{aligned}$$

The second equality follows from the observation at the beginning of the proof of the lemma. ◁

The next lemma bounds $G(y + v)$ by $G(y)$. Intuitively, the first term says that if $\mathcal{E}_i$ does not occur for any i then $G(y+v)$ is the same as $G(y)$. The second term captures the case for $\mathcal{E}_i$ occurs for exactly one $i \in O$. Finally, the probability that $\mathcal{E}_i$ occurs for more than one i is very small (proportional to δ^2) so this effect is negligible. The proof is deferred to Section 5.

▶ **Lemma 11.** *It is the case that,* $G(y+v) \geq (1 - \sum_{i \in I} \beta_i)G(y) + \sum_{i \in I} \sum_{S' \subseteq S \setminus \{i\}} \beta_i \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S' \cup \{i\})) - O(n^2\delta^2 f(O))$.

Next it is observed that the choice of the set I allows us to swap the terms in the expression in the previous lemma by the optimal solution O.

▶ **Lemma 12.** $G(y+v) \geq (1 - \sum_{i \in O} \beta_i)G(y) + \sum_{i \in O} \sum_{S' \subseteq S \setminus \{i\}} \beta_i \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S' \cup \{i\})) - O(n^2\delta^2 f(O))$

Proof. Consider the value of

$$\sum_{i \in I} \beta_i \left(\sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S' \cup \{i\})) - G(y) \right).$$

This equals

$$\sum_{i \in I} \beta_i \left(\sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S'] f(D(S' \cup \{i\})) - G(y) \right).$$

This is equal to the following since elements are sampled independently

$$\sum_{i\in I}\beta_i\left(\sum_{S'\subseteq S}\mathbf{Pr}[R=S']f(D(S'\cup\{i\}))-G(y)\right)=\sum_{i\in I}w_i.$$

By definition, this is only greater than $\sum_{i\in O}w_i$. Reversing the above steps for O and combining with Lemma 11 gives the lemma. ◀

Our remaining goal is to bound part of the expression from the prior lemma,

$$\sum_{S'\subseteq S\setminus\{i\}}\sum_{i\in O}\beta_i\mathbf{Pr}[R=S'\mid\mathcal{E}_i]f(D(S'\cup\{i\}))+\sum_{i\in O}\beta_iG(y).$$

Recall that E_i is the set of intervals starting earlier than i that intersect i and also the interval i itself. The intervals in $E_i\setminus\{i\}$ are the intervals, which if they are sampled to be in R then i will not be in $D(R)$. Let B_i be the set containing intervals that start during interval i and also i. The following fact will be useful for applying submodularity.

▶ **Lemma 13.** *For any set $S'\subseteq S$ consider $\{S'\setminus B_i\}_{i\in O}$, a collection of subsets of S'. It is the case that every interval in S' appears in exactly $|O|-1$ sets in this collection. Further, each interval in S appears in exactly one set B_i.*

Proof. To show the lemma, it suffices to show that every interval in S appears in exactly one set B_i for some $i\in O$. Indeed, we may assume that the intervals in O span the entire time horizon (adding dummy intervals as needed). Then, an interval $j\in S$ can only be in B_i if j starts during i. Knowing that O cannot have two intervals that overlap, we have the lemma. ◀

The next lemma is a technical lemma. The purpose is to take an expression $f(D(S')\setminus B_i)$ depending on a set S' and B_i for $i\in O$ and bound it by an expression depending on $f(D(S'))$ without B_i inside the function input. The lemma follows from submodularity and the previous lemma.

▶ **Lemma 14.** *Fix any set $S'\subseteq S$. It is the case that $\delta f(D(S'))\geq\sum_{i\in O}\beta_i\left(f(D(S'))-f(D(S')\setminus B_i)\right)$.*

Proof. Consider the term $\sum_{i\in O}\beta_i(f(D(S'))-f(D(S')\setminus B_i))$. We will remove all negative terms as they only makes the expression smaller. Let O' be all i where $f(D(S'))-f(D(S')\setminus B_i)>0$. The lemma follows if we prove that $f(D(S'))\geq\sum_{i\in O'}(f(D(S'))-f(D(S')\setminus B_i))$ because this implies $\delta f(D(S'))\geq\delta\sum_{i\in O'}(f(D(S'))-f(D(S')\setminus B_i))\geq\sum_{i\in O'}\beta_i(f(D(S'))-f(D(S')\setminus B_i))$ knowing that $\beta_i\leq\delta$ and all terms are positive.

Now it is established that $f(D(S'))\geq\sum_{i\in O'}(f(D(S'))-f(D(S')\setminus B_i))$, which follows by submodularity. Indeed, let $A_0=D(S')\setminus\cup_{i\in O'}B_i$. Arbitrarily order the sets $B_1,B_2,\ldots B_{|O'|}$ and let $A_i=A_{i-1}\cup(B_i\cap D(S'))$ for $1\leq i\leq|O'|$. By submodularity, $\sum_{i\in O'}(f(D(S'))-f(D(S')\setminus B_i))\leq\sum_{i\in O'}(f(A_i)-f(A_{i-1}))=f(D(S'))-f(A_0)\leq f(D(S'))$. The equality follows from the function being positive and the inequality from submodularity. ◀

Assuming $\mathcal{E}_i$ occurs, the purpose of the following lemma is to separate the cases where at least one interval in E_i is in R and the other where no interval in E_i is in R. Intuitively, if no interval in E_i is in R then i will be in $D(R)$ otherwise i will not. In either case, when $\mathcal{E}_i$ occurs the interval i ensures no interval in B_i is in $D(R)$ and the lemma bounds the cost of removing B_i by applying Lemma 14. The proof is deferred to Section 6.

▶ **Lemma 15.** *It is the case that,*

$$G(y+v) \geq (1-\delta)G(y) + \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] \sum_{i \in O, S' \cap E_i = \emptyset} \beta_i (f(D(S') \setminus B_i \cup \{i\}) - f(D(S') \setminus B_i)) - O((n\delta)^2 f(O)).$$

Our goal now is to bound the second term in the previous lemma by showing this following. This shows that the second term is at least $\frac{\delta}{e^t}$ multiplied by the expected value of adding each element of O to R individually.

▶ **Lemma 16.** $\sum_{S' \subseteq S} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] \sum_{i \in O, S' \cap E_i = \emptyset} \beta_i (f(D(S') \setminus B_i \cup \{i\}) - f(D(S') \setminus B_i)) \geq \frac{\delta}{e^t} \sum_{S' \subseteq S} \mathbf{Pr}[R = S'] \sum_{i \in O} f_{S'}(i)$

Before we prove the lemma, we show how this can be used to complete the proof of Lemma 7.

Proof of Lemma 7. By combining lemmas 15 and 16 we have the following.

$$\begin{aligned} G(y+v) &\geq (1-\delta)G(y) + \frac{\delta}{e^t} \sum_{S' \subseteq S} \mathbf{Pr}[R = S'] \sum_{i \in O} f_{S'}(i) - O((n\delta)^2 f(O)) \\ &\geq (1-\delta)G(y) + \frac{\delta}{e^t} \mathbb{E}[\sum_{i \in O} f_R(i)] - O((n\delta)^2 f(O)) \end{aligned}$$

This completes the proof. ◀

It only remains to prove Lemma 16.

Proof of Lemma 16. Consider the term $\sum_{S' \subseteq S} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] \sum_{i \in O, S' \cap E_i = \emptyset} \beta_i (f(D(S') \setminus B_i \cup \{i\}) - f(D(S') \setminus B_i))$. Rearranging the summations and using the definition of $f_{S'}(i)$ this is equal to $\sum_{i \in O} \sum_{S' \subseteq S, S' \cap E_i = \emptyset} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] \beta_i f_{D(S') \setminus B_i}(i)$. We know that for any set $S' \subseteq S$ if $S' \cap E_i = \emptyset$ then $S' \subseteq (S \setminus E_i)$. Using this, the term is equal to the following.

$$\begin{aligned} &\sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} \beta_i \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f_{D(S') \setminus B_i}(i) \\ = &\sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} \beta_i \mathbf{Pr}[S' = R \setminus E_i \text{ and } R \cap E_i = \emptyset \mid \mathcal{E}_i] f_{D(S') \setminus B_i}(i) \end{aligned}$$

To see why the previous equality holds, notice that $R = S'$ if and only if $S' = R \setminus E_i$ and $R \cap E_i = \emptyset$ for $S' \subseteq (S \setminus E_i)$. Now we continue to lower bound this expression.

$$\begin{aligned} &\sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} \beta_i \mathbf{Pr}[S' = R \setminus E_i \mid \mathcal{E}_i] \mathbf{Pr}[R \cap E_i = \emptyset \mid \mathcal{E}_i] f_{D(S') \setminus B_i}(i) \\ &\quad \text{[Definition of } R \text{ implies independence]} \\ = &\sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} \beta_i \mathbf{Pr}[S' = R \setminus E_i] \mathbf{Pr}[R \cap (E_i \setminus \{i\}) = \emptyset] f_{D(S') \setminus B_i}(i) \\ &\quad \text{[Definition of } \mathcal{E}_i] \\ \geq &\sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} \frac{\beta_i}{e^{t-y^i}} \mathbf{Pr}[S' = R \setminus E_i] f_{D(S') \setminus B_i}(i) \quad \text{[Lemma 5]} \\ = &\frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} (1-y^i) \mathbf{Pr}[R = S' \setminus E_i] f_{D(S') \setminus B_i}(i) \quad \text{[Definition of } \beta_i] \end{aligned} \tag{2}$$

Notice that $1 = \sum_{E \subseteq E_i} \mathbf{Pr}[R \cap E_i = E \mid i \notin R]$ because the right hand side captures all the events in a probability distribution. Further, fix an element $i \in O$ and notice that for any set $S' \subseteq (S \setminus E_i)$ and any set $E \subseteq E_i$ it is the case that $\mathbf{Pr}[S' = R \setminus E_i] \cdot \mathbf{Pr}[R \cap E_i = E \mid i \notin R] = \mathbf{Pr}[R = S' \cup E \mid i \notin R]$. This follows for two reasons. One is because elements are sampled independently. The other is because $\mathbf{Pr}[S' = R \setminus E_i] = \mathbf{Pr}[S' = R \setminus E_i \mid i \notin R]$ since $i \in E_i$ and the independence of sampling elements. Using these facts, the following holds.

$$
\begin{aligned}
(2) \quad &= \frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} (1 - y^i) \mathbf{Pr}[S' = R \setminus E_i] f_{D(S') \setminus B_i}(i) \sum_{E \subseteq E_i} \mathbf{Pr}[R \cap E_i = E \mid i \notin R] \\
&= \frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} (1 - y^i) f_{D(S') \setminus B_i}(i) \sum_{E \subseteq E_i} \mathbf{Pr}[R = S' \cup E \mid i \notin R] \\
&\quad \text{[Independence]} \\
&\geq \frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} (1 - y^i) f_{S' \setminus B_i}(i) \sum_{E \subseteq E_i} \mathbf{Pr}[R = S' \cup E \mid i \notin R] \\
&\quad \text{[Submodularity]} \\
&\geq \frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq (S \setminus E_i)} (1 - y^i) \sum_{E \subseteq E_i} f_{(S' \cup E) \setminus \{i\}}(i) \mathbf{Pr}[R = S' \cup E \mid i \notin R] \\
&\quad \text{[Submodularity and } i \in B_i\text{]} \\
&= \frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq (S \setminus \{i\})} (1 - y^i) f_{S' \setminus \{i\}}(i) \mathbf{Pr}[R = S' \mid i \notin R] \\
&= \frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq (S \setminus \{i\})} f_{S' \setminus \{i\}}(i) \mathbf{Pr}[R = S'] \\
&\quad \text{[}(1 - y^i) = \mathbf{Pr}[i \notin R] \text{ and definition of conditional probability]} \\
&= \frac{\delta}{e^t} \sum_{i \in O} \sum_{S' \subseteq S} f_{S'}(i) \mathbf{Pr}[R = S'] \quad [f_{S'}(i) = 0 \text{ if } i \in S']
\end{aligned}
$$

◀

5 Proof of Lemma 11

This section is devoted to proving Lemma 11.

Consider $G(y + v)$. The value of $G(y + v)$ is equal to $\sum_{S' \subseteq S} \sum_{I' \subseteq I} \mathbf{Pr}[R = S' \text{ and } \mathcal{E}(I')] f(D(S' \cup I'))$. This is equal to the following by breaking this into cases. This is a partitioning of the event space by definition of $\mathcal{E}(I')$.

$$
\begin{aligned}
&\sum_{S' \subseteq S} \mathbf{Pr}[R = S' \text{ and } \mathcal{E}(\emptyset)] f(D(S')) \\
&+ \sum_{i \in I} \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \text{ and } \mathcal{E}(\{i\})] f(D(S' \cup \{i\})) \\
&+ \sum_{I' \subseteq I, |I'| \geq 2} \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \text{ and } \mathcal{E}(I')] f(D(S' \cup \{i\}))
\end{aligned}
$$

Knowing that f is positive, this is greater than the following.

$$
\sum_{S' \subseteq S} \mathbf{Pr}[R = S' \text{ and } \mathcal{E}(\emptyset)] f(D(S')) \tag{3}
$$

$$
+ \sum_{i \in I} \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \text{ and } \mathcal{E}(\{i\})] f(D(S' \cup \{i\})) \tag{4}
$$

The proof bounds these two terms separately. First consider (3). Using Claim 10 this is greater than $\sum_{S' \subseteq S}(1 - \sum_{i \in I \setminus S'} \frac{\beta_i}{1-y^i})\mathbf{Pr}[R = S']f(D(S')) = G(y) - \sum_{i \in I} \frac{\beta_i}{1-y^i} \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S']f(D(S'))$. The definition of β_i gives that this is equal to $G(y) - \sum_{i \in I} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S']f(D(S')) - \sum_{i \in I} y^i e^{-y^i} \delta \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S']f(D(S'))$. We will establish that this is only greater than $(1 - \sum_{i \in I} \beta_i)G(y)$. Consider the last term.

$$\begin{aligned}
& \sum_{i \in I} y^i e^{-y^i} \delta \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S']f(D(S')) \\
= & \sum_{i \in I} y^i e^{-y^i} \delta \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S']\mathbf{Pr}[i \notin R]f(D(S')) \\
= & \sum_{i \in I} y^i e^{-y^i} (1 - y^i) \delta \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S']f(D(S')) \\
= & \sum_{i \in I} y^i \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S']f(D(S'))
\end{aligned}$$

By definition of the algorithm $\frac{w_i}{\beta_i} = \mathbb{E}[f(D(R \cup \{i\})) - f(D(R))] = \sum_{S' \subseteq S} \mathbf{Pr}[R = S'](f(D(S' \cup \{i\})) - f(D(S'))) = \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S'](f(D(S' \cup \{i\})) - f(D(S'))) > 0$ for all $i \in I$. The last equality follows since a term is 0 if i is in S'. Since elements are sampled independently, this gives that the previous term is only less than the following.

$$\begin{aligned}
& \sum_{i \in I} y^i \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S']f(D(S')) \\
\leq & \sum_{i \in I} y^i \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S']f(D(S' \cup \{i\})) \\
= & \sum_{i \in I} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S' \cup \{i\}]f(D(S' \cup \{i\})) \quad [\text{Note that } \mathbf{Pr}[i \in R] = y_i]
\end{aligned}$$

Now we use this to bound (3). (3) is greater than or equal to the following.

$$\begin{aligned}
& G(y) - \sum_{i \in I} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S']f(D(S')) - \sum_{i \in I} y^i e^{-y^i} \delta \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S']f(D(S')) \\
\geq & G(y) - \sum_{i \in I} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S']f(D(S')) - \sum_{i \in I} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S' \cup \{i\}]f(D(S' \cup \{i\})) \\
= & (1 - \sum_{i \in I} \beta_i)G(y)
\end{aligned}$$

It remains to bound (4). Using conditional probability, this can be bounded as follows.

$$\begin{aligned}
& \sum_{i \in I} \mathbf{Pr}[\mathcal{E}(\{i\})] \sum_{S' \subseteq S} \mathbf{Pr}[R = S' | \mathcal{E}(\{i\})]f(D(S' \cup \{i\})) \\
= & \sum_{i \in I} \mathbf{Pr}[\mathcal{E}_i] \prod_{j \in I, j \neq i} \mathbf{Pr}(\overline{\mathcal{E}}_j) \sum_{S' \subseteq S} \mathbf{Pr}[R = S' | \mathcal{E}(\{i\})]f(D(S' \cup \{i\})) \\
= & \sum_{i \in I} \beta_i \prod_{j \in I, j \neq i} (1 - \beta_j) \sum_{S' \subseteq S} \mathbf{Pr}[R = S' | \mathcal{E}(\{i\})]f(D(S' \cup \{i\})) \\
\geq & \prod_{i \in I} (1 - \beta_i) \sum_{i \in I} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S' | \mathcal{E}(\{i\})]f(D(S' \cup \{i\})) \\
\geq & (1 - \sum_{i \in I} \beta_i) \sum_{i \in I} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S' | \mathcal{E}(\{i\})]f(D(S' \cup \{i\}))
\end{aligned}$$

Knowing that $\beta_i \leq \delta$, this is greater than $\sum_{i\in I}\beta_i \sum_{S'\subseteq S}\mathbf{Pr}[R = S'|\mathcal{E}(\{i\})]f(D(S'\cup\{i\})) - O(\delta^2 n^2 f(O))$.

Now we know that,

$$\begin{aligned}
&\sum_{i\in I}\beta_i \sum_{S'\subseteq S}\mathbf{Pr}[R = S'|\mathcal{E}(\{i\})]f(D(S'\cup\{i\}))\\
=\ &\sum_{i\in I}\beta_i \sum_{S'\subseteq S}\mathbf{Pr}[R = S'|\mathcal{E}_i \text{ and } \overline{\mathcal{E}}_j \text{ for } j\neq i)]f(D(S'\cup\{i\})) \quad \text{[Def. of } \mathcal{E}(\{i\})]\\
\geq\ &\sum_{i\in I}\beta_i \prod_{j\in I, j\neq i}(1-\beta_j)\sum_{S'\subseteq S}\mathbf{Pr}[R = S'|\mathcal{E}_i)]f(D(S'\cup\{i\})) \quad \text{[Claim 9 and independence]}\\
\geq\ &\sum_{i\in I}\beta_i \sum_{S'\subseteq S}\mathbf{Pr}[R = S'|\mathcal{E}_i)]f(D(S'\cup\{i\})) - |I|\delta^2 f(O)\\
\geq\ &\sum_{i\in I}\beta_i \sum_{S'\subseteq S\setminus\{i\}}\mathbf{Pr}[R = S'|\mathcal{E}_i)]f(D(S'\cup\{i\})) - |I|\delta^2 f(O)
\end{aligned}$$

The last line follows since if $\mathcal{E}_i$ occurs then R does not contain i. Putting this all together gives the lemma.

6 Proof of Lemma 15

This section is devoted to proving Lemma 15.

Consider the following expression. Lemma 12 gives the following.

$$G(y+v) \geq \sum_{S'\subseteq S}\sum_{i\in O}\beta_i \mathbf{Pr}[R = S' \mid \mathcal{E}_i]f(D(S'\cup\{i\})) \tag{5}$$

$$+(1-\sum_{i\in O}\beta_i)G(y). \tag{6}$$

We see that (5) equals the following.

$$\sum_{i\in O}\beta_i\Big(\sum_{S'\subseteq S, S'\cap E_i=\emptyset}\mathbf{Pr}[R = S' \mid \mathcal{E}_i]f(D(S'\cup\{i\})) + \sum_{S'\subseteq S, S'\cap E_i\neq\emptyset}\mathbf{Pr}[R = S' \mid \mathcal{E}_i]f(D(S'\cup\{i\}))\Big)$$

By definition of G, for any set $S'\subseteq S$ it is the case that $D(S'\cup\{i\})$ includes i only if S' includes no interval in $E_i\setminus\{i\}$. We also know that for any $j\in S'$ it is the case that $j\in D(S'\cup\{i\})$ if and only if $j\in D(S')$ and $j\notin B_i$. Using these two facts, the previous term is equal to the following.

$$\sum_{i\in O}\beta_i\Big(\sum_{S'\subseteq S, S'\cap E_i=\emptyset}\mathbf{Pr}[R = S' \mid \mathcal{E}_i]f(D(S')\setminus B_i\cup\{i\})) + \sum_{S'\subseteq S, S'\cap E_i\neq\emptyset}\mathbf{Pr}[R = S' \mid \mathcal{E}_i]f(D(S')\setminus B_i)\Big)$$

This is equal to the following.

$$\sum_{i\in O}\beta_i\Bigg(\sum_{S'\subseteq S, S'\cap E_i=\emptyset}\mathbf{Pr}[R = S' \mid \mathcal{E}_i]\Big(f(D(S')\setminus B_i\cup\{i\})) - f(D(S')\setminus B_i)\Big) + \sum_{S'\subseteq S}\mathbf{Pr}[R = S' \mid \mathcal{E}_i]f(D(S')\setminus B_i)\Bigg)$$

We focus on bounding $\sum_{i \in O} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S') \setminus B_i)$ along with (6). The rest of the expression is carried to the end of the proof. First we establish a bound on $\sum_{i \in O} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S') \setminus B_i)$ in the following claim and then it is combined with (6). The purpose of the following claim is to remove the conditioning on $\mathcal{E}_i$.

▷ Claim 17. $\sum_{i \in O} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S') \setminus B_i) = \sum_{i \in O} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S'] f(D(S') \setminus B_i)$.

Proof. First note that $\mathbf{Pr}[R = S' \mid \mathcal{E}_i] > 0$ if and only if $i \notin S'$. Thus we have that the left hand side is equal to $\sum_{i \in O} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S') \setminus B_i)$. Using the definition of conditional probability and the definition of $\mathcal{E}_i$ this is equal to $\sum_{i \in O} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \frac{\mathbf{Pr}[R \setminus \{i\} = S' \text{ and } \mathcal{E}_i]}{\mathbf{Pr}[\mathcal{E}_i]} f(D(S') \setminus B_i)$. By independence, this equals $\sum_{i \in O} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S'] f(D(S') \setminus B_i) = \sum_{i \in O} \beta_i \sum_{S' \subseteq S \setminus \{i\}} \mathbf{Pr}[R \setminus \{i\} = S'] f(D(S') \setminus B_i)(\mathbf{Pr}[i \in R] + \mathbf{Pr}[i \notin R]) = \sum_{i \in O} \beta_i \sum_{S' \subseteq S \setminus \{i\}} (\mathbf{Pr}[R = S'] + \mathbf{Pr}(R = S' \cup \{i\})) f(D(S') \setminus B_i)$.

We know that for any $S' \subseteq S \setminus \{i\}$ it is the case that $f(D(S') \setminus B_i) = f(D(S' \cup \{i\}) \setminus B_i)$ because B_i is the set of intervals that are not in the contention resolution scheme if i is input and also i is in B_i. Using this, we have that the previous expression is equal to $\sum_{i \in O} \beta_i \sum_{S' \subseteq S \setminus \{i\}} (\mathbf{Pr}[R = S'] f(D(S') \setminus B_i) + \mathbf{Pr}(R = S \cup \{i\}) f(D(S' \cup \{i\}) \setminus B_i)) = \sum_{i \in O} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S'] f(D(S') \setminus B_i)$. ◁

Going back to $\sum_{i \in O} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] f(D(S') \setminus B_i)$ with the expansion of (6) using the definition of G and the previous claim, we have the following.

$$\sum_{i \in O} \beta_i \sum_{S' \subseteq S} \mathbf{Pr}[R = S'] f(D(S') \setminus B_i) + (1 - \sum_{i \in O} \beta_i) \sum_{S' \subseteq S} \mathbf{Pr}[R = S'] f(D(S'))$$

We apply Lemma 14 for each term S' to get that this is greater than the following.

$$(1 - \delta) \sum_{S' \subseteq S} \mathbf{Pr}[R = S'] f(D(S')) = (1 - \delta) G(y)$$

The above gives that $G(y + v) \geq (1 - \delta) G(y) + \sum_{i \in O} \beta_i \sum_{S' \subseteq S, S' \cap E_i = \emptyset} \mathbf{Pr}[R = S' \mid \mathcal{E}_i] \Big(f(D(S') \setminus B_i \cup \{i\})) - f(D(S') \setminus B_i) \Big)$, giving the lemma.

7 Conclusion

This paper introduces the approach of using contention resolution extensions to optimize a submodular function subject to a set of constraints. This algorithmic approach can be used to improve the best known result when the constraints correspond to independent sets in an interval graph. The next direction is to determine if this approach can be used to improve on the best known approximation for other submodular optimization problems.

References

1. Niv Buchbinder and Moran Feldman. Constrained Submodular Maximization via a Non-symmetric Technique. *CoRR*, abs/1611.03253, 2016. `arXiv:1611.03253`.
2. Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a Submodular Set Function Subject to a Matroid Constraint (Extended Abstract). In *IPCO*, pages 182–196, 2007.
3. Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a Monotone Submodular Function Subject to a Matroid Constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.
4. Chandra Chekuri, T. S. Jayram, and Jan Vondrák. On Multiplicative Weight Updates for Concave and Submodular Function Maximization. In *ITCS*, pages 201–210, 2015.
5. Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent Randomized Rounding via Exchange Properties of Combinatorial Structures. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 575–584, 2010.
6. Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular Function Maximization via the Multilinear Relaxation and Contention Resolution Schemes. *SIAM J. Comput.*, 43(6):1831–1879, 2014.
7. Alina Ene and Huy L Nguyen. Constrained Submodular Maximization: Beyond 1/e. In *FOCS*, 2016.
8. Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing Non-monotone Submodular Functions. *SIAM J. Comput.*, 40(4):1133–1153, 2011.
9. Moran Feldman. *Maximization Problems with Submodular Objective Functions*. PhD thesis, Technion - Israel Institute of Technology, 2013.
10. Moran Feldman, Christopher Harshaw, and Amin Karbasi. Greed Is Good: Near-Optimal Submodular Maximization via Greedy Optimization. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*, pages 758–784, 2017.
11. Moran Feldman, Joseph Naor, and Roy Schwartz. A Unified Continuous Greedy Algorithm for Submodular Maximization. In *FOCS*, pages 570–579, 2011.
12. Ryan Gomes and Andreas Krause. Budgeted Nonparametric Learning from Data Streams. In *ICML*, pages 391–398, 2010.
13. David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the Spread of Influence through a Social Network. *Theory of Computing*, 11:105–147, 2015.
14. Jon Kleinberg and Eva Tardos. *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.
15. Matt J. Kusner, Wenlin Chen, Quan Zhou, Zhixiang Eddie Xu, Kilian Q. Weinberger, and Yixin Chen. Feature-Cost Sensitive Learning with Submodular Trees of Classifiers. In *AAAI*, pages 1939–1945, 2014.
16. Jon Lee, Vahab S. Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Maximizing Nonmonotone Submodular Functions under Matroid or Knapsack Constraints. *SIAM J. Discrete Math.*, 23(4):2053–2078, 2010.
17. Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular Maximization over Multiple Matroids via Generalized Exchange Properties. *Math. Oper. Res.*, 35(4):795–806, 2010.
18. Hui Lin and Jeff A. Bilmes. Multi-document Summarization via Budgeted Maximization of Submodular Functions. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*, pages 912–920, 2010.
19. Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pages 67–74, 2008.
20. Jan Vondrák. Symmetry and Approximability of Submodular Maximization Problems. *SIAM J. Comput.*, 42(1):265–304, 2013.

A Omitted Proofs

Proof of Claim 9. By definition $\mathbf{Pr}[i \in R \mid \overline{\mathcal{E}_i}] = \frac{\mathbf{Pr}[i \in R]}{1-\beta_i}$. To see the other part of the claim, by definition of $\mathcal{E}_i$ it is the case that $\mathbf{Pr}[i \notin R \mid \overline{\mathcal{E}_i}] = \frac{(1-y^i-\beta_i)}{1-\beta_i}$ and $(1-\beta_i)\mathbf{Pr}[i \notin R] = (1-\beta_i)(1-y^i)$. For all $\beta_i \in [0,1)$ it is the case that $\frac{(1-y^i-\beta_i)}{1-\beta_i} \geq (1-\beta_i)(1-y^i)$ if $0 \leq y^i \leq \frac{1-\beta_i}{2}$. Finally, $0 \leq y^i \leq \frac{1-\delta}{2} \leq \frac{1-\beta_i}{2}$ when $t \leq \ln 2 - \delta$. This is because $t \leq \ln 2 - \delta$ by assumption and Lemma 4 states that any entry in y is at most $1-e^{-t} \leq 1-e^{-(\ln 2-\delta)} \leq \frac{1-\delta}{2}$.

Now consider the second part of the lemma. Recall that $\mathcal{E}(\{i\})$ is the event where $\mathcal{E}_i$ occurs as well as $\overline{\mathcal{E}}_j$ for all $j \in I$ where $j \neq i$. We have the following.

$$\mathbf{Pr}[S' = R \mid \mathcal{E}(\{i\})] = \frac{\mathbf{Pr}[S' = R \text{ and } \mathcal{E}(\{i\})]}{\mathbf{Pr}[\mathcal{E}(\{i\})]}$$

By independence this equals the following.

$$\frac{\mathbf{Pr}[S' \setminus I = R \setminus I]}{\mathbf{Pr}[\mathcal{E}(\{i\})]}\mathbf{Pr}[\{i\} \cap S' = \{i\} \cap R \text{ and } \mathcal{E}_i] \prod_{j \in I, j \in S', j \neq i} \mathbf{Pr}[j \in R \text{ and } \overline{\mathcal{E}}_j]$$
$$\cdot \prod_{j \in I, j \notin S', j \neq i} \mathbf{Pr}[j \notin R \text{ and } \overline{\mathcal{E}}_j]$$

By independence we know that $\mathbf{Pr}[\mathcal{E}(\{i\})] = \mathbf{Pr}[\mathcal{E}_i]\prod_{j \in I, j \neq i}\mathbf{Pr}[\overline{\mathcal{E}}_j]$. Using this and conditional probability, the prior term is equal to the following.

$$\frac{\mathbf{Pr}[S' \setminus I = R \setminus I]}{\mathbf{Pr}[\mathcal{E}_i]}\mathbf{Pr}[\{i\} \cap S' = \{i\} \cap R \text{ and } \mathcal{E}_i] \prod_{j \in I, j \in S', j \neq i} \mathbf{Pr}[j \in R \mid \overline{\mathcal{E}}_j]$$
$$\cdot \prod_{j \in I, j \notin S', j \neq i} \mathbf{Pr}[j \notin R \mid \overline{\mathcal{E}}_j]$$

The first argument shown in the lemma gives that this is at least the following. This argument allows us to remove the conditioning on $\overline{\mathcal{E}_j}$.

$$\prod_{j \neq i, j \in I}(1-\beta_j)\frac{\mathbf{Pr}[S' \setminus I = R \setminus I]}{\mathbf{Pr}[\mathcal{E}_i]}\mathbf{Pr}[\{i\} \cap S' = \{i\} \cap R \text{ and } \mathcal{E}_i] \prod_{j \in I, j \in S', j \neq i} \mathbf{Pr}[j \in R]$$
$$\cdot \prod_{j \in I, j \notin S', j \neq i} \mathbf{Pr}[j \notin R]$$

Using independence, this is equal to the following.

$$\prod_{j \neq i, j \in I}(1-\beta_j)\frac{\mathbf{Pr}[S' = R \text{ and } \mathcal{E}_i]}{\mathbf{Pr}[\mathcal{E}_i]}$$

Finally, conditional probability gives the following.

$$\prod_{j \neq i, j \in I}(1-\beta_j)\mathbf{Pr}[S' = R \mid \mathcal{E}_i]$$

◁

Prepare for the Expected Worst: Algorithms for Reconfigurable Resources Under Uncertainty

David Ellis Hershkowitz
Carnegie Mellon University, Pittsburgh, PA, USA
dhershko@cs.cmu.edu

R. Ravi
Carnegie Mellon University, Pittsburgh, PA, USA
ravi@andrew.cmu.edu

Sahil Singla
Princeton University, Princeton, NJ, USA
Institute for Advanced Study, Princeton, NJ, USA
singla@cs.princeton.edu

Abstract

In this paper we study how to optimally balance cheap inflexible resources with more expensive, reconfigurable resources despite uncertainty in the input problem. Specifically, we introduce the MinEMax model to study "build versus rent" problems. In our model different scenarios appear independently. Before knowing which scenarios appear, we may build rigid resources that cannot be changed for different scenarios. Once we know which scenarios appear, we are allowed to rent reconfigurable but expensive resources to use across scenarios. Although computing the objective in our model might seem to require enumerating exponentially-many possibilities, we show it is well estimated by a surrogate objective which is representable by a polynomial-size LP. In this surrogate objective we pay for each scenario only to the extent that it exceeds a certain threshold. Using this objective we design algorithms that approximately-optimally balance inflexible and reconfigurable resources for several NP-hard covering problems. For example, we study variants of minimum spanning and Steiner trees, minimum cuts, and facility location. Up to constants, our approximation guarantees match those of previously-studied algorithms for demand-robust and stochastic two-stage models. Lastly, we demonstrate that our problem is sufficiently general to smoothly interpolate between previous demand-robust and stochastic two-stage problems.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Packing and covering problems; Theory of computation → Routing and network design problems; Theory of computation → Facility location and clustering; Theory of computation → Rounding techniques

Keywords and phrases Approximation Algorithms, Optimization Under Uncertainty, Two-Stage Optimization, Expected Max

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.4

Category APPROX

Related Version A full version of the paper is available at `https://arxiv.org/abs/1811.11635`.

Funding *David Ellis Hershkowitz*: Supported in part by NSF grants CCF-1527110, CCF-1618280, CCF-1814603, CCF-1910588, NSF CAREER award CCF-1750808 and a Sloan Research Fellowship.
R. Ravi: Supported in part by the U. S. Office of Naval Research award N00014-18-1-2099, and the U. S. NSF award CCF-1527032.
Sahil Singla: Supported in part by Schmidt Foundation and NSF awards CCF-1319811, CCF-1536002, and CCF-1617790.



© David Ellis Hershkowitz, R. Ravi, and Sahil Singla;
licensed under Creative Commons License CC-BY

1 Introduction

Optimizing for reconfigurable resources under uncertainty formalizes the challenges of balancing expensive, flexible resources with cheap, inflexible ones. For example, such optimization problems formalize the challenges in "build versus rent" problems. Concretely, consider the algorithmic challenges faced by an Internet service provider (ISP). An ISP must provide content to its customers while balancing between rigid and reconfigurable resources. In particular, it can build out its own network – a rigid resource – or choose to support traffic on a competitor's network – a flexible resource – at a marked up premium. This latter resource is reconfigurable since an ISP can change which edges in a competitor's network it uses at any given time. To minimize the additional load on its network, the competitor charges the ISP for the maximum extra bandwidth it must support at any given moment. Furthermore, an ISP only has probabilistic knowledge of where customer demands will occur: Based on where previous demands have occurred an ISP estimates future demands, but it does not exactly know the future demands. If a demand occurs which the ISP's network cannot service, it must use the competitor's network to support it. Thus, an ISP balances rigid and flexible resources in the face of uncertainty, and pays for the cost of its own network plus the cost of supporting the *expected maximum* traffic routed on its competitor's network.

In this paper, we introduce the MinEMax model to study the algorithmic challenges associated with optimizing reconfigurable resources under uncertainty. In our model we are given a set of *scenarios* that might occur. In the preceding example these scenarios were the sets of possible demands. We think of problems in our model as being divided between a first stage where we "build" rigid resources and a second stage where we "rent" flexible resources. In particular, in the first stage we can build non-reconfigurable resources without knowing which scenarios occur. In the second stage, each scenario independently realizes according to its specified Bernoulli probability, and we can rent reconfigurable resources at an increased cost to use among any of our scenarios. For instance, in the preceding example the ISP first built its own network and then, once it learned where demands occurred, it could rent bandwidth to support different demands over time. In fact, this example is exactly our MinEMax Steiner tree problem. Thus, the objective we minimize is the first stage cost plus the *expected maximum* cost of additional reconfigurable resources required for any realized scenario; hence the name of our model.

Since every scenario is an independent Bernoulli, there are exponentially-many ways in which scenarios realize. It is not even clear how to efficiently compute the expected second-stage cost. Nonetheless, we provide techniques to simplify and reason about the MinEMax cost, and therefore solve various MinEMax problems.

The primary contributions of our work are as follows.

1. We introduce the MinEMax model for optimization of reconfigurable resources under uncertainty.
2. We show that, although evaluating the MinEMax objective function may seem difficult, a MinEMax problem can be approximately reduced to a "TruncatedTwoStage" problem whose objective is representable by an LP.
3. Armed with 2, we adapt various rounding techniques to give approximation algorithms for a variety of two-stage MinEMax problems including spanning and Steiner trees, cuts, and facility location problems.
4. Lastly, we show that the MinEMax model captures the commonly studied two-stage models for optimization under uncertainty: the *stochastic* and *demand-robust* models. We even show that it generalizes a "Hybrid" problem that interpolates between these models.

1.1 Related Work

Significant work has been done in two-stage optimization under uncertainty. The two most commonly studied models are the stochastic model [22, 15, 24] and the demand-robust model [9, 4, 14, 13]. In the **stochastic two-stage model** a probability distribution is given over scenarios and our objective is the *expected* total cost. In the **demand-robust** two-stage model we are given scenarios and our objective is the cost of the *worst-case* scenario given our first stage solution.

Another related model is *Distributionally robust optimization* (DRO) [23, 12, 8, 5]. In DRO we are given a distribution along with a ball of "nearby" distributions, and we must pay the *worst-case expectation* over all these distributions. Similarly to our model, DRO generalizes both the stochastic and demand-robust two-stage models. Our model can be seen as a "flip" of the DRO model: while the DRO model takes the worst-case over distributions our model takes a distribution over worst cases. Like DRO, our model is also sufficiently general to capture stochastic and demand-robust optimization. A recent result [20] – which shows that approximation algorithms are possible in DRO – complements our approximation algorithms in MinEMax.

A well studied measure for risk-aversion from stochastic programming is *conditional value at risk (CVaR)* [1]. Roughly, CVaR gives the average cost in the worst-case case α tail of a distribution. A notable recent work in CVaR presents a data-driven approach to two-stage risk aversion [18]. Theorem 1 in their work is reminiscent of our reduction of MinEMax to Hybrid; this theorem shows that their objective can be reformulated as a combination of the CVaR cost and the worst-case distribution. We emphasize that while CVaR might appear similar to the TruncatedTwoStage metric studied in this work, these two metrics are distinct and not readily comparable. Two salient differences are: (1) the threshold in the TruncatedTwoStage objective is the minimizing threshold while in CVaR the threshold is fixed, and (2) the TruncatedTwoStage objective sums up the truncated cost over a set of Bernoulli random variables whereas CVaR takes a truncated average cost with respect to a single distribution. Moreover, to the best of our knowledge, CVaR has not been studied in the context of approximation algorithms.

Several additional models for optimization under uncertainty – some of which even interpolate between stochastic and demand-robust – have also been studied. A series of papers [26, 25, 27] examined various models of two-stage optimization that capture risk-aversion. Notably, the model of [27] interpolates between stochastic and demand-robust while also accommodating *black-box* distributions. Other papers (e.g., [15]) studied algorithms for stochastic optimization given access to black-box distributions. There has also been work on two-stage stochastic models in which – as in our model – independent stochastic outcomes factor prominently. For example, Immorlica et al. [17] study a two-stage stochastic model in which each "client" activates independently and the realized scenario consists of all activated clients. The primary difference between their model and ours is that for us entire scenarios – rather than clients – activate independently. Moreover, reconfigurability of resources is not factored in their model.

Lastly, in our reduction from MinEMax to TruncatedTwoStage, we make use of a bound which has appeared before in other settings [19, 21, 6, 11]. For example, [6] use this bound to tightly estimate the optimum value in an optimization problem where the cost function is random and only marginal distributions for the coefficients of the cost function are known. Unlike our work, these works do not design approximation algorithms for two-stage problems.

1.2 Models

We now formally define our new MinEMax model and the prior models that we generalize. We study two-stage covering problems, defined as follows.

1.2.1 Two-Stage Covering

Let U be the universe of *clients* (or demand requirements), and let X be the set of *elements* that we can purchase. Every scenario $S_1, S_2, \ldots, S_m$ is a subset of clients. Let $sol(S_s)$ for $s \in [m]$ denote the sets in 2^X which are feasible to *cover* scenario S_s. In covering problems if $A \subseteq B$ and $A \in sol(S_s)$, then $B \in sol(S_s)$. We are also given a cost function $\mathsf{cost} : 2^X \times 2^X \to \mathbb{R}$. For a given a specification of cost, scenarios, clients, and feasibility constraints, we must find a set of elements $X_1 \subseteq X$ to be bought in the first stage, and a set of elements $X_2^{(s)} \subseteq X$ to be bought in the second stage s.t. $X_1 \cup X_2^{(s)} \in sol(S_s)$ for every s. Our goal is to find a solution of minimal cost where the cost of a solution is discussed below.

This paper makes the common assumption that cost is *linear*, i.e., $\mathsf{cost}(X_1, X_2^{(s)})$ equals $\mathsf{cost}(\emptyset, X_2^{(s)}) + \mathsf{cost}(X_1, \emptyset)$ for any $X_1, X_2^{(s)} \subseteq 2^X$. Let $\boldsymbol{X_2} := (X_2^{(1)}, \ldots, X_2^{(m)})$; throughout the paper a bold variable denotes a vector.

We now describe and discuss how different cost functions yield different two-stage covering models.

1.2.2 Prior Models

In the *demand-robust* two-stage covering model the cost of solution $(X_1, \boldsymbol{X_2})$ is the maximum cost over all the scenarios:

$$\mathsf{cost}_{\mathsf{Rob}}(X_1, \boldsymbol{X_2}) := \max_{s \in [m]} \Big\{ \mathsf{cost}(X_1, X_2^{(s)}) \Big\}. \tag{1}$$

In the *stochastic* two-stage covering model we are given a probability distribution $\mathcal{D}$ over m scenarios with which exactly one of them realizes; i.e. $\sum_{s \in [m]} \mathcal{D}(s) = 1$. The cost of solution $(X_1, \boldsymbol{X_2})$ is the expected cost:

$$\mathsf{cost}_{\mathsf{Stoch}}(X_1, \boldsymbol{X_2}) := \mathbb{E}_{s \sim \mathcal{D}}[\mathsf{cost}(X_1, X_2^{(s)})]. \tag{2}$$

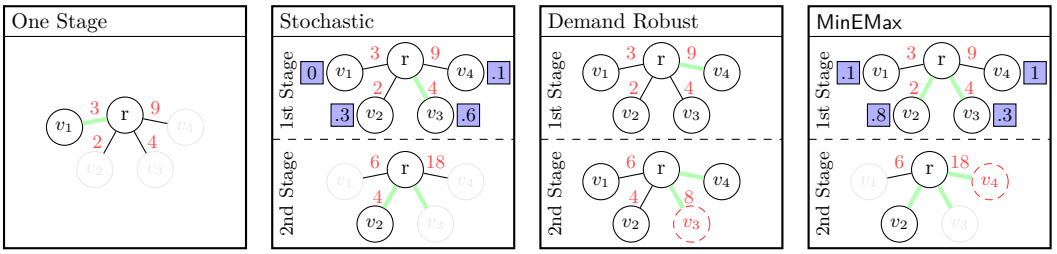
1.2.3 Our New MinEMax Model

In the MinEMax two-stage covering model we are given probabilities $\mathbf{p} = \{p_1, \ldots, p_m\}$ with which each scenario *independently* realizes. The cost of solution $(X_1, \boldsymbol{X_2})$ is the expected maximum cost among the realized scenarios:

$$\mathsf{cost}_{\mathsf{EMax}}(X_1, \boldsymbol{X_2}) := \mathbb{E}_{A \sim \mathbf{p}} \Big[\max_{s \in A} \Big\{ \mathsf{cost}(X_1, X_2^{(s)}) \Big\} \Big] \tag{3}$$

where A contains each s independently w.p. p_s. To avoid confusion, we reiterate that unlike the stochastic model, in MinEMax multiple scenarios may simultaneously appear in A because each of them independently realizes. We shall assume without loss of generality that $\sum_s p_s \geq 1$ throughout this paper since one can always ensure this without affecting solutions to the problem by adding dummy scenarios of cost 0 and probability 1.

As a concrete example of these models, consider the following star covering problem. We are given a star graph with root r and leaves $v_1, \ldots, v_m$. Each edge $e_i = (r, v_i)$ can be purchased in the first stage at cost c_i and in the second stage at an inflated cost $\sigma \cdot c_i$ for $\sigma > 1$. Our goal is to connect r to an *unknown* vertex v_s with minimum total two-stage cost.



(a) If scenario to be covered is known to be v_1, problem is trivial.

(b) Exactly one scenario realizes according to a probability distribution.

(c) Given first stage soln., adversary chooses costliest scenario.

(d) Given first stage soln., adversary chooses costliest realized scenario.

Figure 1 Star graph MinEMax for $m = 4$. Green edges: edges bought by solution. e_i labeled by its cost in each stage for $\sigma = 2$. Non-opaque second-stage node: realized scenario. Blue square: probability of scenario. Dashed red nodes: nodes chosen by an adversary.

In particular, v_s is only revealed after we purchase our first-stage edges, X_1, at which point we must purchase e_s in a second stage at cost $\sigma \cdot c_s$ if e_s was not already purchased in the first stage. In all three models we initially buy some set of edges. In the stochastic version of this problem a single v_s then appears according to a distribution and we must pay to connect v_s if we have not already. In the demand-robust version of this problem, v_s is always chosen so as to maximize our second stage cost. However, in our MinEMax version of this problem several v_s appear and we must pay for a budget of reconfigurable edge resources to be reused for every v_s. See Figure 1 for an illustration.

1.3 Technical Results and Intuition

We now discuss our technical results. As earlier noted, capturing the MinEMax objective seems challenging: scenarios may realize in exponentially-many ways and so even computing the objective seems computationally infeasible. We solve this issue by showing that to solve a MinEMax problem, P_{EMax}, it suffices to solve its TruncatedTwoStage version, P_{Trunc}. A TruncatedTwoStage problem is identical to a MinEMax problem but the cost of a solution $(X_1, \boldsymbol{X_2})$ is its truncated sum:

$$\mathsf{cost}_{\mathsf{Trunc}}(X_1, \boldsymbol{X_2}) := \min_B \left[B + \textstyle\sum_{s\in[m]} p_s \cdot (\mathsf{cost}(X_1, X_2^{(s)}) - B)^+ \right]. \tag{4}$$

We will later see that P_{Trunc} can be represented by an LP and, therefore, can be efficiently approximated by various rounding techniques. The following theorem shows that to approximate a MinEMax problem, it suffices to consider its TruncatedTwoStage version.

▶ **Theorem 1.** *Let P_{EMax} be a* MinEMax *problem and let P_{Trunc} be its* TruncatedTwoStage *version. An α-approximation algorithm for P_{Trunc} is a $\left(\frac{\alpha}{1-1/e}\right)$-approximation algorithm for P_{EMax}.*

The main observation we use to show this theorem is that a set of expensive scenarios with large total probability mass dominates the cost of a given MinEMax solution. We illustrate this observation with an example. Let $(X_1, \boldsymbol{X_2})$ be a solution for a MinEMax problem. Now WLOG let $\mathsf{cost}(X_1, X_2^{(s)}) \geq \mathsf{cost}(X_1, X_2^{(s+1)})$ for all s, i.e., the sth scenario is more expensive than the $(s+1)$th scenario for our solution. Let $M := [k]$ be the indices of the first k scenarios such that $\sum_{s\leq k} p_s$ is large; say, at least 1. Let the border $B := \mathsf{cost}(X_1, X_2^{(k)})$ be the cost of the least expensive scenario with an index in M. Because there is a great deal of

probability mass among scenarios in M we know that with large probability some scenario in M will always appear. Whenever a scenario of cost less than B appears we know that with good probability something in M has also appeared of greater cost. Thus, as far as the expected max is concerned, a scenario that costs less than B can be ignored. Lastly, while it is not immediately clear how to represent $\mathsf{cost}_{\mathsf{Trunc}}$ function in an LP, we show using a simple convexity argument how this can accomplished.

Next, we design approximation algorithms for two-stage covering problems in the MinEMax model.

▶ **Theorem 2.** *For two-stage covering problems there exist polynomial-time approximation algorithms with the following guarantees.*[1]

MinEMax *Problem*	*Steiner tree*	*UFL*	*MST*	*Min-cut*	*k-center*
Approximation	$\frac{30}{1-1/e}$	$\frac{8}{1-1/e}$	$O(\log n + \log m)$	$\frac{4}{1-1/e}$	$O(1)$

Our earlier Theorem 1 demonstrated that to solve a MinEMax problem, P_{EMax}, we need to only solve its TruncatedTwoStage version, P_{Trunc}. While it is not clear how to represent P_{EMax} with an LP, P_{Trunc} can be represented with an LP. Furthermore, by adapting previous two-stage optimization rounding techniques to the TruncatedTwoStage setting, we are able to approximately solve the TruncatedTwoStage versions of uncapacitated facility location (UFL), Steiner tree, minimum spanning tree (MST), and min-cut. We defer details on min-cut to the full version of our paper.

We use different techniques to give an approximation algorithm for k-center. The intuition for our k-center proof is similar to that of Theorem 1: Truncated costs approximate MinEMax cost. However, for k-center we truncate more aggressively. Rather than truncating costs of scenarios, we truncate distances in the input metric. To do this, we draw on methods of Chakrabarty and Swamy [7].[2]

It is also worth noting that Anthony et al. [4] proved hardness of approximation for a two-stage k-center problem. In particular, they show stochastic k-center where scenarios consist of *multiple* clients is as hard to approximate as dense k-subgraph. Thus, since our MinEMax model generalizes the stochastic model, we restrict our attention in k-center to scenarios consisting of *single* clients; otherwise our problem would be prohibitively hard to approximate. Since our scenarios consist of single clients the stochastic and demand-robust versions of the k-center problem we solve correspond to k-median and k-center respectively. We defer details on our k-center results to the full version of our paper.

Our last theorem shows that MinEMax generalizes the stochastic and demand-robust models as well as a Hybrid model which smoothly interpolates between stochastic and demand-robust optimization.

▶ **Theorem 3.** *An α-approximation for a two-stage covering algorithm in the* MinEMax *model implies an α-approximation for the corresponding two-stage covering problem in the stochastic, demand-robust, and* Hybrid *models.*

We defer a formal definition and discussion of the Hybrid model as well as the intuition and proof for Theorem 3 to the full version of our paper. As a corollary of Theorems 2 and 3, we immediately recover polynomial-time approximations for Hybrid MST, UFL, Steiner tree and min-cut.[3]

[1] The $O(1)$ in the k-center approximation is roughly 57.
[2] We also note here that, unlike the previous problems we study, the cost function in k-center is not linear as described in §1.2.
[3] Though not k-center since its cost function is not linear.

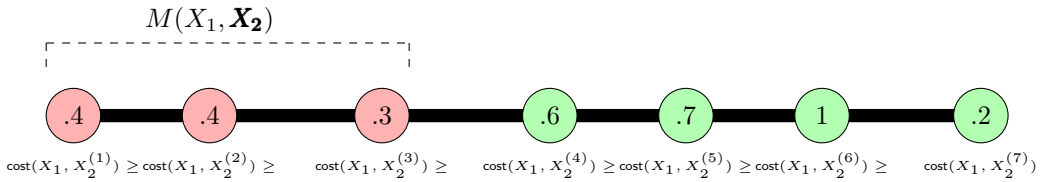


Figure 2 $M(X_1, \boldsymbol{X_2})$. $B(X_1, \boldsymbol{X_2}) = \mathsf{cost}(X_1, X_2^{(3)})$. Red circles: scenarios in $M(X_1, \boldsymbol{X_2})$. Green circles: all other scenarios. Numbers in circles: probabilities. Scenarios arranged left to right in descending order of $\mathsf{cost}(X_1, X_2^{(s)})$.

2 Reducing MinEMax to TruncatedTwoStage

In this section, we demonstrate a technique to simplify both computing and reasoning about $\mathsf{cost}_{\mathsf{EMax}}$ by reducing a MinEMax problem to a TruncatedTwoStage problem with only a small loss in the approximation factor. Specifically, we show the following theorem.

▶ **Theorem 1.** *Let P_{EMax} be a* MinEMax *problem and let P_{Trunc} be its* TruncatedTwoStage *version. An α-approximation algorithm for P_{Trunc} is a $\left(\frac{\alpha}{1-1/e}\right)$-approximation algorithm for P_{EMax}.*

As earlier noted, we show this by observing that a set of expensive scenarios with "large" total probability mass dominates the cost of a given MinEMax solution.

We begin by observing that the expected max of a set of independent random variables is approximately bounded by the most expensive of these random variables whose probabilities sum to 1. We remark that this result can be seen to follow from results regarding the "correlation gap" [2, 3] which show a similar bound where instead of max we have any sub-modular function. We give a different proof in §A for completeness that we find simpler in our setting where we consider the max and not any sub-modular function.

▶ **Lemma 4.** *Let $\boldsymbol{Y} = \{Y_1, \ldots, Y_m\}$ be a set of independent Bernoulli r.v.s, where Y_s is 1 with probability p_s, and 0 otherwise. Let $v_s \in \mathbb{R}_{\geq 0}$ be a value associated with Y_s. WLOG assume $v_s \geq v_{s+1}$ for $s \in [m-1]$. Let $b = \min\{a : \sum_{s=1}^{a} p_s \geq 1\}$. Then*

$$\left(1 - \frac{1}{e}\right)\left(v_b + \sum_s p_s \cdot (v_s - v_b)^+\right) \leq \mathbb{E}_{\boldsymbol{Y}}\left[\max_s\{Y_s \cdot v_s\}\right] \leq v_b + \sum_s p_s \cdot (v_s - v_b)^+,$$

where $x^+ := \max\{x, 0\}$.

For a *given solution* $(X_1, \boldsymbol{X}_2)$ to MinEMax, Lemma 4 yields a computationally tractable form of $\mathsf{cost}_{\mathsf{EMax}}$. Specifically, let our scenarios be indexed such that $\mathsf{cost}(X_1, X_2^{(s)}) \geq \mathsf{cost}(X_1, X_2^{(s+1)})$ and let b be the smallest positive integer such that $\sum_{s=1}^{b} p_s \geq 1$. We define the following terms analogous to those in the lemma (see Figure 2 for an illustration):

$$M(X_1, \boldsymbol{X}_2) := [b] \qquad \text{and} \qquad B(X_1, \boldsymbol{X}_2) := \mathsf{cost}(X_1, X_2^{(b)}). \tag{5}$$

Notice that $\sum_{s \in M(X_1, \boldsymbol{X_2})} p_s < 2$. Now, by letting $B(X_1, \boldsymbol{X_2})$ be v_b in Lemma 4, we can approximate $\mathsf{cost}_{\mathsf{EMax}}(X_1, \boldsymbol{X_2})$. However, we would like to estimate $\mathsf{cost}_{\mathsf{EMax}}(X_1, \boldsymbol{X}_2)$ within an LP where $(X_1, \boldsymbol{X}_2)$ are variables since our algorithms are LP based. Unfortunately, it is not clear how to capture v_b in an LP and so it is not clear how to directly use Lemma 4 to estimate $\mathsf{cost}_{\mathsf{EMax}}(X_1, \boldsymbol{X}_2)$ within an LP.

For this reason, we derive an even simpler form of the above approximation of the expected max which can be computed using an LP. In particular, we show that the expected max is approximately the $\mathsf{cost}_{\mathsf{Trunc}}$ objective. We remind the reader that, as per Eq.(4), $\mathsf{cost}_{\mathsf{Trunc}}(X_1, \boldsymbol{X_2}) := \min_B [B + \sum_{s\in[m]} p_s \cdot (\mathsf{cost}(X_1, X_2^{(s)}) - B)^+]$. The following lemma shows that the B achieving the minimum in $\mathsf{cost}_{\mathsf{Trunc}}(X_1, \boldsymbol{X_2})$ is $B(X_1, \boldsymbol{X_2})$ and therefore shows that $\mathsf{cost}_{\mathsf{Trunc}}$ is a good approximation of $\mathsf{cost}_{\mathsf{EMax}}$.

▶ **Lemma 5.** *Let $(X_1, \boldsymbol{X_2})$ be a solution to a* TruncatedTwoStage *or* MinEMax *problem. We have*

$$B(X_1, \boldsymbol{X_2}) = \arg\min_B \left[B + \sum_{s\in[m]} p_s \cdot (\mathsf{cost}(X_1, X_2^{(s)}) - B)^+\right],$$

where the arg min *takes the largest B minimizing the relevant quantity.*

Proof Sketch. The rough idea of the proof is to show that $B + \sum_s p_s(\mathsf{cost}(X_1, X_2^{(s)} - B)^+$ is convex in B and that $B(X_1, \boldsymbol{X_2})$ is a local minimum. In particular, imagine that B is currently set at $B(X_1, \boldsymbol{X_2})$ and consider what happens to $B + \sum_s p_s(\mathsf{cost}(X_1, X_2^{(s)} - B)^+$ if we shift B to be smaller. Recall that we have at least one probability mass across elements which are larger than B by definition of $B(X_1, \boldsymbol{X_2})$. Thus, when we shift B to be smaller, B decreases slower than $\sum_s p_s(\mathsf{cost}(X_1, X_2^{(s)} - B)^+$ increases and so $B + \sum_s p_s(\mathsf{cost}(X_1, X_2^{(s)} - B)^+$ becomes larger overall. The case when B is made larger is symmetric. The full proof is available in §A. ◀

Using Lemma 4 and Lemma 5, it is easy to show the following two lemmas. These lemmas – proved in §A – upper and lower bound the MinEMax cost of a solution with respect to its TruncatedTwoStage solution respectively.

▶ **Lemma 6.** *For feasible solution $(X_1, \boldsymbol{X}_2)$ of any P_{EMax} we have, $\mathsf{cost}_{EMax}(X_1, \boldsymbol{X}_2) \le \mathsf{cost}_{Trunc}(X_1, \boldsymbol{X}_2)$.*

▶ **Lemma 7.** *Let P_{EMax} be a* MinEMax *problem and P_{Trunc} be its truncated version. Let $(E_1, \boldsymbol{E_2})$ and $(T_1, \boldsymbol{T_2})$ be optimal solutions to P_{EMax} and P_{Trunc} respectively. We have that $\mathsf{cost}_{Trunc}(T_1, \boldsymbol{T_2}) \le \left(\frac{1}{1-1/e}\right) \mathsf{cost}_{EMax}(E_1, \boldsymbol{E_2})$.*

The preceding lemmas allow us to conclude that an α-approximation algorithm for a TruncatedTwoStage problem is an $O(\alpha)$-approximation algorithm for the corresponding MinEMax problem.

Proof of Theorem 1. Let $(\hat{T}_1, \hat{\boldsymbol{T}}_{\boldsymbol{2}})$ be the solution returned by an α-approximation algorithm for P_{Trunc}. Let $(E_1, \boldsymbol{E_2})$ and $(T_1, \boldsymbol{T_2})$ be the optimal solutions to P_{EMax} and P_{Trunc} respectively. By Lemma 6 we have $\mathsf{cost}_{\mathsf{EMax}}(\hat{T}_1, \hat{\boldsymbol{T}}_{\boldsymbol{2}}) \le \mathsf{cost}_{\mathsf{Trunc}}(\hat{T}_1, \hat{\boldsymbol{T}}_{\boldsymbol{2}})$. Since $(\hat{T}_1, \hat{\boldsymbol{T}}_{\boldsymbol{2}})$ is an α-approximation we have this is at most $\alpha \cdot \mathsf{cost}_{\mathsf{Trunc}}(T_1, \boldsymbol{T_2})$. Applying Lemma 7 this is at most $\left(\frac{\alpha}{1-1/e}\right) \mathsf{cost}_{\mathsf{EMax}}(E_1, \boldsymbol{E_2})$. Since any solution that is feasible for P_{Trunc} is also feasible for P_{EMax}, we conclude that $(\hat{T}_1, \hat{\boldsymbol{T}}_{\boldsymbol{2}})$ is a feasible solution for P_{EMax} with cost in P_{EMax} at most $\left(\frac{\alpha}{1-1/e}\right) \mathsf{cost}_{\mathsf{EMax}}(E_1, \boldsymbol{E_2})$, giving our theorem. ◀

3 Applications to Linear Two-Stage Covering Problems

In this section we give an $O(\log n + \log m)$-approximation algorithm for MinEMax MST and $O(1)$ approximation algorithms for MinEMax Steiner tree, MinEMax facility location, and MinEMax min-cut. Our algorithms are LP based. To derive our algorithms we use our

reduction from §2 to transform a MinEMax problem into a TruncatedTwoStage problem with only a small constant loss in the approximation factor. This transformation allows us to adapt existing LP rounding techniques in which every scenario has a rounding cost close to its fractional cost [22, 15, 24] to solve our TruncatedTwoStage problems and, therefore, our MinEMax problems.

We first give two general techniques to solve a TruncatedTwoStage problem.

3.1 General Techniques

Our *first technique* is to represent $\mathsf{cost}_{\mathsf{Trunc}}$ as an LP objective. For this technique we need to extend the definition of $\mathsf{cost}_{\mathsf{Trunc}}$ from an integral solution $(X_1, \boldsymbol{X_2})$ to a fractional solution $(x_1, \boldsymbol{x_2})$. To do so, in each of our problems we locally define $\mathsf{cost}(x_1, x_2^{(s)})$ for fractional solution $(x_1, x_2^{(s)})$ to scenario s and let $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2})$ be defined similarly to the integral case, i.e. for fractional $(x_1, \boldsymbol{x_2})$,

$$\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2}) := \min_B \left[B + \sum_s p_s (\mathsf{cost}(x_1, x_2(s)) - B)^+ \right]. \tag{6}$$

Given a minimization LP, it is easy to see that by introducing an additional variable to represent B and additional variables to represent $(\mathsf{cost}(x_1, x_2(s)) - B)^+$ for every s, we can represent $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2})$ in an LP. For cleanliness of exposition, when we write our LPs we omit these additional variables and simply write our objective as "$\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2})$." Moreover, even though some of our LPs have an exponential number of constraints, we rely on the existence of efficient separation oracles for these LPs. It is easy to verify that this holds even after one introduces the additional variables needed to represent $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2})$.

We also extend M and B from the integral case as defined in §2 to the fractional case in the following natural way. Given a fractional solution $(x_1, \boldsymbol{x_2})$ and a cost function on fractional solutions, cost, WLOG let our scenarios be indexed such that $\mathsf{cost}(x_1, x_2^{(s)}) \geq \mathsf{cost}(x_1, x_2^{(s+1)})$. Let b be the smallest positive integer such that $\sum_{s=1}^{b} p_s \geq 1$. For fractional $(x_1, \boldsymbol{x_2})$, we define

$$M(x_1, \boldsymbol{x}_2) := [b] \tag{7}$$

$$B(x_1, \boldsymbol{x}_2) := \min_{s \in M(x_1, \boldsymbol{x}_2)} \mathsf{cost}(x_1, x_2^{(s)}). \tag{8}$$

▶ Remark 8. It is easy to verify that the proof of Lemma 5 also holds for $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2})$ for fractional $(x_1, \boldsymbol{x_2})$. We will therefore invoke it on fractional $(x_1, \boldsymbol{x_2})$, even though it is stated only for integral $(X_1, \boldsymbol{X_2})$.

Our *second technique* is a generic rounding technique for TruncatedTwoStage problems. Several past works in two-stage optimization show that it is possible to round an LP solution such that the resulting integral solution has cost roughly the same as the fractional solution for *every scenario*. We prove the following lemma to make use of such rounding algorithms.

▶ **Lemma 9.** *Let P_{Trunc} be a* TruncatedTwoStage *problem. Let $(X_1, \boldsymbol{X_2})$ and $(Y_1, \boldsymbol{Y_2})$ be integral or fractional solutions to P_{Trunc}. If for every scenario s we have $\mathsf{cost}(X_1, X_2^{(s)}) \leq c \cdot \mathsf{cost}(Y_1, Y_2^{(s)})$ then*

$$\mathsf{cost}_{Trunc}(X_1, \boldsymbol{X_2}) \leq c \cdot \mathsf{cost}_{Trunc}(Y_1, \boldsymbol{Y_2}).$$

Proof. We have

$$\mathsf{cost}_{\mathsf{Trunc}}(X_1, \boldsymbol{X_2}) = \min_B \left[B + \sum_s p_s \cdot (\mathsf{cost}(X_1, X_2^{(s)}) - B)^+) \right] \tag{9}$$

$$\leq c \cdot B(Y_1, \boldsymbol{Y_2}) + \sum_s p_s \cdot (\mathsf{cost}(X_1, X_2^{(s)}) - c \cdot B(Y_1, \boldsymbol{Y_2}))^+) \tag{10}$$

$$\leq c \cdot B(Y_1, \boldsymbol{Y_2}) + \sum_s p_s \cdot (c \cdot \mathsf{cost}(Y_1, Y_2^{(s)}) - c \cdot B(Y_1, \boldsymbol{Y_2}))^+) \tag{11}$$

$$= c \cdot \left(B(Y_1, \boldsymbol{Y_2}) + \sum_s p_s \cdot (\mathsf{cost}(Y_1, Y_2^{(s)}) - B(Y_1, \boldsymbol{Y_2}))^+) \right) \tag{12}$$

$$= c \cdot \mathsf{cost}_{\mathsf{Trunc}}(Y_1, \boldsymbol{Y_2}) \tag{13}$$

where Eq.(10) is by letting $B = c \cdot B(Y_1, \boldsymbol{Y_2})$, Eq.(11) is by $\mathsf{cost}(X_1, X_2^{(s)}) \leq c \cdot \mathsf{cost}(Y_1, Y_2^{(s)})$ and Eq.(13) is by Lemma 5. ◀

3.2 Steiner Tree

In this section we give a $\left(\frac{30}{1-1/e}\right)$-approximation for MinEMax rooted Steiner tree.

▶ **Definition 10** (MinEMax Rooted Steiner tree)**.** *We are given a graph $G = (V, E)$, a root $r \in V$, a cost c_e for each edge e. We are also given scenarios $S_1, \ldots, S_m \subseteq V$, each with an associated probability p_s and an* inflation factor *$\sigma_s > 0$. We must find a first stage solution $X_1 \subseteq E$ and a second-stage solution for every scenario, $X_2^{(j)} \subseteq E$. A solution is feasible if for every s we have $X_1 \cup X_2^{(s)}$ connects $\{r\} \cup S_s$. The cost for scenario s in solution $(X_1, \boldsymbol{X_2})$ is*

$$\mathsf{cost}(X_1, X_2^{(s)}) := \sum_{e \in X_1} c_e + \sigma_s \cdot \sum_{e \in X_2^{(s)}} c_e. \tag{14}$$

The total cost we pay for $(X_1, \boldsymbol{X_2})$ is $\mathsf{cost}_{EMax}(X_1, \boldsymbol{X_2}) := E_{A \sim \boldsymbol{p}} \left[\max_{s \in A} \{\mathsf{cost}(X_1, X_2^{(s)})\} \right]$.

Our algorithm is based on an LP rounding algorithm of Gupta et al. [16] for two-stage stochastic Steiner tree. Roughly, we use Lemma 9 to argue that the first stage solution for every optimal TruncatedTwoStage solution is, up to small constants, a tree rooted at r. This structural property allows us to write an LP that approximately captures TruncatedTwoStage Steiner tree. Gupta et al. [16] showed that this LP can be rounded s.t. every scenario has a good cost. We then combine this rounding with Lemma 9 to derive an approximation algorithm for TruncatedTwoStage Steiner tree, which is sufficient for approximating MinEMax Steiner tree by Theorem 1.

We begin by arguing that up to small constants, the optimal first stage solution is a tree rooted at r.

▶ **Lemma 11.** *There exists an integral solution $(\hat{X}_1, \hat{\boldsymbol{X}}_{\boldsymbol{2}})$ to* TruncatedTwoStage *Steiner tree s.t. $G[\hat{X}_1]$ is a tree rooted at r and $\mathsf{cost}_{Trunc}(\hat{X}_1, \hat{\boldsymbol{X}}_{\boldsymbol{2}}) \leq 2 \cdot \mathsf{cost}_{Trunc}(O_1, \boldsymbol{O_2})$, where $(O_1, \boldsymbol{O_2})$ is the optimal solution to* TruncatedTwoStage *Steiner tree.*

Proof. Lemma 4.1 of Dhamdhere et al. [9] shows that given $(O_1, \boldsymbol{O_2})$ it is possible to modify it to a feasible solution $(\hat{X}_1, \hat{\boldsymbol{X}}_{\boldsymbol{2}})$ such that $G[\hat{X}_1]$ is a tree rooted at r and $\mathsf{cost}(\hat{X}_1, \hat{X}_2^{(s)}) \leq 2 \cdot \mathsf{cost}(O_1, O_2^{(s)})$ for every s. It follows by Lemma 9 that $\mathsf{cost}_{\mathsf{Trunc}}(\hat{X}_1, \hat{\boldsymbol{X}}_{\boldsymbol{2}}) \leq 2 \cdot \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2})$. ◀

We now describe how to formulate an LP that leverages the structural property in Lemma 11. In particular, this indicates that as one gets closer to r, one must fractionally buy edges to a greater and greater extent. This constraint can be captured in an LP. Specifically, every node in a scenario (a.k.a. terminal) is the source of one unit of flow that is ultimately routed to r; this flow follows a path whose fractional "first stage-ness" is monotonically increasing.

More formally, we copy each edge $e = \{u, v\}$ into two directed edges (u, v) and (v, u). Let $\vec{e}$ be either one of these directed edges. Next, for each such directed edge $\vec{e}$ and every terminal in $t \in \bigcup_s S_s$, we define variables $r_1(t, \vec{e})$ and $r_2^{(s)}(t, \vec{e})$ for every s to represent how much t is connected to r by e in the first stage and in scenario s, respectively. Also, for undirected edge e, define variables $x_1(e)$ and $x_2^{(s)}(e)$ to stand for how much we buy e in the first stage and scenario s, respectively. For fractional $(x_1, \boldsymbol{x_2})$, we define

$$\mathsf{cost}_{\mathsf{Trunc}}(x_1, x_2^{(s)}) := \sum_e c_e \cdot x_1(e) + \sigma_s \cdot c_e \cdot x_2(e),$$

which as described by Eq.(6) also defines $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x}_2)$. Letting $\delta^-(v)$ and $\delta^+(v)$ stand for all directed edges going into and out of v, respectively. The following is our LP.

$$\begin{aligned}
\min \quad & \mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2}) && && \text{(ST LP)}\\
\text{s.t.} \quad & \sum_{\vec{e} \in \delta^+(v)} r_1(t, \vec{e}) + r_2^{(s)}(t, \vec{e}) = \sum_{\vec{e} \in \delta^-(v)} r_1(t, \vec{e}) + r_2^{(s)}(t, \vec{e}) && \forall s, t \in S_s, v \notin \{t, r\}\\
& \sum_{\vec{e} \in \delta^+(t)} r_1(t, \vec{e}) + r_2^{(s)}(t, \vec{e}) - \sum_{\vec{e} \in \delta^-(t)} r_1(t, \vec{e}) + r_2^{(s)}(t, \vec{e}) \geq 1 && \forall s, t \in S_s\\
& \sum_{\vec{e} \in \delta^-(v)} r_1(t, \vec{e}) \leq \sum_{\vec{e} \in \delta^+(v)} r_1(t, \vec{e}) && \forall s, t \in S_s, v \notin \{t, r\}\\
& r_1(t, \vec{e}) \leq x_1(e); r_2^{(s)}(t, \vec{e}) \leq x_2^{(s)}(e) && \forall s, t \in S_s, \vec{e}\\
& r, x_1, \boldsymbol{x_2} \geq 0
\end{aligned}$$

Notably, the third family of constraints enforces that terminal t is serviced by the first stage more and more as one moves closer to the root. The characteristic vector of $(\hat{X}_1, \hat{\boldsymbol{X}}_{\boldsymbol{2}})$ as described in Lemma 11 gives a feasible solution to ST LP. As a result, Lemma 11 demonstrates that ST LP has nearly optimal objective as stated in the following corollary.

▶ **Corollary 12.** *Let $(x_1, \boldsymbol{x_2})$ be the optimal solution of ST LP. We have $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2}) \leq 2 \cdot \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2})$, where $(O_1, \boldsymbol{O_2})$ is the optimal solution to* TruncatedTwoStage *Steiner tree.*

Proof. Let $(\hat{x}_1, \hat{\boldsymbol{x}}_{\boldsymbol{2}})$ be the characteristic vector of $(\hat{X}_1, \hat{\boldsymbol{X}}_{\boldsymbol{2}})$ from Lemma 11. Fix an arbitrary terminal t. Let P_2 for terminal t be the shortest path from t to $\hat{X}_1$ in $G[\hat{X}_2]$. Let u_t be the sink of P_2 and let P_1 be the shortest path from u_t to r in $G[\hat{X}_1]$. Notice that $(\hat{x}_1, \hat{\boldsymbol{x}}_{\boldsymbol{2}})$ along with r_2 which sends one unit of flow from t to u_t along P_2 and r_1 which sends one unit of flow from u_t to r along P_1 for every t is a feasible solution to ST LP. Moreover, notice that cost of this solution is $\mathsf{cost}_{\mathsf{Trunc}}(\hat{x}_1, \boldsymbol{x_2}) = \mathsf{cost}_{\mathsf{Trunc}}(X_1, \boldsymbol{X_2}) \leq 2 \cdot \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2})$ by Lemma 11. ◀

Previous work of Gupta et al. [16] shows that it is possible to round a fractional solution of ST LP such that *every* scenario has a good cost.

▶ **Lemma 13** ([16])**.** *A fractional solution $(x_1, \boldsymbol{x_2})$ to ST LP can be rounded in polynomial time to a feasible integral solution $(X_1, \boldsymbol{X_2})$ s.t. $\mathsf{cost}(X_1, X_2^{(s)}) \leq 15 \cdot \mathsf{cost}(x_1, x_2^{(s)})$ for every s.*

Since Corollary 12 gives ST LP has a good optimal solution, we can round ST LP such that every scenario has a low cost. Now Lemma 9 tells us that such a rounding preserves the cost of a solution for TruncatedTwoStage optimization. This gives the following theorem.

▶ **Theorem 14.** MinEMax *Steiner tree can be* $\left(\frac{30}{1-1/e}\right)$*-approximated in polynomial time.*

Proof. Our algorithm first solves ST LP to get fractional solution $(x_1, \boldsymbol{x_2})$. Next, we apply Lemma 13 to round $(x_1, \boldsymbol{x_2})$ in polynomial time to give $(X_1, \boldsymbol{X_2})$ as our solution. Thus, we have

$$\begin{aligned} \mathsf{cost}_{\mathsf{Trunc}}(X_1, \boldsymbol{X_2}) &\le 15 \cdot \mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2}) && \text{(by Lemma 9, Lemma 13)} \\ &\le 30 \cdot \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2}), && \text{(by Corollary 12)} \end{aligned}$$

where $(O_1, \boldsymbol{O_2})$ is the optimal TruncatedTwoStage Steiner tree solution. This implies we have a 30-approximation algorithm for TruncatedTwoStage Steiner tree. Now by Theorem 1, we have a $\left(\frac{30}{1-1/e}\right)$-approximation for MinEMax Steiner tree.

Lastly, each of our subroutines has a polynomial runtime by previous lemmas, and so we conclude that our algorithm has a polynomial runtime. ◀

3.3 Uncapacitated Facility Location

In this section we give a polynomial-time $\left(\frac{8}{1-1/e}\right)$-approximation algorithm for MinEMax uncapacitated facility location (UFL).

▶ **Definition 15** (MinEMax UFL)**.** *We are given a set of facilities F and a set of clients $\mathcal{D}$ with a metric c_{ij} specifying the distances between every client j and facility i. We are also given scenarios $S_1, \ldots, S_m \subseteq \mathcal{D}$, where in scenario S_s client j has demand $d_j^s \in \{0,1\}$[4], and a probability p_s for each scenario. Facility i's opening cost is $f_{1,i}$ in the first stage and $f_{2,i}^{(s)}$ in scenario S_s. These opening costs can be ∞, which indicates the facility cannot be opened. A feasible solution consists of a set of first and second stage facilities $(X_1, \boldsymbol{X_2})$ s.t. $X_1 \cup \bigcup_s X_2^{(s)} \neq \emptyset$. The cost for scenario s in solution $(X_1, \boldsymbol{X_2})$ is*

$$\mathsf{cost}(X_1, X_2^{(s)}) := \sum_{i \in X_1} f_{1,i} + \sum_{i \in X_2^{(s)}} f_{2,i}^{(s)} + \sum_{j \in S_s} \min_{i \in X_1 \cup X_2^{(s)}} c_{ij}.$$

The total cost of solution $(X_1, \boldsymbol{X_2})$ is $\mathsf{cost}_{EMax}(X_1, \boldsymbol{X_2}) := \mathbb{E}_{A \sim \boldsymbol{p}}\left[\max_{s \in A}\{\mathsf{cost}(X_1, X_2^{(s)})\}\right]$.

Our algorithm is based on the work of Ravi and Sinha [22] on two-stage stochastic UFL. This work shows how to round an LP such that every scenario has a "good" cost after rounding. Applying Lemma 9 to this rounding gives an algorithm that approximates TruncatedTwoStage UFL, which by Theorem 1 is sufficient to approximate MinEMax UFL.

We use the following LP. Variable $z_{ij}^{(s)}$ corresponds to whether client j is served by facility i in scenario s. Variables $x_1(i)$ and $x_2^{(s)}(i)$ corresponds to whether facility i is opened in the first stage or scenario s, respectively. For a fractional solution $(x_1, \boldsymbol{x_2})$, we define

$$\mathsf{cost}(x_1, x_2^{(s)}) := \sum_{i \in F}\left[x_1(i) \cdot f_{1,i} + x_2^{(s)}(i) \cdot f_{2,i}^{(s)} + \sum_{j \in \mathcal{D}} \hat{z}_{ij}^{(s)} \cdot c_{ij}\right],$$

[4] This easily generalizes to more demand.

where $\hat{z}_{ij}^{(s)}$ is the natural fractional assignment given fractional facilities $(x_1, x_2^{(s)})$; namely, one that sends clients to their nearest fractionally opened facilities. As described by Eq.(6), this definition of $\mathsf{cost}(x_1, x_2^{(s)})$ defines $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2})$ for fractional $(x_1, \boldsymbol{x_2})$, which allows us to define our LP.

$$\begin{aligned} \min \quad & \mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2}) && && \text{(UFL LP)} \\ \text{s.t.} \quad & \sum_{i \in F} z_{ij}^{(s)} \geq d_j^{(s)} && \forall j \in \mathcal{D}, \forall s \\ & z_{ij}^{(s)} \leq x_1(i) + x_2^{(s)}(i) && \forall i \in F, \forall j \in \mathcal{D}, \forall s \\ & 0 \leq x_1, \boldsymbol{x_2}, \boldsymbol{z} \end{aligned}$$

Note that an integral solution to the above LP is a feasible solution for MinEMax UFL. Ravi and Sinha showed how to round this LP.

▶ **Lemma 16** (Theorem 2, Lemma 1 in [22]). *Given a fractional solution $(x_1, \boldsymbol{x_2})$ to UFL LP, it is possible to round it to integral $(X_1, \boldsymbol{X_2})$ in polynomial-time s.t. for every scenario s we have $\mathsf{cost}(X_1, X_2^{(s)}) \leq 8 \cdot \mathsf{cost}(x_1, x_2^{(s)})$.*

We now give our approximation algorithm for MinEMax UFL.

▶ **Theorem 17.** MinEMax *UFL can be $\left(\frac{8}{1-1/e}\right)$-approximated in polynomial time.*

Proof. Our algorithm starts by solving UFL LP to get a fractional $(x_1, \boldsymbol{x_2})$. Next, round $(x_1, \boldsymbol{x_2})$ using Lemma 16 to integral $(X_1, \boldsymbol{X_2})$. Return $(X_1, \boldsymbol{X_2})$.

Let $(O_1, \boldsymbol{O_2})$ be the optimal integral solution to the TruncatedTwoStage instance of our problem and let $(o_1, \boldsymbol{o_2})$ be its corresponding characteristic function. By definition, $\mathsf{cost}_{\mathsf{Trunc}}(o_1, \boldsymbol{o_2}) = \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2})$. Now using Lemma 9 and Lemma 16 it follows that

$$\mathsf{cost}_{\mathsf{Trunc}}(X_1, X_2) \leq 8 \cdot \mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2}).$$

Since $(o_1, \boldsymbol{o_2})$ feasible for UFL LP, we get

$$\mathsf{cost}_{\mathsf{Trunc}}(X_1, X_2) \quad \leq \quad 8 \cdot \mathsf{cost}_{\mathsf{Trunc}}(o_1, \boldsymbol{o_2}) \quad = \quad 8 \cdot \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2}).$$

Thus, our algorithm is an 8-approximation for TruncatedTwoStage UFL. Applying Theorem 1 gives a $\left(\frac{8}{1-1/e}\right)$-approximation for MinEMax UFL.

Lastly, notice that our algorithm is trivially polynomial-time. ◀

3.4 MST

In this section we give a randomized polynomial-time algorithm which with high probability has expected cost $O(\log n + \log m)$ times the optimal MinEMax minimum spanning tree (MST) on an n-node graph with m different scenarios.

▶ **Definition 18** (MinEMax MST). *We are given a graph $G = (V, E)$ where $|V| = n$, a set of m scenarios $S_1, \ldots S_m$ where each scenario S_s has an associated second-stage cost function $\mathsf{cost}_2^{(s)} : E \to \mathbb{Z}^+$ and a probability p_s. We are also given a first-stage cost function, $\mathsf{cost}_1 : E \to \mathbb{Z}^+$. We must provide a first stage solution $X_1 \subseteq E$ and a solution $X_2^{(s)} \subseteq E$ for every scenario s, which is feasible if $G[X_1 \cup X_2^{(s)}]$ spans V for every s. The cost for scenario s in solution $(X_1, \boldsymbol{X_2})$ is*

$$\mathsf{cost}(X_1, X_2^{(s)}) := \sum_{e \in X_1} \mathsf{cost}_1(e) + \sum_{e \in X_2^{(s)}} \mathsf{cost}_2^{(s)}(e). \tag{15}$$

The total cost for solution $(X_1, \boldsymbol{X_2})$ is $\mathsf{cost}_{EMax}(X_1, \boldsymbol{X_2}) := \mathbb{E}_{A \sim \boldsymbol{p}} \left[\max_{s \in A} \{\mathsf{cost}(X_1, X_2^{(s)})\}\right]$.

Our algorithm is based on the work of Dhamdhere et al. [10] on two-stage stochastic MST. They give a rounding technique that produces integral solutions where every scenario has a cost close to the fractional cost. Using this rounding, and applying Lemma 9, we get an approximation algorithm for TruncatedTwoStage MST, which by Theorem 1 is also sufficient to approximate MinEMax MST.

Notice that since MinEMax generalizes two-stage robust optimization, our MinEMax result gives a $O(\log n + \log m)$ approximation for two-stage robust MST as a corollary. To the best of our knowledge, this is the first non-trivial algorithm for two-stage robust MST.

Our algorithm is based on an LP. We have $m+1$ variables for each edge e, namely $x_1(e)$ and $x_2^{(s)}(e)$ for $s \in [m]$ indicating if we take e in the first stage and in the second stage for scenario s, respectively. For a fractional solution $(x_1, \boldsymbol{x_2})$, we define

$$\mathsf{cost}(x_1, x_2^{(s)}) := \sum_e x_1(e) \cdot \mathsf{cost}_1(e) + x_2^{(s)}(e) \cdot \mathsf{cost}_2(e), \tag{16}$$

which as described in Eq.(6), defines $\mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x}_2)$ for fractional $(x_1, \boldsymbol{x_2})$. Letting $\delta(S)$ be all edges with exactly one endpoint in $S \subseteq V$. The following is our LP.

$$\begin{aligned} \min \quad & \mathsf{cost}_{\mathsf{Trunc}}(x_1, \boldsymbol{x_2}) && && \text{(MST LP)} \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} \Big(x_1(e) + x_2^{(s)}(e) \Big) \geq 1 && \forall \emptyset \subset S \subset V, s \in [m] \\ & x_1, \boldsymbol{x_2} \geq 0 \end{aligned}$$

Note that an integral solution to MST LP is a feasible solution for the TruncatedTwoStage MST problem as a set of edges with at least one edge leaving every cut is a spanning tree.[5] Also, although this LP has super-polynomial constraints, it is easy to obtain an efficient separation by solving min-cut; see Dhamdhere et al. [10].

We need the following result of Dhamdhere et al. [10] to round MST LP such that every scenario has a low cost.

▶ **Lemma 19** ([10]). *It is possible to randomly round a feasible fractional solution $(x_1, \boldsymbol{x_2})$ to MST LP to an integral solution $(X_1, \boldsymbol{X_2})$ in polynomial time s.t. with probability at least $1 - \frac{1}{mn^2}$ for every scenario s we have $\mathbb{E}[\mathsf{cost}(X_1, X_2^{(s)})] \leq \mathsf{cost}(x_1, x_2^{(s)}) \cdot (40 \log n + 16 \log m)$. Here the expectation is taken over the randomness of our rounding and m is the number of scenarios.*

We can now design our approximation algorithm for MinEMax MST.

▶ **Theorem 20.** *There exists a randomized polynomial-time algorithm that with probability at least $1 - \frac{1}{mn^2}$ in expectation $O(\log n + \log m)$-approximates* MinEMax *MST where $n = |V|$ and m is the number of scenarios.*

Proof. Our algorithm starts by following MST LP to get a fractional solution $(x_1, \boldsymbol{x_2})$. Next, apply Lemma 19 to round $(x_1, \boldsymbol{x_2})$ to an integral solution $(X_1, \boldsymbol{X_2})$. Return $(X_1, \boldsymbol{X_2})$.

Next consider the cost of $(X_1, \boldsymbol{X_2})$. Let $(O_1, \boldsymbol{O_2})$ be the optimal integral solution to our TruncatedTwoStage MST problem and let $(o_1, \boldsymbol{o_2})$ be the corresponding characteristic vector. Notice that $(o_1, \boldsymbol{o_2})$ is a feasible solution to MST LP. Moreover, it is easy to

[5] If such a solution has any cycles it is not necessarily an MST, though one can always delete an edge from such a cycle and improve the cost of the solution.

verify that $\mathsf{cost}_{\mathsf{Trunc}}(o_1, \boldsymbol{o_2}) = \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2})$. Taking expectations over the randomness of our algorithm and applying Lemma 9 and Lemma 19, we have with probability at least $1 - \frac{1}{mn^2}$ that

$$\begin{aligned}\mathbb{E}[\mathsf{cost}_{\mathsf{Trunc}}(X_1, \boldsymbol{X_2})] &\leq (40 \log n + 16 \log m) \cdot \mathsf{cost}_{\mathsf{Trunc}}(o_1, \boldsymbol{o_2}) \\ &= (40 \log n + 16 \log m) \cdot \mathsf{cost}_{\mathsf{Trunc}}(O_1, \boldsymbol{O_2}).\end{aligned}$$

Thus, with probability at least $1 - \frac{1}{mn^2}$ our algorithm's expected TruncatedTwoStage cost is within $(40 \log n + 16 \log m)$ of the cost of the optimal TruncatedTwoStage MST solution. We conclude by Theorem 1 that with high probability in expectation our algorithm $O(\log n + \log m)$-approximates MinEMax MST.[6]

Our algorithm is trivially polynomial-time by the separability of our LP and Lemma 19. ◀

References

1. Carlo Acerbi and Dirk Tasche. Expected shortfall: a natural coherent alternative to value at risk. *Economic notes*, 31(2):379–388, 2002.
2. Shipra Agrawal, Yichuan Ding, Amin Saberi, and Yinyu Ye. Price of correlations in stochastic optimization. *Operations Research*, 60(1):150–162, 2012.
3. Saeed Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. *SIAM Journal on Computing (SICOMP)*, 43(2):930–972, 2014.
4. Barbara M. Anthony, Vineet Goyal, Anupam Gupta, and Viswanath Nagarajan. A Plant Location Guide for the Unsure. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2008.
5. D. Bertsimas, D. Brown, and C. Caramanis. Theory and Applications of Robust Optimization. *SIAM Review*, 53(3):464–501, 2011.
6. Dimitris Bertsimas, Karthik Natarajan, and Chung-Piaw Teo. Probabilistic combinatorial optimization: Moments, semidefinite programming, and asymptotic bounds. *SIAM Journal on Optimization*, 15(1):185–209, 2004.
7. Deeparnab Chakrabarty and Chaitanya Swamy. Interpolating between k-Median and k-Center: Approximation Algorithms for Ordered k-Median. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, pages 29:1–29:14, 2018.
8. Erick Delage and Yinyu Ye. Distributionally Robust Optimization Under Moment Uncertainty with Application to Data-Driven Problems. *Operations Research*, 58(3):595–612, 2010.
9. Kedar Dhamdhere, Vineet Goyal, R. Ravi, and Mohit Singh. How to Pay, Come What May: Approximation Algorithms for Demand-Robust Covering Problems. In *Proceedings of the Symposium on the Foundations of Computer Science (FOCS)*, pages 367–378, 2005.
10. Kedar Dhamdhere, R Ravi, and Mohit Singh. On two-stage stochastic minimum spanning trees. In *Proceedings of International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 321–334, 2005.
11. Anulekha Dhara and Karthik Natarajan. On the polynomial solvability of distributionally robust k-sum optimization. *Optimization Methods and Software*, 32(4):738–753, 2017.
12. Joel Goh and Melvyn Sim. Distributionally Robust Optimization and Its Tractable Approximations. *Operations Research*, 58(4-part-1):902–917, 2010.
13. Daniel Golovin, Vineet Goyal, Valentin Polishchuk, R. Ravi, and Mikko Sysikaski. Improved approximations for two-stage min-cut and shortest path problems under uncertainty. *Mathematical Programming*, 149(1):167–194, February 2015.

[6] Although Theorem 1 and Lemma 9 do not explicitly account for an expectation taken over the randomness of an algorithm, it is easy to verify that the such an expectation does not affect these results.

14 Anupam Gupta, Viswanath Nagarajan, and R Ravi. Thresholded covering algorithms for robust and max-min optimization. *Automata, Languages and Programming*, pages 262–274, 2010.

15 Anupam Gupta, Martin Pal, R Ravi, and Amitabh Sinha. Boosted sampling: approximation algorithms for stochastic optimization. In *Proceedings of the Symposium on the Theory of Computing (STOC)*, pages 417–426, 2004.

16 Anupam Gupta, R Ravi, and Amitabh Sinha. An edge in time saves nine: LP rounding approximation algorithms for stochastic network design. In *Proceedings of the Symposium on the Foundations of Computer Science (FOCS)*, pages 218–227, 2004.

17 Nicole Immorlica, David Karger, Maria Minkoff, and Vahab S Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 691–700, 2004.

18 Ruiwei Jiang and Yongpei Guan. Risk-averse two-stage stochastic program with distributional ambiguity. *Operations Research*, 66(5):1390–1405, 2018.

19 TL Lai and Herbert Robbins. Maximally dependent random variables. *Proceedings of the National Academy of Sciences*, 73(2):286–288, 1976.

20 Andre Linhares and Chaitanya Swamy. Approximation Algorithms for Distributionally-Robust Stochastic Optimization with Black-Box Distributions. In *Proceedings of the Symposium on the Theory of Computing (STOC)*, 2019.

21 Isaac Meilijson and Arthur Nádas. Convex majorization with an application to the length of critical paths. *Journal of Applied Probability*, 16(3):671–677, 1979.

22 R Ravi and Amitabh Sinha. Hedging uncertainty: Approximation algorithms for stochastic optimization problems. In *Proceedings of International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 101–115, 2004.

23 Herbert Scarf. A Min-Max Solution of an Inventory Problem. *Studies in the Mathematical Theory of Inventory and Production*, pages 201–209, 1958.

24 David B. Shmoys and Chaitanya Swamy. An Approximation Scheme for Stochastic Linear Programming and Its Application to Stochastic Integer Programs. *Journal of the ACM (JACM)*, 53(6):978–1012, November 2006.

25 Anthony Man-Cho So, Jiawei Zhang, and Yinyu Ye. Stochastic combinatorial optimization with controllable risk aversion level. *Mathematics of Operations Research*, 34(3):522–537, 2009.

26 Aravind Srinivasan. Approximation algorithms for stochastic and risk-averse optimization. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1305–1313, 2007.

27 Chaitanya Swamy. Risk-averse stochastic optimization: probabilistically-constrained models and algorithms for black-box distributions. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1627–1646, 2011.

A Deferred Proofs of §2

▶ **Lemma 4.** *Let* $\boldsymbol{Y} = \{Y_1, \ldots, Y_m\}$ *be a set of independent Bernoulli r.v.s, where* Y_s *is* 1 *with probability* p_s*, and* 0 *otherwise. Let* $v_s \in \mathbb{R}_{\geq 0}$ *be a value associated with* Y_s*. WLOG assume* $v_s \geq v_{s+1}$ *for* $s \in [m-1]$*. Let* $b = \min\{a : \sum_{s=1}^{a} p_s \geq 1\}$*. Then*

$$\left(1 - \frac{1}{e}\right)\left(v_b + \sum_s p_s \cdot (v_s - v_b)^+\right) \leq \mathbb{E}_{\boldsymbol{Y}}\left[\max_s\{Y_s \cdot v_s\}\right] \leq v_b + \sum_s p_s \cdot (v_s - v_b)^+,$$

where $x^+ := \max\{x, 0\}$.

Proof. We begin by showing the lower bound on $\mathbb{E}_{A\sim\boldsymbol{Y}}\left[\max_{s\in A} v_s\right]$. Let $M := [b]$. Consider the new set of probabilities

$$p'_s = \begin{cases} 1 - \sum_{s<b} p_s & \text{if } s = b \\ p_s & \text{otherwise} \end{cases} \tag{17}$$

and let $\boldsymbol{Y'}$ be the corresponding Bernoulli r.v.s. Notice that $\sum_{s\in M} p'_s = 1$.

Since $p'_s \le p_s$, clearly we have that $\mathbb{E}_{A\sim\boldsymbol{Y}}\left[\max_{s\in A} v_s\right] \ge \mathbb{E}_{A\sim\boldsymbol{Y'}}\left[\max_{s\in A} v_s\right]$. Thus, we will focus on lower bounding $\mathbb{E}_{A\sim\boldsymbol{Y'}}\left[\max_{s\in A} v_s\right]$. The probability that no element of M is in A when drawn from $\boldsymbol{Y'}$ is

$$\prod_{s\in M}(1-p'_s) \quad \le \quad e^{-\sum_{s\in M} p'_s} \quad = \quad \frac{1}{e}$$

because $1 - x \le e^{-x}$ and $\sum_{s\in M} p'_s = 1$. It follows that

$$\begin{aligned}
\mathbb{E}_{A\sim\boldsymbol{Y}}\left[\max_{s\in A} v_s\right] &\ge \mathbb{E}_{A\sim\boldsymbol{Y'}}\left[\max_{s\in A} v_s\right] \\
&\ge \left(1-\frac{1}{e}\right)\mathbb{E}_{A\sim\boldsymbol{Y'}}\left[\max_{s\in A} v_s \mid \text{at least 1 element from } M \text{ in } A\right] \\
&\ge \left(1-\frac{1}{e}\right)\mathbb{E}_{A\sim\boldsymbol{Y'}}\left[\max_{s\in A} v_s \mid \text{exactly 1 element from } M \text{ in } A\right] \\
&= \left(1-\frac{1}{e}\right)\sum_{s\in M} v_s \frac{p'_s}{\sum_{i\in M} p'_s} \quad = \left(\frac{1-1/e}{1}\right)\sum_{s\in M} p'_s v_s
\end{aligned}$$

where the last line follows since $\sum_{s\in M} p'_s = 1$.

Thus, we have that

$$\begin{aligned}
\mathbb{E}_{A\sim\boldsymbol{Y}}\left[\max_{s\in A} v_s\right] &\ge \left(1-\frac{1}{e}\right)\sum_{s\in M} p'_s v_s \\
&= \left(1-\frac{1}{e}\right)\sum_{s\in M} p'_s\Big((v_s - v_b)^+ + v_b\Big) && \text{(by } v_s \ge v_b \text{ for } s \in M\text{)} \\
&\ge \left(1-\frac{1}{e}\right)\Big(v_b + \sum_{s\in M} p'_s\Big((v_s - v_b)^+\Big)\Big) && \left(\text{by } 1 = \sum_{s\in M} p'_s\right) \\
&\ge \left(1-\frac{1}{e}\right)\Big(v_b + \sum_{s\in M} p_s\Big((v_s - v_b)^+\Big)\Big) && \text{(by } (v_b - v_b)^+ = 0\text{)} \\
&= \left(1-\frac{1}{e}\right)\Big(v_b + \sum_{s} p_s\Big((v_s - v_b)^+\Big)\Big) && \text{(by } v_s > v_b \text{ iff } s \in M\text{)}
\end{aligned}$$

which gives our lower bound.

We now show the upper bound. Recall $x^+ := \max(x, 0)$. Notice that we have for any t,

$$\max(x, y) \le t + (x - t)^+ + (y - t)^+. \tag{18}$$

In particular, Eq. (18) follows because the RHS in each of the following cases is always $\ge \max\{x, y\}$.

- if $t \ge \max\{x, y\}$ we get t for the RHS.
- if $t \ge x$ and $t < y$ we get $t + y - t = y = \max\{x, y\}$ for the RHS; the symmetric case also holds.
- if $t < x$ and $t < y$ we get $t + x - t + y - t = x + y - t \ge \max\{x, y\}$ for the RHS.

It is easy to verify that this holds for a max of more than two inputs; i.e. for a set S of reals we have $\max(S) \le t + \sum_{s\in S}(s-t)^+$. Thus, we have

$$\mathbb{E}_{A\sim\boldsymbol{Y}}\Big[\max_{s\in A} v_s\Big] \le \mathbb{E}_{A\sim\boldsymbol{Y}}\Big[v_b + \sum_{s\in A}(v_s - v_b)^+\Big] \quad = \quad v_b + \mathbb{E}_{A\sim\boldsymbol{Y}}\Big[\sum_{s\in A}(v_s - v_b)^+\Big] \tag{19}$$

$$= v_b + \mathbb{E}_{A\sim\boldsymbol{Y}}\Big[\sum_{s\in A\cap M}(v_s - v_b)^+ + \sum_{s\in A\setminus M}(v_s - v_b)^+\Big] \tag{20}$$

$$= v_b + \mathbb{E}_{A\sim\boldsymbol{Y}}\Big[\sum_{s\in A\cap M}(v_s - v_b)^+\Big] \tag{21}$$

$$= v_b + \mathbb{E}_{A\sim\boldsymbol{Y}}\Big[\sum_{s\in A\cap M}(v_s - v_b)\Big] \tag{22}$$

$$= v_b + \sum_{s\in M} p_s\cdot(v_s - v_b) \tag{23}$$

$$= v_b + \sum_{s} p_s\cdot(v_s - v_b)^+, \tag{24}$$

where Eq.(19) is by Eq.(18), Eq.(21) is by $v_s > v_b$ iff $s \le b$, Eq.(22) is by $v_s \ge v_b$ for $s \in M$ and Eq.(24) is by $v_s > v_b$ iff $s \in M$. This is exactly the desired upper bound. ◀

▶ **Lemma 5.** *Let $(X_1, \boldsymbol{X_2})$ be a solution to a* TruncatedTwoStage *or* MinEMax *problem. We have*

$$B(X_1, \boldsymbol{X_2}) = \arg\min_B \Big[B + \sum_{s\in[m]} p_s\cdot(\mathsf{cost}(X_1, X_2^{(s)}) - B)^+\Big],$$

where the arg min *takes the largest B minimizing the relevant quantity.*

Proof. To clear our notation we let $\bar{B} := B(X_1, \boldsymbol{X_2})$, $c_s := \mathsf{cost}(X_1, X_2^{(s)})$ and $\bar{M} := M(X_1, \boldsymbol{X_2})$. Let $f(B) := B + \sum_{s\in[m]} p_s\cdot(c_s - B)^+$. We argue that $\bar{B}$ is the largest global minimum of f by showing that for any $\epsilon > 0$ we know that $f(\bar{B}) < f(\bar{B}+\epsilon)$ and $f(\bar{B}) \le f(\bar{B}-\epsilon)$.

We begin by noting that for any reals $a \le b$ we have

$$a^+ - b^+ \ge a - b \tag{25}$$

by casing on which of a and b are larger than 0.

Let $\hat{M} := \{s \in \bar{M} : c_s > \bar{B}\}$. Notice that $\sum_{s\in\hat{M}} p_s < 1$. For fixed and arbitrary $\epsilon > 0$ consider the relative values of $f(\bar{B})$ and $f(\bar{B}+\epsilon)$. We have

$$f(\bar{B}+\epsilon) - f(\bar{B}) = \epsilon + \sum_{s\in[m]} p_s\cdot\left((c_s - \bar{B} - \epsilon)^+ - (c_s - \bar{B})^+\right)$$

$$= \epsilon + \sum_{s\in\hat{M}} p_s\cdot\left((c_s - \bar{B} - \epsilon)^+ - (c_s - \bar{B})^+\right), \tag{26}$$

where Eq.(26) follows since for $s \notin \hat{M}$ we have $c_s \le \bar{B}$ and so $\left((c_s - \bar{B} - \epsilon)^+ - (c_s - \bar{B})^+\right) = 0$ for $s \notin \hat{M}$. Now noticing that for every s we have $(c_s - \bar{B} - \epsilon) \le (c_s - \bar{B})$, applying (25) to (26) gives

$$f(\bar{B}+\epsilon) - f(\bar{B}) \ge \epsilon + \sum_{s\in\hat{M}} p_s\cdot(-\epsilon) = \epsilon\left(1 - \sum_{s\in\hat{M}} p_s\right) > 0,$$

where the last inequality uses $\sum_{s\in\hat{M}} p_s < 1$. Thus, we have $f(\bar{B}+\epsilon) > f(\bar{B})$.

Now consider the relative values of $f(\bar{B})$ and $f(\bar{B}-\epsilon)$. We have

$$\begin{aligned} f(\bar{B}-\epsilon)-f(\bar{B}) &= -\epsilon+\sum_s p_s\cdot\left((c_s-\bar{B}+\epsilon)^+-(c_s-\bar{B})^+\right) \\ &\geq -\epsilon+\sum_{s\in\bar{M}} p_s\cdot\left((c_s-\bar{B}+\epsilon)^+-(c_s-\bar{B})^+\right) && (27)\\ &\geq -\epsilon+\sum_{s\in\bar{M}} p_s\cdot\left((c_s-\bar{B}+\epsilon)-(c_s-\bar{B})\right) && (28)\\ &\geq \epsilon\Big(1-\sum_{s\in\bar{M}} p_s\Big) \quad \geq \quad 0 && (29)\end{aligned}$$

where Eq.(27) is by $(c_s-\bar{B}+\epsilon)^+\geq(c_s-\bar{B})^+$, Eq.(28) is by $c_s\geq\bar{B}$ for $s\in\bar{M}$ and Eq.(29) is by $\sum_{s\in\bar{M}} p_s\geq 1$. Thus, for any $\epsilon>0$ we know that $f(\bar{B})<f(\bar{B}+\epsilon)$ and $f(\bar{B})\leq f(\bar{B}-\epsilon)$. It follows that, not only is $\bar{B}$ a global minimum of f but it is the largest global minimum. The lemma follows immediately. ◀

▶ **Lemma 6.** *For feasible solution $(X_1,\boldsymbol{X}_2)$ of any P_{EMax} we have,* $\mathsf{cost}_{EMax}(X_1,\boldsymbol{X}_2)\leq \mathsf{cost}_{Trunc}(X_1,\boldsymbol{X}_2)$.

Proof. We have

$$\begin{aligned} \mathsf{cost}_{\mathsf{EMax}}(X_1,\boldsymbol{X_2}) &= \mathbb{E}_A[\max_{s\in A}\{\mathsf{cost}(X_1,X_2^{(s)})\}] && (30)\\ &\leq B(X_1,\boldsymbol{X_2})+\sum_s p_s\cdot\Big(\mathsf{cost}(X_1,X_2^{(s)})-B(X_1,\boldsymbol{X_2})\Big)^+ && (31)\\ &= \mathsf{cost}_{\mathsf{Trunc}}(X_1,\boldsymbol{X}_2) && (32)\end{aligned}$$

where Equation (31) is by Lemma 4 and Equation (32) is by Lemma 5. ◀

▶ **Lemma 7.** *Let P_{EMax} be a* MinEMax *problem and P_{Trunc} be its truncated version. Let $(E_1,\boldsymbol{E_2})$ and $(T_1,\boldsymbol{T_2})$ be optimal solutions to P_{EMax} and P_{Trunc} respectively. We have that* $\mathsf{cost}_{Trunc}(T_1,\boldsymbol{T_2})\leq\left(\frac{1}{1-1/e}\right)\mathsf{cost}_{EMax}(E_1,\boldsymbol{E_2})$.

Proof. We have that

$$\begin{aligned} &\mathsf{cost}_{\mathsf{Trunc}}(T_1,\boldsymbol{T_2})\\ &\leq \mathsf{cost}_{\mathsf{Trunc}}(E_1,\boldsymbol{E_2}) && \text{(by } (T_1,\boldsymbol{T_2}) \text{ minimizes } \mathsf{cost}_{\mathsf{Trunc}})\\ &= \min_B\Big[B+\sum_s p_s\cdot(\mathsf{cost}(E_1,E_2^{(s)})-B)^+\Big]\\ &\leq B(E_1,\boldsymbol{E_2})+\sum_s p_s\cdot(\mathsf{cost}(E_1,E_2^{(s)})-B(E_1,\boldsymbol{E_2}))^+\\ &\leq \left(\frac{1}{1-1/e}\right)\mathbb{E}_A[\max_{s\in A}\{\mathsf{cost}(E_1,E_2^{(s)})\}] && \text{(by Lemma 4)}\\ &= \left(\frac{1}{1-1/e}\right)\mathsf{cost}_{\mathsf{EMax}}(E_1,\boldsymbol{E_2}). \end{aligned}$$ ◀

Streaming Hardness of Unique Games



Venkatesan Guruswami
Computer Science Department, Carnegie Mellon University,
5000 Forbes Ave, Pittsburgh, PA, USA, 15213
venkatg@cs.cmu.edu

Runzhou Tao[1]
Institute for Interdisciplinary Information Sciences,
Tsinghua University, Beijing, China 100084
trz15@mails.tsinghua.edu.cn

Abstract

We study the problem of approximating the value of a Unique Game instance in the streaming model. A simple count of the number of constraints divided by p, the alphabet size of the Unique Game, gives a trivial p-approximation that can be computed in $O(\log n)$ space. Meanwhile, with high probability, a sample of $\tilde{O}(n)$ constraints suffices to estimate the optimal value to $(1+\epsilon)$ accuracy. We prove that any single-pass streaming algorithm that achieves a $(p-\epsilon)$-approximation requires $\Omega_\epsilon(\sqrt{n})$ space. Our proof is via a reduction from lower bounds for a communication problem that is a p-ary variant of the Boolean Hidden Matching problem studied in the literature. Given the utility of Unique Games as a starting point for reduction to other optimization problems, our strong hardness for approximating Unique Games could lead to down*stream* hardness results for streaming approximability for other CSP-like problems.

2012 ACM Subject Classification Theory of computation → Communication complexity; Theory of computation → Streaming models

Keywords and phrases Communication complexity, CSP, Fourier Analysis, Lower bounds, Streaming algorithms, Unique Games

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.5

Category APPROX

Funding *Venkatesan Guruswami*: Research supported in part by NSF grants CCF-1422045 and CCF-1526092.
Runzhou Tao: Most of this work was done during a visit by the author to Carnegie Mellon University.

1 Introduction

The Unique Games (UG) problem is a type of constraint satisfaction problem on a graph. Given an alphabet $[p] = \{0, 1, \ldots, p-1\}$ and a graph $G = (V, E)$, we need to find a label assignment $x : V \to [p]$. The constraint on an edge $(u, v) \in E$ is specified described by a permutation $\pi_{uv} : [p] \to [p]$ and we want to find the assignment to maximize the number of equations $\pi_{uv}(x_u) = x_v$ that are satisfied. This maximum possible value over all possible assignments is called the optimal value of the UG instance. Simply picking a random assignments satisfies a fraction $1/p$ of the constraints in expectation, giving a trivial factor p approximation algorithm to the optimal value of any instance. More sophisticated algorithms based on semidefinite programming give better approximation guarantees [1], but even on almost-satisfiable instances where the optimal value is a $(1-\epsilon)$ fraction of the total number of constraints, the algorithm satisfies only a fraction $\approx p^{-\epsilon/2}$ of the constraints. Under Khot's celebrated Unique Games conjecture [11], this guarantee cannot be improved [12],

[1] Now affiliated with Columbia University, New York, USA.



© Venkatesan Guruswami and Runzhou Tao;
licensed under Creative Commons License CC-BY



and the conjecture further implies optimal hardness results for a host of problems. In terms of proven hardness results (under say the standard assumption that P $\neq$ NP), we know that Unique Games does not admit any constant factor approximations [3], and in an exciting recent line of work this was also established on instances that have optimum value close to a fraction 1/2 [2, 15].

To shed further light on the (difficulty of the) Unique Games problem from a different angle, in this work, we consider the Unique Games problem in the streaming model of computation. The constraints are assumed to arrive one-by-one in a single pass. The algorithm is only given a limited amount of memory, so cannot store the entire instance as it passes by. The goal of the algorithm is to estimate the optimal value of the Unique Games instance. That is, it must output a value T which is a lower bound on the optimum number of constraints that can be satisfied, and which is at most an approximation factor f from the optimum. In recent years, numerous algorithms and hardness for problems in the streaming model have been developed, and this work address the important Unique Games problem from the streaming perspective.

The simple-minded algorithm which simply counts the number of constraints and outputs a $1/p$ fraction of it as a valid estimate for every instance (by virtue of the random assignment algorithm), and delivers a factor p approximation. This algorithm can obviously be implemented in the streaming model using $O(\log n)$ space. Meanwhile, if we are given $\tilde{O}(n)$ space, we can sample a random $\tilde{O}(n)$-size subset of constraints and the answer of sampled unique game gives us an arbitrarily close approximation for the original stream.[2] A natural question which arises, and which motivates this work, is thus: *can we do better than the trivial factor p approximation in polylogarithmic space?*

In a beautiful work, Kapralov, Khanna, and Sudan [9] showed that the problem of Max-CUT, which is a special case of the Unique Games problem with alphabet size 2, does not admit an approximation better than the trivial factor 2 in $o(\sqrt{n})$ space in the streaming model where the edges arrive one-by-one. On the other hand, a recent work [7] showed that for the Max 2CSP problem (arbitrary Boolean arity two constraints) and Max-DICUT (the analog of Max-CUT on directed graphs), one can in fact beat the trivial factor 4 algorithm (that outputs 1/4'th the number of constraints, which is the expected value of a random assignment), and achieve a $\approx 5/2$-approximation using $O(\log n)$ space. The status of the streaming approximability of Unique Games over larger alphabet sizes was not addressed and remained open until our work.

1.1 Our Result

We show that for Unique Games with alphabet size p, a single-pass streaming algorithm requires at least $\tilde{\Omega}(\sqrt{n})$ space to have any chance of delivering a better estimate than the trivial factor p approximation. In particular, we cannot beat the trivial constraint-counting algorithm in the worst-case in polylogarithmic space.

▶ **Theorem 1.** *Let $p \geq 2$ be an integer and $\epsilon > 0$ be a small constant. Any streaming algorithm giving $(p-\epsilon)$-approximation for Unique Games with alphabet size p with success probability at least $9/10$ over its internal randomness must use $c_{p,\epsilon}\sqrt{n}$ space, for some positive constant $c_{p,\epsilon}$ that depends only on p, ϵ.*

[2] Note that We do not place any computational restriction on the algorithm, only on the amount of space it may use. Also, since we are talking about sub-linear space, we do not focus on finding an approximate solution, but only outputting an estimate of the optimal value. Since our focus in on lower bounds, this only makes our technical result stronger.

Furthermore, the hardness holds for distinguishing between satisfiable instances and those for which at most a fraction $(1/p+\epsilon)$ of the constraints can be satisfied by any assignment, and when the Unique Games constraints are linear (of the form $x_u+x_v = \alpha_{uv}$ over integers mod p).

1.2 Proof Structure

In our proof, we first introduce in Section 3, a communication problem called the p-ary Hidden Matching problem, which is a p-ary variant of the (Boolean) Hidden Matching problem proposed by Gavinsky et al.[4] and first used for streaming lower bounds by Verbin and Yu in [16]. The (distributional) p-ary Hidden Matching problem is a two-party one-way communication problem where Alice holds a random p-ary vector $x \in \mathbb{Z}_p^n$ and Bob holds a random matching of size $r = \alpha n$ (for some suitable $\alpha \in (0,1)$) and a vector $w \in \mathbb{Z}_p^r$. Alice must send one message to Bob, based on which he must distinguish between two distributions on the inputs. In both distributions x is uniformly random, and M is a random matching of the prescribed size. In the YES distribution, we set $w_e = x_u + x_v$ for each $e = (u, v)$ in the matching (i.e., $w = Mx$ where $M \in \{0,1\}^{\alpha n \times n}$ is the incidence matrix of the matching); in the NO distribution, w is uniformly random. We prove a communication lower bound of this problem using Fourier-analytic methods, which is similar to [8].

The vector w and the matching in the p-ary Hidden Matching problem can be seen as a description of some Unique Game constraints $x_u + x_v = w_e$. Of course each such instance individually is trivially always satisfiable. We can construct hard instances of Unique Game by combining together $O(1/\epsilon^2)$ independent copies of the random matching and corresponding w. In the YES case, we let w be according to the *same* (random) x, so that the constraints can be satisfied by x. In the NO case, the various choices of w are random and independent. This implies that every assignment $x \in \mathbb{Z}_p^n$ is close in performance to a random assignment, and thus satisfies only $\approx 1/p$ of the constraints, by concentration bounds.

We prove that a low-space streaming algorithm cannot distinguish between these distributions, which then implies Theorem 1. To prove this indistinguishability result, we give a reduction from the p-ary Hidden Matching problem. The proof is a classical hybrid argument since the streaming instance can be seen as a "multi-stage" version of the communication problem.

1.3 Differences from [9]

Our approach heavily borrows from the Max-CUT streaming lower bound from of Kapralov, Khanna, and Sudan [9]. Compared to their work, we only prove Theorem 1 for a *worst-case* arrival order of constraints, whereas the Max-CUT hardness result is shown even for a *random* arrival order for the edges. At each stage, instead a matching, Kapralov et. al. used a sub-critical random Erdös-Rényi graph with edge probability $\approx \alpha/n$. If the parameter α is sufficiently small, the graph obtained by putting together edges from all the stages is close in distribution to a random graph. As a result the arrival of edges in a random order does not help the streaming algorithm. For the analysis of each stage, they use a communication problem called the Boolean Hidden Partition problem that is variant of the Boolean Hidden Matching problem, since they have to work with Erdös-Rényi graphs rather than random matchings. This requires changes to some components in the proof outline of [4, 16].

Our communication problem still concerns matchings (rather than sub-critical Erdös-Rényi graphs), though we allow for (components of) x, w to take values from $\mathbb{Z}_p$ instead of Boolean values. By using Fourier analysis over the group $\mathbb{Z}_p$ instead of $\mathbb{Z}_2$, we are able to adapt the communication lower bound of [4].

It remains an interesting question to prove a streaming hardness for Unique Games similar to Theorem 1 for the case of random arrival order of constraints.

2 Preliminaries

Let $\mathbb{Z}_p = \{0, 1, \ldots, p-1\}$ be the ring with addition and multiplication modulo p. (We do not assume that p is a prime.) Fourier analysis over $\mathbb{Z}_p^n$ plays a key role in our proof. Consider the space of functions $\mathbb{Z}_p^n \to \mathbb{C}$. We define the inner product and 2-norm in it by

$$\langle f, g\rangle = \frac{1}{p^n}\sum_{x\in\mathbb{Z}_p^n} f(x)\overline{g(x)} \qquad \|f\|_2^2 = \langle f, f\rangle = \frac{1}{p^n}\sum_{x\in\mathbb{Z}_p^n} |f(x)|^2$$

The Fourier transform of f is a function $\hat{f} : \mathbb{Z}_p^n \to \mathbb{C}$ defined by

$$\hat{f}(z) = \langle f, \chi_z\rangle = \frac{1}{p^n}\sum_{x\in\mathbb{Z}_p^n} f(x)\overline{\omega^{z\cdot x}}$$

where $\chi_z : \mathbb{Z}_p^n \to \mathbb{C}$ is the character $\chi_z(x) = \omega^{z\cdot x}$ with "$\cdot$" being the scalar product and $\omega = e^{2\pi i/p}$ being the primitive p'th root of unity. For $z \in \mathbb{Z}_p^n$, we denote by $|z|$ the number of nonzero entries in z.

In our later proof, we use the following two lemmas concerning Parseval's identity and hypercontractivity.

▶ **Lemma 2** (Parseval). *For every function $f : \mathbb{Z}_p^n \to \mathbb{C}$, we have*

$$\|f\|_2^2 = \sum_{z\in\mathbb{Z}_p^n} |\hat{f}(z)|^2.$$

▶ **Lemma 3** (Hypercontractivity Theorem, [13]). *For function $f \in L_2(\mathbb{Z}_p^n)$, if $1 < q < 2$ and $0 \le \rho \le \sqrt{q-1}(1/p)^{1/q-1/2}$, we have*

$$\left\|T_\rho f\right\|_2 \le \|f\|_q$$

where T_ρ is the operator defined by $T_\rho f(x) = \sum_{z\in\mathbb{Z}_p^n} \hat{f}(z)\rho^{|z|}\chi_z(x)$.

Using the above theorem, we can derive an estimate on the sum of Fourier coefficients weighted by its support size.

▶ **Lemma 4.** *For a set $A \subseteq \mathbb{Z}_p^n$ and let f be its indicator function and let $|z|$ denote the number of non-zero coordinates of $z \in \mathbb{Z}_p^n$. Then for every $\delta \in [0, 1/p]$, we have*

$$\sum_{z\in\mathbb{Z}_p^n} \delta^{|z|}|\hat{f}(z)|^2 \le \left(\frac{|A|}{p^n}\right)^{2/(1+p\delta)}.$$

Proof. Let $\rho = \sqrt{q-1}(1/p)^{1/2} \le \sqrt{q-1}(1/p)^{1/q-1/2}$, then $q = 1 + p\rho^2$. By the hypercontractivity theorem, we know that

$$\left\|T_\rho f\right\|_2 \le \|f\|_{1+p\rho^2}$$

Meanwhile, we have $\left\|T_\rho f\right\|_2^2 = \sum_{z\in\mathbb{Z}_p^n} \rho^{2|z|}|\hat{f}(z)|^2$. Taking the square of the equation above and setting $\delta = \rho^2$ will get our desired result. ◀

3 p-ary Hidden Matching

In this section, we analyze a two-party (distributional) one-way communication problem, defined as follows.

p-ary Hidden Matching problem. Alice gets a random vector $x \in \mathbb{Z}_p^n$. Bob gets a random α-partial matching G (i.e., a matching of size αn on $\{1, 2, \ldots, n\}$) and a vector $w \in \mathbb{Z}_p^{\alpha n}$. Let $M \in \{0,1\}^{\alpha n \times n}$ be the incidence matrix of G, i.e., $M_{ev} = 1$ if v is an endpoint of e and 0 otherwise. There are two choices for the distribution of w, distinguishing which is the communication problem.

- In the **YES** distribution w is correlated with x as $w = Mx$ (arithmetic done in $\mathbb{Z}_p$);
- in the **NO** distribution, w is uniformly random in $\mathbb{Z}_p^n$ (and thus independent of x).

Alice must send a message to Bob, based on which Bob needs to distinguish distribution w belongs to. Formally, Bob must output Yes or No (based on Alice's message and his input w), and we say a protocol achieves advantage ϵ if the difference in probability of Bob outputting Yes differs under the Yes and No distributions by at least ϵ. The following shows that Alice needs to send at least $\Omega(\sqrt{n})$ bits for Bob to achieve constant advantage.

▶ **Theorem 5.** *For $\alpha \in (0, 1/4]$, any protocol that achieves advantage $\epsilon > 0$ for the p-ary Hidden Matching problem requires at least $\Omega(\epsilon\sqrt{n})$ bits of communication from Alice to Bob.*

The proof of the above lemma is the main result of this section. Our proof closely follows the structure of [4], from which the main difference is that our proof has to work for the p-ary case.

Before we embark on the proof, we need some more lemmas. We begin with an application of hypercontractivity to bound the Fourier mass at any level.

▶ **Lemma 6.** *For a set $A \subseteq \mathbb{Z}_p^n$ with size at least $p^n/2^c$ and let f be its indicator function and let $|z|$ denote the number of non-zero coordinates of $z \in \mathbb{Z}_p^n$. Then for every $k \leq 4c$ we have*

$$\frac{p^{2n}}{|A|^2} \sum_{|z|=k} |\hat{f}(z)|^2 \leq \left(\frac{4\sqrt{2}pc}{k}\right)^k .$$

Proof. By Lemma 4, given some constant $0 \leq \delta \leq 1/p$, we have

$$\begin{aligned}
\frac{p^{2n}}{|A|^2} \sum_{|z|=k} |\hat{f}(z)|^2 &\leq \frac{p^{2n}}{|A|^2} \frac{1}{\delta^k} \sum_{z \in \mathbb{Z}_p^n} \delta^{|z|} |\hat{f}(z)|^2 \\
&\leq \frac{p^{2n}}{|A|^2} \frac{1}{\delta^k} \left(\frac{|A|}{p^n}\right)^{2/(1+p\delta)} \\
&= \frac{1}{\delta^k} \left(\frac{p^n}{|A|}\right)^{2p\delta/(1+p\delta)} \\
&\leq \frac{1}{\delta^k} \left(\frac{p^n}{|A|}\right)^{2p\delta} .
\end{aligned}$$

Choosing $\delta = k/4cp$ will give our desired result. ◀

We also need a combinatorial lemma about counting of some matchings.

▶ **Lemma 7.** *Let G be a uniformly random α-partial matching and M be its incidence matrix. If $x \in \mathbb{Z}_p^n$ has $|x| = k$ for some even*[3] *k, then*

$$\Pr_G[\exists z \in \mathbb{Z}_p^{\alpha n} \text{s.t.} M^T z = x] \leq \binom{\alpha n}{k/2} \Big/ \binom{n}{k} .$$

Proof. We know that the total number of all α-partial matchings of n vertices is $n!/(2^{\alpha n}(\alpha n)!(n-2\alpha n)!)$. And if there exists some z such that $M^T z = x$, then G must have exactly $k/2$ edges between those vertices v with $x_v \neq 0$. There are $k!/(2^{k/2}(k/2)!)$ number of ways to choose those edges. Also, we need to choose $\alpha n - k/2$ edges amongst those v whose $x_v = 0$, which we have $(n-k)!/(2^{n-k}(\alpha n - k/2)!(n-2\alpha n)!)$ ways to do. Combining them together leads to the lemma. ◀

From the lemmas above, we can derive an important result in our proof.

▶ **Lemma 8.** *Let $A \subseteq \mathbb{Z}_p^n$ be of size at least $p^n/2^c$ for some $c \geq 1$, G be a uniformly random α-partial matching for some $0 < \alpha \leq 1/4$ and M be its incidence matrix. There exists a constant γ independent of n, c and α, such that for all $\epsilon > 0$, if $c \leq \gamma\epsilon\sqrt{n/\alpha}$ then*

$$E_M[\|p_M - U\|_{tvd}] \leq \epsilon,$$

*where $p_M(w) = |\{x \in A \mid Mx = w\}|/|A|$ is the distribution of w in the **YES** case when x is uniformly random in A.*

Proof. To show that $E_M[\|p_M - U\|_{tvd}] \leq \epsilon$, we can start by bounding the Fourier coefficients of p_M. In fact they are closely related to $\hat{f}$ (where recall that f is the indicator function for membership in the set A):

$$\begin{aligned}
\widehat{p_M}(z) &= \frac{1}{p^{\alpha n}} \sum_{w \in \mathbb{Z}_p^{\alpha n}} p_M(z)\omega^{-w\cdot z} \\
&= \frac{1}{|A|p^{\alpha n}} \sum_{k=0}^{p-1} \omega^{-k} |\{x \in A | (Mx)\cdot z = k\}| \\
&= \frac{1}{|A|p^{\alpha n}} \sum_{k=0}^{p-1} \omega^{-k} |\{x \in A | x \cdot (M^T z) = k\}| \\
&= \frac{1}{|A|p^{\alpha n}} \sum_{x \in A} \omega^{-x\cdot(M^T z)} \\
&= \frac{p^n}{|A|p^{\alpha n}} \widehat{f}(M^T z)
\end{aligned}$$

From the bound of Fourier coefficients, we can give a bound on squared total variation distance

$$\begin{aligned}
E_M[\|p_M - U\|_{tvd}^2] &\leq p^{2\alpha n} E_M[\|p_M - U\|_2^2] \\
&= p^{2\alpha n} E_M \left[\sum_{z \in \mathbb{Z}_p^{\alpha n} \setminus \{0^{\alpha n}\}} |\widehat{p_M}(z)|^2 \right] \\
&= \frac{p^{2n}}{|A|^2} E_M \left[\sum_{z \in \mathbb{Z}_p^{\alpha n} \setminus \{0^{\alpha n}\}} |\widehat{f}(M^T z)|^2 \right]
\end{aligned}$$

[3] We note that if $|x|$ is odd, then there can be no z such that $M^T z = x$.

by Cauchy-Schwarz inequality, Parseval equality and the bound above. Since there is at most one $z \in \mathbb{Z}_p^{\alpha n}$ such that $x = M^T z$ for given x, we have

$$\begin{aligned}
&= \frac{p^{2n}}{|A|^2} E_M \left[\sum_{x \in \mathbb{Z}_p^n \setminus \{0^n\}} |\widehat{f}(x)|^2 |\{z \in \mathbb{Z}_p^{\alpha n} | x = M^T z\}| \right] \\
&= \frac{p^{2n}}{|A|^2} \sum_{x \in \mathbb{Z}_p^n \setminus \{0^n\}} \Pr_M[\exists z \in \mathbb{Z}_p^{\alpha n} \text{s.t. } M^T z = x] |\hat{f}(x)|^2 \\
&\leq \frac{p^{2n}}{|A|^2} \sum_{k=2, k\text{even}}^{2\alpha n} \frac{\binom{\alpha n}{k/2}}{\binom{n}{k}} \sum_{|x|=k} |\hat{f}(x)|^2.
\end{aligned}$$

We then split the sum into two parts $k < 4c$ and $k \geq 4c$. For $k < 4c$, using $(n/k)^k \leq \binom{n}{k} \leq (en/k)^k$, we have

$$\begin{aligned}
\frac{p^{2n}}{|A|^2} \sum_{k=2, k\text{even}}^{4c-2} \frac{\binom{\alpha n}{k/2}}{\binom{n}{k}} \sum_{|x|=k} |\hat{f}(x)|^2 &\leq \sum_{k=2, k\text{even}}^{4c-2} \frac{(2e\alpha n/k)^{k/2}}{(n/k)^k} \left(\frac{4\sqrt{2}pc}{k} \right)^k \quad \text{(using Lemma 6)} \\
&\leq \sum_{k=2, k\text{even}}^{4c-2} \left(\frac{64 e \gamma^2 \epsilon^2 p^2}{k} \right)^{k/2},
\end{aligned}$$

which is at most $\epsilon^2/2$ when γ is sufficiently small. For $k \geq 4c$ note that $\sum_x |\hat{f}(x)|^2 = |A|/p^n$ by Parseval and $\binom{\alpha n}{k/2}/\binom{n}{k}$ is decreasing for even $k \leq 2\alpha n$, we have

$$\begin{aligned}
\frac{p^{2n}}{|A|^2} \sum_{k=4c, k\text{even}}^{2\alpha n} \frac{\binom{\alpha n}{k/2}}{\binom{n}{k}} \sum_{|x|=k} |\hat{f}(x)|^2 &\leq 2^c \frac{\binom{\alpha n}{2c}}{\binom{n}{4c}} \\
&\leq 2^c \left(\frac{8c\alpha e}{n} \right)^{2c} \\
&\leq \left(8\sqrt{2} e \gamma \epsilon \sqrt{\alpha/n} \right)^{2c} \leq \epsilon^2/2.
\end{aligned}$$

The last inequality holds because $n \geq 1$ and $c \geq 1$, and we let γ be a sufficiently small constant. Thus, in total we have $E_M[\|p_M - U\|_{tvd}^2] \leq \epsilon^2$, which means by Jensen $E_M[\|p_M - U\|_{tvd}] \leq \epsilon$. ◀

From the lemma above, we can prove the communication lower bound of p-ary Hidden Matching problem.

Proof of Theorem 5. By fixing the randomness of the protocol, we can assume without loss of generality that the protocol is deterministic . Fix $\epsilon > 0$ to a small constant and let $c = \gamma\epsilon\sqrt{n/\alpha}$. Consider any protocol that communicates at most $C = c - \log(1/\epsilon)$ bits. In the protocol, Alice's message gives an partition of $\mathbb{Z}_p^n$ into 2^C subsets. We call the sets with size $\epsilon p^n / 2^C = p^n/2^c$ be "large sets", then for a uniformly random $x \in \mathbb{Z}_p^n$, with probability $1 - \epsilon$, x belongs to a large set. When x is in a large set, by Lemma 8, Bob can get an advantage of at most ϵ. Together with the advantage from small sets, the overall advantage Bob can get is at most $O(\epsilon)$, which completes the proof. ◀

4 Reduction to Streaming Algorithm for Unique Games

In this section, we will prove Theorem 1. Towards this end, we will describe a pair of distributions, $\mathbf{Y}$ and $\mathbf{N}$, where $\mathbf{Y}$ is supported on satisfiable instances of Unique Games, and $\mathbf{N}$ is supported with high probability on instances where at most $\approx 1/p$ fraction of constraints can be satisfied. We will then establish, via reduction from the p-ary Hidden Matching communication problem, that any low-space streaming algorithm cannot distinguish between these distributions, thus establishing Theorem 1.

4.1 Input distributions

We construct the above-mentioned distributions in a "multi-stage" way (using k stages) based on the **YES** and **NO** distributions (defined at the beginning of Section 3) for p-ary Hidden Matching. First we independently sample k α-partial matchings on n vertices a The Unique Games instance graph G will be the union of these matchings. It will thus have n vertices and $k\alpha n$ edges (we allow multiple edges should they be sampled). We next specify the Unique Games constraints, which will be two-variable linear equations, one for each edge.

- In the $\mathbf{Y}$ distribution, we sample a random $z \in \mathbb{Z}_p^n$ uniformly. We let the constraint on edge (u, v) of G be $x_u + x_v = z_u + z_v$.
- In the $\mathbf{N}$ distribution, for each edge (u, v) of G, we let the constraint be $x_u + x_v = q$ for a random $q \in \mathbb{Z}_p$, independently chosen for each edge.

For instances sampled in the $\mathbf{Y}$ distribution, the best solution is obviously $x_u = z_u$ for all $u \in [n]$, which satisfies all the constraints. For the $\mathbf{N}$ distribution, we can use Chernoff bounds to upper bound the value of the optimal solution.

▶ **Lemma 9.** *Let $0 < \epsilon < 1$. If $k = Cp \log p/(\alpha\epsilon^2)$ for some large constant $C > 0$, then for a Unique Games instance sampled from the $\mathbf{N}$ distribution, the optimal fraction of satisfiable constraints is at most $(1 + \epsilon)/p$ with high probability.*

Before we proceed to the proof, we first state the Chernoff bound for negatively correlated random variables.

▶ **Lemma 10** ([14])**.** *Let $X_1, \ldots, X_n$ be negatively correlated Bernoulli random variables and $X = X_1 + \cdots + X_n$. Then we have*

$$\Pr[X \geq (1 + \epsilon)E[X]] \leq \exp(-E[X]\epsilon^2/3).$$

Proof of Lemma 9. Fix an assignment $x \in \mathbb{Z}_p^n$. For $1 \leq \ell \leq k$, let $X_{ij}^{(\ell)}$ be the indicator of the following event: "in the ℓ-th stage, the edge (i, j) is included in the α-partial matching and is satisfied by the assignment x." Then, $S = \sum_{\ell,i,j} X_{ij}^{(\ell)}$, summed over $1 \leq \ell k$, and $1 \leq i < j \leq n$, is the random variable counting the number of constraints satisfied by the assignment x. Note that $\mathbb{E}[S] = k\alpha n/p$ is the expected number of constraints by the assignment x. And we know that each $X_{ij}^{(\ell)}$ is a Bernoulli random variable with probability of equaling 1 being $2\alpha n/(pn(n-1))$.

We first claim that these random variables are negatively correlated. In fact, edges in different stages are independent. For edges in the same stage ℓ, consider that we know that random variables $X_{i_1 j_1}^{(\ell)}, X_{i_2 j_2}^{(\ell)}, \ldots, X_{i_t j_t}^{(\ell)}$ have value 1, and a vertex pair (i_0, j_0). If i_0 or j_0 is occurred in some i_s or j_s, then $X_{i_0 j_0}^{(\ell)}$ must be 0. Otherwise the conditional expectation of $X_{i_0 j_0}^{(\ell)}$ is $2(\alpha n - t)/(p(n-t)(n-1-t))$, which is less than the unconditional expectation of $2\alpha n/(pn(n-1))$. In all cases we have $E[X_{i_0 j_0}^{(\ell)} \mid X_{i_1 j_1}^{(\ell)} = X_{i_2 j_2}^{(\ell)} = \cdots = X_{i_t j_t}^{(\ell)} = 1] \leq E[X_{i_0 j_0}^{(\ell)}]$, which in turn means negative correlation.

Thus, by Chernoff bound for negatively random variables, we know that

$$\Pr[S \geq (1+\epsilon)k\alpha n/p] \leq \exp(-\epsilon^2 k\alpha n/3p) = p^{-Cn/3} \leq p^{-2n}.$$

The proof is now complete by a union bound over all p^n candidate assignments. ◀

4.2 Reduction from p-ary Hidden Matching

Note that each stage of constraints in the Unique Games instance corresponds to the p-ary Hidden Matching problem, with the $\mathbf{Y}$ distribution (resp. $\mathbf{N}$ distribution) coinciding with the YES distribution (NO distribution) of the Hidden Matching problem. Using this, we can link the hardness of the two problems via a hybrid argument. Recall that we say that a decision algorithm distinguishes between two distributions D_1 and D_2 with advantage η if it accepts samples from one distribution with probability at least η more than those from the other distribution.

▶ **Lemma 11.** *Suppose there exists a streaming algorithm **ALG** using c bits of memory that can achieve advantage $1/4$ in distinguishing between instances from the **Y** and **N** distributions of Unique Games instances. Then there exists a protocol with c bits of communication for the p-ary Hidden matching problem with advantage $\Omega(1/k)$ in distinguishing between **YES** and **NO** distributions.*

We now prepare for the proof of Lemma 11. Our proof follows along the lines of a similar argument in [9]. In the execution of **ALG** on instances from the $\mathbf{Y}$ and $\mathbf{N}$ distributions, let the memory after receiving the i-th stage constraints be S_i^Y and S_i^N respectively. Thus S_i^Y, S_i^N are random variables in $\{0,1\}^c$. Without loss of generality, we assume that $S_0^Y = S_0^N = 0$.

We now define the notion of an informative index, as in [9].

▶ **Definition 12** (Informative index). *An index $j \in \{0,\ldots,k-1\}$ is said to be δ-informative for $\delta > 0$ if*

$$\left\|S_{j+1}^Y - S_{j+1}^N\right\|_{tvd} \geq \left\|S_j^Y - S_j^N\right\|_{tvd} + \delta$$

We now show the existence of a $\Omega(1/k)$-informative index for any streaming algorithm that distinguishes between $\mathbf{Y}$ and $\mathbf{N}$ distributions.

▶ **Lemma 13.** *Suppose a streaming algorithm **ALG** uses c bits of memory and distinguishes the **Y** and **N** distributions with advantage $1/4$. Then the algorithm has a $\Omega(1/k)$-informative index.*

Proof. At first, $\left\|S_0^Y - S_0^N\right\|_{tvd} = 0$; at the end of the algorithm, since advantage is at least $1/4$, $\left\|S_k^Y - S_k^N\right\|_{tvd}$ must be at least some constant C. Let j be the first index such that $\left\|S_{j+1}^Y - S_{j+1}^N\right\|_{tvd} \geq C(j+1)/k$, then j is a C/k-informative index. ◀

Let j^* be a $\Omega(1/k)$-informative index of a streaming algorithm **ALG**. Using **ALG**, we can devise a communication protocol for the p-ary Hidden Matching problem as follows.

1. Suppose Alice holds as input a random string $x \in \mathbb{Z}_p^n$. She samples j^* random α-partial matchings and feeds the streaming algorithm UG constraints for the first j^* stages that follow the $\mathbf{Y}$ distribution with the setting $z = x$.
2. Alice sends the memory contents of **ALG** after j^* stages to Bob.

3. Bob samples an α-partial matching and gives constraints $x_u + x_v = w_e$ for $e = (u, v)$ according to his w. He then continues running **ALG** on these constraints as the (j^*+1)'th stage.

   Let the memory Bob gets be s.
4. Let the resulting memory distribution under the two cases (depending on w's distribution) be $\tilde{S}^{\text{YES}}$ and $\tilde{S}^{\text{NO}}$. (Note that these distribution can be computed by Bob since **ALG** is known.)

   Bob outputs 1 if $\Pr[\tilde{S}^{\text{YES}} = s] \geq \Pr[\tilde{S}^{\text{NO}} = s]$, and otherwise 0.

The above completes the description of the reduction. Before we analyze it and proceed to the proof of Lemma 11, we need the following fact about the statistical (total variation) distance between random variabls.

▶ **Lemma 14** (Claim 6.5, [9]). *Let X, Y be two random variables and W be independent of (X, Y). Then for any function f, we have*

$$\left\|f(X,W) - f(Y,W)\right\|_{tvd} \leq \|X - Y\|_{tvd} \ .$$

Proof of Lemma 11. We argue that the above protocol for p-ary Hidden Matching achieves the claimed advantage of $\Omega(1/k)$ in distinguishing between **YES** and **NO** distributions.

Let f be the function that maps the memory after stage j^* and constraints of stage (j^*+1) to the memory after stage (j^*+1). Thus we have $\tilde{S}^{\text{YES}} = S^Y_{j^*+1} = f(S^Y_j, C^Y)$ and $\tilde{S}^{\text{NO}} = f(S^Y_j, C^N)$, where C^Y, C^N be the constraints Bob generated in both cases. We also know that $S^N_{j^*+1} = f(S^N_j, C^N)$.

By Lemma 14, we know that

$$\left\|\tilde{S}^{\text{NO}} - S^N_{j^*+1}\right\|_{tvd} = \left\|f(S^Y_{j^*}, C^N) - f(S^N_{j^*}, C^N)\right\|_{tvd} \leq \left\|S^Y_{j^*} - S^N_{j^*}\right\|_{tvd}.$$

Hence, we have

$$\begin{aligned}
\left\|\tilde{S}^{\text{YES}} - \tilde{S}^{\text{NO}}\right\|_{tvd} &\geq \left\|S^Y_{j^*+1} - S^N_{j^*+1}\right\|_{tvd} - \left\|\tilde{S}^{\text{NO}} - S^N_{j^*+1}\right\|_{tvd} \\
&\geq \left\|S^Y_{j^*+1} - S^N_{j^*+1}\right\|_{tvd} - \left\|S^Y_{j^*} - S^N_{j^*}\right\|_{tvd} \\
&\geq \Omega(1/k).
\end{aligned}$$

The strategy in Step 4 that Bob uses distinguishes between $\tilde{S}^{\text{YES}}$ and $\tilde{S}^{\text{NO}}$ with advantage exactly $\left\|\tilde{S}^{\text{YES}} - \tilde{S}^{\text{NO}}\right\|_{tvd}$, which is at least $\Omega(1/k)$. This concludes the proof of Lemma 11. ◀

Our main result, Theorem 1, now follows by choosing $\alpha = 1/8$ and $k = \lceil Cp \log p/\epsilon^2 \rceil$ for a large enough absolute constant C, and combining together Theorem 5, Lemma 11, and Lemma 9.

5 Conclusion

We proved that Unique Games is hard for single-pass streaming algorithms in a strong sense: even if the instance is perfectly satisfiable, the algorithm cannot certify that it is even $(1/p + \epsilon)$-satisfiable, where p is the alphabet size, and $\epsilon > 0$ is an arbitrary constant. Some natural directions to extend our lower bound would be to multi-pass algorithms, and for random arrival order of the constraints.

An interesting direction for future work would be to establish limitations of streaming algorithms for other approximation problems which are only known to be "Unique Games-hard." An example, which partly motivated this work initially, is the Maximum Acyclic Subgraph (MAS) problem. The MAS problem is another one of those notorious problems for which there is a trivial algorithm that achieves approximation ratio of 2 (the algorithm is simply to order the vertices arbitrarily, and take either all the forward-going or backward-going edges as an acyclic subgraph with at least 1/2 the edges), and no efficient algorithm achieving a factor $(2-\epsilon)$-approximation is known for any fixed $\epsilon > 0$. On the other hand, known NP-hardness results are rather weak, but under the Unique Games conjecture, it is known that there is no efficient $(2-\epsilon)$-approximation for MAS [6, 5].

One can try to explain the difficulty of MAS in the streaming model, by proving a result similar in spirit to the result we established for Unique Games. Specifically, given as input a directed graph whose edges arrive one-by-one, can a low-space single-pass streaming algorithm distinguish between the cases when the directed graph is acyclic and when it has no acyclic subgraph with even $1/2+\epsilon$ of the edges? (The 1/2 threshold being trivial, since any directed graph has an acyclic subgraph with 1/2 the edges.) A result of this flavor was shown with 1/2 replaced by 7/8 in [7].

The reduction from Unique Games to $(2-\epsilon)$-approximating MAS [6] and our inapproximability result for UG in the streaming model gives hope to prove the desired streaming hardness for MAS as well, by implementing the reduction in a streaming manner. Since reductions involving CSPs are usually local, the arrival of one constraint of problem $\mathbb{A}$ can be mimicked by the arrival of the constraints of problem $\mathbb{B}$ that implement it. The reduction from UG to MAS (and indeed many other CSPs), however, introduces constraints between all pairs of variables that share a constraint with a UG vertex u. So to implement it one would need the UG streaming hardness under a "vertex arrival" model, where the graph is bipartite, and all constraints involving a left hand side vertex arrive in sequence. We *can* adapt the reduction in [6] to something local, based only on a single constraint, thereby making it more friendly to the edge arrival model. However, this only yields a weaker hardness result that distinguishing DAGs from graphs whose MAS has at most $\approx 3/4$ edges requires $\Omega(\sqrt{n})$ space.

Obtaining a tight streaming hardness result for MAS, and more broadly leveraging our tight streaming hardness result for Unique Games toward streaming inapproximability results for other optimization problems for which we have optimal reductions from Unique Games, are interesting directions for future work. Further, given the hardness results in this work and [9], one can ask which CSPs and related problems admit non-trivial approximate estimation algorithms in the streaming model. Even though one might suspect that strong hardness results should be pervasive, it seems that it is rather non-trivial to establish strong limitations of streaming algorithms, and the algorithms for Max 2CSP in [7] suggest that there might be more interesting cases where streaming algorithms can provide non-trivial guarantees.

In recent work [10], Kapralov and Krachun give an $\tilde{\Omega}(n)$ space lower bound on beating a 2-approximation for MAX-CUT by a single-pass streaming algorithm. A generalization of their techniques to the p-ary case may lead to a near-tight streaming space lower bound for Unique Games.

References

1 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Near-optimal algorithms for unique games. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 205–214, 2006.

2 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing*, pages 376–389, 2018.

3 Uriel Feige and Daniel Reichman. On Systems of Linear Equations with Two Variables per Equation. In *Approximation, Randomization, and Combinatorial Optimization, Algorithms and Techniques (APPROX, RANDOM)*, pages 117–127, 2004.

4 Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald De Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In *Proceedings of the 39th annual ACM symposium on Theory of computing*, pages 516–525. ACM, 2007.

5 Venkatesan Guruswami, Johan Håstad, Rajsekar Manokaran, Prasad Raghavendra, and Moses Charikar. Beating the Random Ordering Is Hard: Every Ordering CSP Is Approximation Resistant. *SIAM Journal on Computing*, 40(3):878–914, 2011.

6 Venkatesan Guruswami, Rajsekar Manokaran, and Prasad Raghavendra. Beating the Random Ordering is Hard: Inapproximability of Maximum Acyclic Subgraph. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 573–582, 2008.

7 Venkatesan Guruswami, Ameya Velingker, and Santhoshini Velusamy. Streaming complexity of approximating Max 2CSP and Max Acyclic Subgraph. In *20th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 8:1–8:19, 2017.

8 Jeff Kahn, Gil Kalai, and Nathan Linial. The influence of variables on Boolean functions. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, pages 68–80, 1988.

9 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Streaming lower bounds for approximating MAX-CUT. In *Proceedings of the 26 annual ACM-SIAM symposium on Discrete Algorithms*, pages 1263–1282. Society for Industrial and Applied Mathematics, 2015.

10 Michael Kapralov and Dmitry Krachun. An Optimal Space Lower Bound for Approximating MAX-CUT. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, pages 277–288, 2019.

11 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 767–775, 2002.

12 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal Inapproximability Results for MAX-CUT and Other 2-Variable CSPs? *SIAM Journal on Computing*, 37(1):319–357, 2007.

13 Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

14 Alessandro Panconesi and Aravind Srinivasan. Randomized Distributed Edge Coloring via an Extension of the Chernoff–Hoeffding Bounds. *SIAM Journal on Computing*, 26(2):350–368, 1997.

15 Khot Subhash, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 592–601. IEEE, 2018.

16 Elad Verbin and Wei Yu. The streaming complexity of cycle counting, sorting by reversals, and other problems. In *Proceedings of the 26 annual ACM-SIAM symposium on Discrete Algorithms*, pages 11–25, 2011.

On Strong Diameter Padded Decompositions

Arnold Filtser
Ben Gurion University of the Negev, Beersheva, Israel
arnold273@gmail.com

Abstract

Given a weighted graph $G = (V, E, w)$, a partition of V is Δ-bounded if the diameter of each cluster is bounded by Δ. A distribution over Δ-bounded partitions is a β-padded decomposition if every ball of radius $\gamma\Delta$ is contained in a single cluster with probability at least $e^{-\beta\cdot\gamma}$. The weak diameter of a cluster C is measured w.r.t. distances in G, while the strong diameter is measured w.r.t. distances in the induced graph $G[C]$. The decomposition is weak/strong according to the diameter guarantee.

Formerly, it was proven that K_r free graphs admit weak decompositions with padding parameter $O(r)$, while for strong decompositions only $O(r^2)$ padding parameter was known. Furthermore, for the case of a graph G, for which the induced shortest path metric d_G has doubling dimension ddim, a weak $O(\text{ddim})$-padded decomposition was constructed, which is also known to be tight. For the case of strong diameter, nothing was known.

We construct strong $O(r)$-padded decompositions for K_r free graphs, matching the state of the art for weak decompositions. Similarly, for graphs with doubling dimension ddim we construct a strong $O(\text{ddim})$-padded decomposition, which is also tight. We use this decomposition to construct $\big(O(\text{ddim}), \tilde{O}(\text{ddim})\big)$-sparse cover scheme for such graphs. Our new decompositions and cover have implications to approximating unique games, the construction of light and sparse spanners, and for path reporting distance oracles.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Approximation algorithms analysis; Theory of computation → Random projections and metric embeddings

Keywords and phrases Padded decomposition, Strong Diameter, Sparse Cover, Doubling Dimension, Minor free graphs, Unique Games, Spanners, Distance Oracles

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.6

Category APPROX

Related Version `https://arxiv.org/abs/1906.09783`

Funding *Arnold Filtser*: Supported in part by ISF grant No. (1817/17) and by BSF grant No. 2015813.

Acknowledgements The author would like to thank Ofer Neiman for helpful discussions.

1 Introduction

Divide and conquer is a widely used algorithmic approach. In many distance related graph problems, it is often useful to randomly partition the vertices into clusters, such that small neighborhoods have high probability to be clustered together. Given a weighed graph $G = (V, E, w)$, a partitions is Δ-*bounded* if the diameter of every cluster is at most Δ. A distribution $\mathcal{D}$ over partitions is called a (β, δ, Δ)-*padded decomposition*, if every partition is Δ-bounded, and for every vertex $v \in V$ and $\gamma \in [0, \delta]$, the probability that the entire ball $B_G(v, \gamma\Delta)$ of radius $\gamma\Delta$ around v is clustered together, is at least $e^{-\beta\gamma}$. If G admits a (β, δ, Δ)-padded decomposition for every $\Delta > 0$, we say that G is (β, δ)-decomposable. If in addition $\delta = \Omega(1)$ is a universal constant, we say that G is β-decomposable. Among other


© Arnold Filtser;
licensed under Creative Commons License CC-BY



applications, padded decompositions have been used for multi-commodity flow [33], metric embeddings [41, 40, 34], edge and vertex cut problems [37, 24], routing [4], near linear SDD solvers [12], approximation algorithms [16], and many more.

The *weak* diameter of a cluster $C \subseteq V$ is the maximal distance between a pair of vertices in the cluster w.r.t. the shortest path metric in the entire graph G, i.e. $\max_{u,v \in C} d_G(u,v)$. The *strong* diameter is the maximal distance w.r.t. the shortest path metric in the induced graph $G[C]$, i.e. $\max_{u,v \in C} d_{G[C]}(u,v)$. Padded decomposition can be weak/strong according to the provided guarantee on the diameter of each cluster. It is considerably harder to construct padded decompositions with strong diameter. Nevertheless, strong diameter is more convenient to use, and some applications indeed require that (e.g. for routing, spanners etc.).

Previous results on padded decompositions are presented in Table 1. General n-vertex graphs are strongly $O(\log n)$-decomposable [10], which is also tight. In a seminal work, given a K_r free graph G, Klein, Plotkin and Rao [33] showed that G is weakly $O(r^3)$-decomposable. Fakcharoenphol and Talwar [23] improved the decomposability of K_r free graph to $O(r^2)$ (weak diameter). Finally, Abraham et al. [5] improved the decomposition parameter to $O(r)$, still with weak diameter. The first result on strong diameter for K_r free graphs is by Abraham et al. [6], who constructed decompositions with padding parameter exponential in r. In fact, they study a somewhat weaker notion of decomposition called separating decompositions (see Definition 16). Afterwards, in the same paper providing the state of the art for weak diameter, Abraham et al. [5] proved that K_r free graphs are strongly $(O(r^2), \Omega(\frac{1}{r^2}))$-decomposable. It was conjectured [5] that K_r free graphs are $O(\log r)$-decomposable. However, even improving strong diameter decompositions to match the state of the art of weak diameter remained elusive.

Another family of interest are graph with bounded doubling dimension[1]. Abraham, Bartal and Neiman [2] showed that a graph with doubling dimension ddim is weakly $O(\text{ddim})$-decomposable, generalizing a result from [29]. No prior strong diameter decomposition for this family is known.

A related notion to padded decompositions is *sparse cover*. A collection $\mathcal{C}$ of clusters is a (β, s, Δ)-sparse cover if it is strongly Δ-bounded, each ball of radius $\frac{\Delta}{\beta}$ is contained in some cluster, and each vertex belongs to at most s different clusters. A graph admits (β, s)-sparse cover scheme if it admits (β, s, Δ)-sparse cover for every $\Delta > 0$. Awerbuch and Peleg [9] showed that for $k \in \mathbb{N}$, general n-vertex graphs admit a strong $(2k-1, 2k \cdot n^{\frac{1}{k}})$-sparse cover scheme. For K_r free graphs, Abraham et al. [6] constructed $(O(r^2), 2^r(r+1)!)$-sparse cover scheme. Busch, LaFortune and Tirthapura [15] constructed $(4, f(r) \cdot \log n)$-sparse cover scheme for K_r free graphs[2] .

For the case of graphs with doubling dimension ddim, Abraham et al. [4] constructed a $(2, 4^{\text{ddim}})$-sparse cover scheme. No other tradeoff are known. In particular, if ddim is larger than $\log \log n$, the only way to get a sparse cover where each vertex belongs to $O(\log n)$ clusters is through [9], with only $O(\log n)$ padding.

1.1 Results and Organization

In our first result (Theorem 15 in Section 5), we prove that K_r free graphs are strongly $(O(r), \Omega(\frac{1}{r}))$-decomposable. Providing quadratic improvement compared to [5].

---

[1] A metric space (X, d) has doubling dimension ddim if every ball of radius $2r$ can be covered by 2^{ddim} balls of radius r. The doubling dimension of a graph is the doubling dimension of its induced shortest path metric.

[2] $f(r)$ is a function coming from the Robertson and Seymour structure theorem [42].

Table 1 Summery of all known and new padding decompositions for various graph families.

Family	Partition type	Padding	δ	Reference
Previous results				
General graphs	Strong	$O(\log n)$	$\Omega(1)$	[10]
Doubling	Weak	$O(\text{ddim})$	$\Omega(1)$	[29, 2]
K_r minor free	Weak	$O(r^3)$	$\Omega(1)$	[33]
K_r minor free	Weak	$O(r^2)$	$\Omega(1)$	[23]
K_r minor free	Weak	$O(r)$	$\Omega(1)$	[5]
K_r minor free	Strong	$\exp(r)$	$\exp(-r)$	[6] [3]
K_r minor free	Strong	$O(r^2)$	$\Omega(\frac{1}{r^2})$	[5]
Our results				
Doubling	Strong	$O(\text{ddim})$	$\Omega(1)$	Corollary 9
K_r minor free	Strong	$O(r)$	$\Omega(\frac{1}{r})$	Theorem 15

Our second result (Corollary 9 in Section 4) is the first strong diameter padded decompositions for doubling graphs, which is also asymptotically tight. Specifically, we prove that graphs with doubling dimension ddim are strongly $O(\text{ddim})$-decomposable.

Both of these padded decomposition constructions are based on a technical theorem (Theorem 4 in Section 3). Given a set of centers N, such that each vertex has a center at distance at most Δ and at most τ centers at distance at most 3Δ ($\forall v,\ |B_G(v,3\Delta)\cap N| \leq \tau$), we construct a strong $(O(\log \tau), \Omega(1), 4\Delta)$-padded decomposition. We also provide an alternative construction for the decomposition of Theorem 4 in Appendix B. All of our decompositions can be efficiently constructed in polynomial time. See Table 1 for a summery of results on padded decompositions.

Our third result (Theorem 10 in Section 4) is a sparse cover for doubling graphs. For every parameter $t \geq 1$, we construct an $\big(O(t), O(2^{\text{ddim}/t} \cdot \text{ddim} \cdot \log t)\big)$-sparse cover scheme. Note that for $t = 1$ we (asymptotically) obtain the result of [6]. However, we also get the entire spectrum of padding parameters. In particular, for $t = \text{ddim}$ we get an $(O(\text{ddim}), O(\text{ddim} \cdot \log \text{ddim}))$-sparse cover scheme.

Next, we overview some of the previously known applications of strong diameter padded decomposition, and analyze the various improvements achieved using our results. Specifically:

1. Given an instance of the unique games problem where the input graph is K_r free, Alev and Lau [7] showed that if there exist an assignment that satisfies all but an ϵ-fraction of the edges, then there is an efficient algorithm that finds an assignment that satisfies all but an $O(r \cdot \sqrt{\epsilon})$-fraction. Using our padded decompositions for minor-free graphs, we can find an assignment that satisfies all but an $O(\sqrt{r \cdot \epsilon})$-fraction of the edges. See Section 6.1.
2. Using the framework of Filtser and Neiman [26], given an n vertex graph, with doubling dimension ddim, for every parameter $t > 1$ we construct a graph-spanner with stretch $O(t)$, lightness $O(2^{\frac{\text{ddim}}{t}} \cdot t \cdot \log^2 n)$ and $O(n \cdot 2^{\frac{\text{ddim}}{t}} \cdot \log n \cdot \log \Lambda)$ edges [4]. The only previous spanner of this type appeared in [26], was based on weak diameter decompositions, had the same stretch and lightness, while having no bound whatsoever on the number of edges. See Section 6.2.

[3] In fact [6] studied separating decompositions instead of padded (see Definition 16).
[4] Lightness is the ratio between the weight of the spanner to the weight of the MST. $\Lambda = {\max_{u,v\in V} d_G(u,v)}/{\min_{u,v\in V} d_G(u,v)}$ is the aspect ratio.

3. Elkin, Neiman and Wulff-Nilsen [19] constructed a path reporting distance oracle for K_r free graphs with stretch $O(r^2)$, space $O(n \cdot \log \Lambda \cdot \log n)$ and query time $O(\log \log \Lambda)$. That is, on a query $\{u, v\}$ the distance oracle returns a $u - v$ path P of weight at most $O(r^2) \cdot d_G(u, v)$ in $O(|P| + \log \log \Lambda)$ time. Using our strong diameter padded decompositions we improve the stretch to $O(r)$, while keeping the other parameters intact. See Appendix A.
4. We further use the framework of [19] to create a path reporting distance oracle for graphs having doubling dimension ddim with stretch $O(\text{ddim})$, space $O(n \cdot \text{ddim} \log \Lambda)$ and query time $O(\log \log \Lambda)$. This is the first path reporting distance oracle for doubling graphs. The construction uses our sparse covers. See Appendix A.

1.2 Related Work

Other than padded decompositions, separating decompositions have been studied. Here, instead of analyzing the probability to cut a ball, we analyze the probability to cut an edge [8, 36, 16, 22]. Separating decompositions been used to minimize the number of inter-cluster edges in a partition. In particular, strong diameter version of such partitions were used for SDD solvers [12].

Miller et al. [38] constructed strong diameter partitions for general graphs, which they later used to construct spanners and hop-sets in parallel and distributed regimes (see also [18]). Hierarchical partitions with strong diameter had been studied and used for constructing distributions over spanning trees with small expected distortion [17, 1], Ramsey spanning trees [3] and for universal Steiner trees [14]. Another type of partitions studied is when we require only weak diameter, and in addition for each cluster to be connected [21, 25].

Padded decompositions were studied for additional graph families. Kamma and Krauthgamer [31] showed that treewidth r graphs are weakly $O(\log r + \log \log n)$-decomposable. Abraham et al. [5] showed that treewidth r graphs are strongly $O(\log r + \log \log n)$-decomposable and strongly $(O(r), \Omega(\frac{1}{r}))$-decomposable. [5] also showed that pathwidth r graphs are strongly $O(\log r)$- decomposable. Finally [5] proved that genus g graphs are strongly $O(\log g)$-decomposable, improving a previous weak diameter version of Lee and Sidiropoulos [35].

1.3 Technical Ideas

The basic approach for creating padded decompositions is by ball carving [10, 2]. That is, iteratively create clusters by taking a ball centered around some vertex, with radius drawn according to exponential distribution. The process halts when all the vertices are clustered. Intuitively, if every vertex might join the cluster associated with at most τ centers, the padding parameter is $O(\log \tau)$. We think of these centers as *threateners*. This approach worked very well for general graphs as the number of vertices is n. Similarly it also been used for doubling graphs, where the number of threateners is bounded by $2^{O(\text{ddim})}$. However, in doubling graphs ball carving produces only weak diameter clustering.

Our main technical contribution is a proof that the intuition above holds for strong diameter as well. Specifically, we show that if there is a set N of centers such that each vertex has a center at distance at most Δ, and at most τ centers at distance 3Δ (these are the threateners), then the graph is strongly $(O(\log \tau), \Omega(1), 4\Delta)$-decomposable. We use the clustering approach of Miller et al. [38] with exponentially distributed starting times. In short, in [38] clustering each center x samples a starting time δ_x. Vertex v joins the cluster of the center x_i maximizing $\delta_x - d_G(x, v)$. This approach guaranteed to creates strong

diameter clusters. The key observation is that if for every center $y \neq x_i$, $(\delta_{x_i} - d_G(x_i, v)) - (\delta_y - d_G(y, v)) \geq 2\gamma\Delta$, then the ball $B_G(v, \gamma\Delta)$ is fully contained in the cluster of x_i. Using truncated exponential distribution, we lower bound the probability of this event by $e^{-\gamma \cdot O(\log \tau)}$. It is the first time [38]-like algorithm is used to create padded decompositions.

In addition to the [38]-based algorithm, we also show a simpler algorithm, based on cone carving ([17]). The cone approach, although less involved, is inherently sequential and implies dependencies of each vertex on the entire center set. [38] algorithm can be efficiently implemented in distributed and parallel setting. Moreover, as each vertex depends only on centers in its local area, we are able to use the Lovász Local Lemma to create a sparse cover from padded decompositions.

Decompositions of K_r free graphs did not use ball carving directly. Rather, they tend to use the topological structure of the graph. We say that a cluster of G has an r-core with radius Δ if it contains at most r shortest paths (w.r.t. d_G) such that each vertex is at distance at most Δ from one of these paths. [5]'s strong decomposition for K_r free graphs is based on a partition into 1-core clusters, such that a ball with radius $\gamma\Delta$ is cut with probability at most $1 - e^{-O(\gamma r^2)}$. This partition is the reason for their $O(r^2)$ padding parameter. Although not stated explicitly, [5] also constructed a partition into r-core clusters, such that a ball with radius $\gamma\Delta$ is cut with probability at most $1 - e^{-O(\gamma r)}$. Apparently, [5] lacked an algorithm for partitioning r-clusters. Taking a union of Δ-nets from each shortest path to the center set N, it will follow that each vertex has at most $O(r)$ centers in its $O(\Delta)$ neighborhood. In particular, our theorem above implies a clustering of each r-core cluster into bounded diameter clusters. Our strong decomposition with parameter $O(r)$ follows.

2 Preliminaries

Graphs. We consider connected undirected graphs $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_{\geq 0}$. We say that vertices v, u are neighbors if $\{v, u\} \in E$. Let d_G denote the shortest path metric in G. $B_G(v, r) = \{u \in V \mid d_G(v, u) \leq r\}$ is the ball of radius r around v. For a vertex $v \in V$ and a subset $A \subseteq V$, let $d_G(x, A) := \min_{a \in A} d_G(x, a)$, where $d_G(x, \emptyset) = \infty$. For a subset of vertices $A \subseteq V$, let $G[A]$ denote the induced graph on A, and let $G \setminus A := G[V \setminus A]$. The *diameter* of a graph G is $\mathrm{diam}(G) = \max_{v,u \in V} d_G(v, u)$, i.e. the maximal distance between a pair of vertices. Given a subset $A \subseteq V$, the *weak*-diameter of A is $\mathrm{diam}_G(A) = \max_{v,u \in A} d_G(v, u)$, i.e. the maximal distance between a pair of vertices in A, w.r.t. to d_G. The *strong*-diameter of A is $\mathrm{diam}(G[A])$, the diameter of the graph induced by A.

A graph H is a *minor* of a graph G if we can obtain H from G by edge deletions/contractions, and isolated vertex deletions. A graph family $\mathcal{G}$ is *H-minor-free* if no graph $G \in \mathcal{G}$ has H as a minor. Some examples of minor free graphs are planar graphs (K_5 and $K_{3,3}$ free), outer-planar graphs (K_4 and $K_{3,2}$ free), series-parallel graphs (K_4 free) and trees (K_3 free).

Doubling dimension. The doubling dimension of a metric space is a measure of its local "growth rate". A metric space (X, d) has doubling constant λ if for every $x \in X$ and radius $r > 0$, the ball $B(x, 2r)$ can be covered by λ balls of radius r. The doubling dimension is defined as $\mathrm{ddim} = \log_2 \lambda$. A d-dimensional ℓ_p space has $\mathrm{ddim} = \Theta(d)$, and every n point metric has $\mathrm{ddim} = O(\log n)$. We say that a weighted graph $G = (V, E, w)$ has doubling dimension ddim, if the corresponding shortest path metric (V, d_G) has doubling dimension ddim. The following lemma gives the standard packing property of doubling metrics (see, e.g., [29]).

▶ **Lemma 1** (Packing Property). *Let (X, d) be a metric space with doubling dimension* ddim*. If $S \subseteq X$ is a subset of points with minimum interpoint distance r that is contained in a ball of radius R, then $|S| \leq \left(\frac{2R}{r}\right)^{O(\mathrm{ddim})}$.*

Nets. A set $N \subseteq V$ is called a Δ-net, if for every vertex $v \in V$ there is a net point $x \in N$ at distance at most $d_G(v,x) \leq \Delta$, while every pair of net points $x, y \in N$, is farther than $d_G(x,y) > \Delta$. A Δ-net can be constructed efficiently in a greedy manner. In particular, by Lemma 1, given a Δ-net N in a graph of doubling dimension ddim, a ball of radius $R \geq \Delta$, will contain at most $\left(\frac{2R}{\Delta}\right)^{O(\text{ddim})}$ net points.

Padded Decompositions and Sparse Covers. Consider a *partition* $\mathcal{P}$ of V into disjoint clusters. For $v \in V$, we denote by $P(v)$ the cluster $P \in \mathcal{P}$ that contains v. A partition $\mathcal{P}$ is strongly Δ-*bounded* (resp. weakly Δ-bounded) if the strong-diameter (resp. weak-diameter) of every $P \in \mathcal{P}$ is bounded by Δ. If the ball $B_G(v, \gamma\Delta)$ of radius $\gamma\Delta$ around a vertex v is fully contained in $P(v)$, we say that v is γ-*padded* by $\mathcal{P}$. Otherwise, if $B_G(v, \gamma\Delta) \not\subseteq P(v)$, we say that the ball is *cut* by the partition.

▶ **Definition 2** (Padded Decomposition). *Consider a weighted graph $G = (V, E, w)$. A distribution $\mathcal{D}$ over partitions of G is strongly (resp. weakly) (β, δ, Δ)-padded decomposition if every $\mathcal{P} \in \text{supp}(\mathcal{D})$ is strongly (resp. weakly) Δ-bounded and for any $0 \leq \gamma \leq \delta$, and $z \in V$,*

$$\Pr[B_G(z, \gamma\Delta) \subseteq P(z)] \geq e^{-\beta\gamma} .$$

We say that a graph G admits a strong (resp. weak) (β, δ)-padded decomposition scheme, if for every parameter $\Delta > 0$ it admits a strongly (resp. weakly) (β, δ, Δ)-padded decomposition that can be sampled in polynomial time.

A related notion to padded decompositions is sparse covers.

▶ **Definition 3** (Sparse Cover). *A collection of clusters $\mathcal{C} = \{C_1, ..., C_t\}$ is called a (β, s, Δ)-sparse cover if the following conditions hold.*

1. *Bounded diameter: The strong diameter of every $C_i \in \mathcal{C}$ is bounded by Δ.*
2. *Padding: For each $v \in V$, there exists a cluster $C_i \in \mathcal{C}$ such that $B_G(v, \frac{\Delta}{\beta}) \subseteq C_i$.*
3. *Overlap: For each $v \in V$, there are at most s clusters in $\mathcal{C}$ containing v.*

We say that a graph G admits a (β, s)-sparse cover scheme, if for every parameter $\Delta > 0$ it admits a (β, s, Δ)-sparse cover that can be constructed in expected polynomial time.

Truncated Exponential Distributions. To create padded decompositions, similarly to previous works, we will use truncated exponential distribution. That is, exponential distribution conditioned on the event that the outcome lays in a certain interval. The $[\theta_1, \theta_2]$-*truncated exponential distribution* with parameter λ is denoted by $\mathsf{Texp}_{[\theta_1,\theta_2]}(\lambda)$, and the density function is: $f(y) = \frac{\lambda e^{-\lambda \cdot y}}{e^{-\lambda\cdot\theta_1} - e^{-\lambda\cdot\theta_2}}$, for $y \in [\theta_1, \theta_2]$. For the $[0,1]$-*truncated exponential distribution* we drop the subscripts and denote it by $\mathsf{Texp}(\lambda)$; the density function is $f(y) = \frac{\lambda e^{-\lambda\cdot y}}{1-e^{-\lambda}}$.

3 Strongly Padded Decomposition

In this section we prove the main technical theorem of this paper.

▶ **Theorem 4.** *Let $G = (V, E, w)$ be a weighted graph and $\Delta > 0, \tau = \Omega(1)$ parameters. Suppose that we are given a set $N \subseteq V$ of center vertices such that for every $v \in V$:*

- Covering. *There is $x \in N$ such that $d_G(v,x) \leq \Delta$.*
- Packing. *There are at most τ vertices in N at distance 3Δ, i.e. $|B_G(v, 3\Delta) \cap N| \leq \tau$.*

Then G admits a strongly $\left(O(\ln \tau), \frac{1}{16}, 4\Delta\right)$-padded decomposition that can be efficiently sampled.

We start with description of the [38] algorithm (with some adaptations), and its properties. Later, in Section 3.2 we will prove Theorem 4. An alternative construction is given in Appendix B.

3.1 Clustering Algorithm Based on Starting Times

As we make some small adaptations, and the role of the clustering algorithm is essential, we provide full details. Let $\Delta > 0$ be some parameter and let $N \subseteq V$ be some set of centers such that for every $v \in V$, $d_G(v, N) \leq \Delta$. For each center $x \in N$, let $\delta_x \in [0, \Delta]$ be some parameter. The choice of $\{\delta_x\}_{x \in N}$ differs among different implementations of the algorithm. In our case we will sample δ_x using truncated exponential distribution. Each vertex v will join the cluster C_x of the center $x \in N$ for which the value $\delta_x - d_G(x, v)$ is maximized. Ties are broken in a consistent manner [5]. Note that it is possible that a center $x \in N$ will join the cluster of a different center $x' \in N$. An intuitive way to think about the clustering process is as follows: each center x wakes up at time $-\delta_x$ and begins to "spread" in a continuous manner. The spread of all centers done in the same unit tempo. A vertex v joins the cluster of the first center that reaches it.

▷ Claim 5. Every non-empty cluster C_x created by the algorithm has strong diameter at most 4Δ.

Proof. Consider a vertex $v \in C_x$. First we argue that $d_G(v, x) \leq 2\Delta$. This will already imply that C_x has weak diameter 4Δ. Let x_v be the closest center to v, then $d_G(v, x_v) \leq \Delta$. As v joined the cluster of x, it holds that $\delta_x - d_G(v, x) \geq \delta_{x_v} - d_G(v, x_v)$. In particular $d_G(v, x) \leq \delta_x + d_G(v, x_v) \leq 2\Delta$.

Let $\mathcal{I}$ be the shortest path in G from v to x. For every vertex $u \in \mathcal{I}$ and center $x' \in N$, it holds that

$$\begin{aligned}\delta(x) - d_G(u, x) = \delta(x) - (d_G(v, x) - d_G(v, u)) &\geq \delta(x') - d_G(v, x') + d_G(v, u) \\ &\geq \delta(x') - d_G(u, x') \ .\end{aligned}$$

We conclude that $\mathcal{I} \subseteq C_x$, in particular $d_{G[C_x]}(v, x) \leq 2\Delta$. The claim now follows. ◁

▷ Claim 6. Consider a vertex v, and let $x_1, x_2, \ldots$ be an ordering of the centers w.r.t. $\delta(x_i) - d_G(v, x_i)$. That is $\delta(x_1) - d_G(v, x_1) \geq \delta(x_2) - d_G(v, x_2) \geq \ldots$. Set $\Upsilon = (\delta(x_1) - d_G(v, x_1)) - (\delta(x_2) - d_G(v, x_2))$. Then for every vertex u such that $d_G(v, u) < \frac{\Upsilon}{2}$ it holds that $u \in C_{x_1}$.

Proof. For every center $x_i \neq x_1$ it holds that,

$$\delta(x_1) - d_G(u, x_1) > \delta(x_1) - d_G(v, x_1) - \frac{\Upsilon}{2} \geq \delta(x_i) - d_G(v, x_i) + \frac{\Upsilon}{2} > \delta(x_i) - d_G(u, x_i) \ .$$

In particular, $u \in C_{x_1}$. ◁

3.2 Proof of Theorem 4

For every center $x \in N$, we sample $\delta'_x \in [0, 1]$ according to $\mathsf{Texp}(\lambda)$ truncated exponential distribution with parameter $\lambda = 2 + 2\ln\tau$. Set $\delta_x = \delta'_x \cdot \Delta \in [0, \Delta]$. We execute the clustering algorithm from Section 3.1 with parameters $\{\delta_x\}_{x \in N}$ to get a partition $\mathcal{P}$.

[5] That is we have some order $x_1, x_2, \ldots$. Among the centers x_i that minimize $\delta_{x_i} - d_G(x_i, v)$, v joins the cluster of the center with minimal index.

According to Claim 5, we created a distribution over strongly 4Δ-bounded partitions. Consider some vertex $v \in V$ and parameter $\gamma \leq \frac{1}{4}$. We will argue that the ball $B = B_G(v, \gamma\Delta)$ is fully contained in $P(v)$ with probability at least $e^{-O(\gamma \log \tau)}$. Let N_v be the set of centers x for which there is non zero probability that C_x intersects B. Following the calculation in Claim 5, each vertex joins the cluster of a center at distance at most 2Δ. By triangle inequality, all the centers in N_v are at distance at most $(2+\gamma)\Delta \leq 3\Delta$ from v. In particular $|N_v| \leq \tau$.

Set $N_v = \{x_1, x_2, \dots\}$ ordered arbitrarily. Denote by $\mathcal{F}_i$ the event that v joins the cluster of x_i, i.e. $v \in C_{x_i}$. Denote by $\mathcal{C}_i$ the event that v joins the cluster of x_i, but not all of the vertices in B joined that cluster, that is $v \in C_{x_i} \cap B \neq B$. To prove the theorem, it is enough to show that $\Pr[\cup_i \mathcal{C}_i] \leq 1 - e^{-O(\gamma\cdot\lambda)}$. Set $\alpha = e^{-2\gamma\cdot\lambda}$.

▷ Claim 7. For every i, $\Pr[\mathcal{C}_i] \leq (1-\alpha)\left(\Pr[\mathcal{F}_i] + \frac{1}{e^\lambda - 1}\right)$.

Proof. As the order in N_v is arbitrary, assume w.l.o.g. that $i = |N_v|$ and denote $x = x_{|N_v|}$, $\mathcal{C} = \mathcal{C}_i$, $\mathcal{F} = \mathcal{F}_i$, $\delta = \delta_{x_i}$ and $\delta' = \delta'_{x_i}$. Let $X \in [0,1]^{|N_v|-1}$ be the vector where the j'th coordinate equals δ'_{x_j}. Set $\rho_X = \frac{1}{\Delta}\cdot\left(d_G(x,v) + \max_{j<|N_v|}\left\{\delta_{x_j} - d_G(x_j,v)\right\}\right)$. Note that ρ_X is the minimal value of δ' such that if $\delta' > \rho_X$, that x has the maximal value $\delta_x - d_G(x,v)$, and therefor v will join the cluster of x. Note that it is possible that $\rho_X > 1$. Conditioning on the samples having values X, and assuming that $\rho_X \leq 1$ it holds that

$$\Pr[\mathcal{F} \mid X] = \Pr[\delta' > \rho_X] = \int_{\rho_X}^{1} \frac{\lambda \cdot e^{-\lambda y}}{1 - e^{-\lambda}} dy = \frac{e^{-\rho_X\cdot\lambda} - e^{-\lambda}}{1 - e^{-\lambda}} \ .$$

If $\delta' > \rho_X + 2\gamma$ then $\delta - d_G(x,v) > \max_{j\neq i}\{\delta_{x_i} - d_G(x_i,v)\} + 2\gamma\Delta$. In particular, by Claim 6 the ball B will be contained in C_x. We conclude

$$\begin{aligned}
\Pr[\mathcal{C} \mid X] &\leq \Pr[\rho_X \leq \delta' \leq \rho_X + 2\gamma] \\
&= \int_{\rho_X}^{\max\{1,\rho_X+2\gamma\}} \frac{\lambda \cdot e^{-\lambda y}}{1 - e^{-\lambda}} dy \\
&\leq \frac{e^{-\rho_X\cdot\lambda} - e^{-(\rho_X+2\gamma)\cdot\lambda}}{1 - e^{-\lambda}} \\
&= \left(1 - e^{-2\gamma\cdot\lambda}\right)\cdot\frac{e^{-\rho_X\cdot\lambda}}{1 - e^{-\lambda}} \\
&= (1-\alpha)\cdot\left(\Pr[\mathcal{F} \mid X] + \frac{1}{e^\lambda - 1}\right) \ .
\end{aligned}$$

Note that if $\rho_X > 1$ then $\Pr[\mathcal{C} \mid X] = 0 \leq (1-\alpha)\cdot\left(\Pr[\mathcal{F} \mid X] + \frac{1}{e^\lambda - 1}\right)$ as well. Denote by f the density function of the distribution over all possible values of X. Using the law of total probability, we can bound the probability that the cluster of x cuts B

$$\begin{aligned}
\Pr[\mathcal{C}] &= \int_X \Pr[\mathcal{C} \mid X]\cdot f(X)\ dX \\
&\leq (1-\alpha)\cdot\int_X \left(\Pr[\mathcal{F} \mid X] + \frac{1}{e^\lambda - 1}\right)\cdot f(X)\ dX \\
&= (1-\alpha)\cdot\left(\Pr[\mathcal{F}] + \frac{1}{e^\lambda - 1}\right)
\end{aligned}$$

◁

We bound the probability that the ball B is cut.

$$\begin{aligned}\Pr\left[\cup_i \mathcal{C}_i\right] = \sum_{i=1}^{|N_v|} \Pr\left[\mathcal{C}_i\right] &\leq (1-\alpha)\cdot \sum_{i=1}^{|N_v|}\left(\Pr\left[\mathcal{F}_i\right] + \frac{1}{e^\lambda - 1}\right)\\ &\leq \left(1-e^{-2\gamma\cdot\lambda}\right)\cdot\left(1+\frac{\tau}{e^\lambda-1}\right)\\ &\leq \left(1-e^{-2\gamma\cdot\lambda}\right)\cdot\left(1+e^{-2\gamma\cdot\lambda}\right) = 1-e^{-4\gamma\cdot\lambda}\ ,\end{aligned}$$

where the last inequality follows as $e^{-2\gamma\lambda} = \frac{e^{-2\gamma\lambda}\left(e^\lambda-1\right)}{e^\lambda-1} \geq \frac{e^{-2\gamma\lambda}\cdot e^{\lambda-1}}{e^\lambda-1} \geq \frac{e^{\frac{\lambda}{2}-1}}{e^\lambda-1} = \frac{\tau}{e^\lambda-1}$.

▶ Remark 8. Actually we can prove a generalized version of Theorem 4. Suppose that there is a set N of centers such that each vertex $v \in V$ has at least one center at distance at most Δ and at most τ_v centers at distance 3Δ. Then for every parameter $\lambda = \Omega(1)$, there is a distribution over partitions with strong diameter 4Δ such that for every parameter $\gamma \in (0, \frac{1}{4})$, the ball around every vertex v of radius $\gamma\Delta$ is cut with probability at most $(1-e^{-2\gamma\lambda})(1+\frac{\tau_v}{e^\lambda-1})$.

4 Doubling Dimension

Our strongly padded decompositions for doubling graphs are a simple corollary of Theorem 4.

▶ **Corollary 9.** *Let $G=(V,E,w)$ be a weighted graph with doubling dimension* ddim*. Then G admits a strong $(O(\text{ddim}), \Omega(1))$-padded decomposition scheme.*

Proof. Fix some $\Delta > 0$. Let N be a Δ-net of X. According to Lemma 1, for every vertex v, the number of net points at distance 3Δ is bounded by $2^{O(\text{ddim})}$. The corollary follows by Theorem 4. ◀

Next, we construct a sparse cover scheme.

▶ **Theorem 10.** *Let $G=(V,E,w)$ be a weighted graph with doubling dimension* ddim *and parameter $t = \Omega(1)$. Then G admits an $\left(O(t), O(2^{\text{ddim}/t}\cdot\text{ddim}\cdot\log t)\right)$-sparse cover scheme. In particular, there is an $(O(\text{ddim}), O(\text{ddim}\cdot\log\text{ddim}))$-sparse cover scheme.*

Proof. Let $\Delta > 0$ be the diameter parameter. Let $\alpha = \theta(1)$ be a constant to be determined later, set $\beta = \alpha\cdot t$. We will construct a $\left(\beta, O(2^{\text{ddim}/t}\cdot\text{ddim}\cdot\log t), 4\Delta\right)$-sparse cover. As Δ is arbitrary, this will imply $\left(4\beta, O(2^{\text{ddim}/t}\cdot\text{ddim}\cdot\log t)\right)$-sparse cover scheme.

The sparse cover is constructed by sampling $O(2^{\text{ddim}/t}\cdot\text{ddim}\cdot\log t)$ independent partitions using Corollary 9 with diameter parameter Δ, and taking all the clusters from all the partitions to the cover. The sparsity and strong diameter properties are straightforward. To argue that each vertex is padded in some cluster we will use the constructive version of the Lovász Local Lemma by Moser and Tardos [39].

▶ **Lemma 11** (Constructive Lovász Local Lemma)**.** *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a set of events determined by these variables. For $A\in\mathcal{A}$ let $\Gamma(A)$ be a subset of $\mathcal{A}$ satisfying that A is independent from the collection of events $\mathcal{A}\setminus(\{A\}\cup\Gamma(A))$. If there exist an assignment of reals $x:\mathcal{A}\to(0,1)$ such that*

$$\forall A\in\mathcal{A}\ :\quad \Pr[A] \leq x(A)\Pi_{B\in\Gamma(A)}(1-x(B))\ ,$$

then there exists an assignment to the variables $\mathcal{P}$ not violating any of the events in $\mathcal{A}$. Moreover, there is an algorithm that finds such an assignment in expected time $\sum_{A\in\mathcal{A}}\frac{x(A)}{1-x(A)}\cdot\text{poly}\left(|\mathcal{A}|+|\mathcal{P}|\right)$.

Formally, recall the construction of Theorem 4 used in Corollary 9. Let N be a Δ-net, that we will use as centers. Consider a vertex $v \in V$, and fix some sample of the starting times $\{\delta_x\}_{x\in N}$. Let x_v be the vertex maximizing $\delta_x - d_G(x,v)$ and y_v the second largest. In other words, $\delta_{x_v} - d_G(x_v,v) \geq \delta_{y_v} - d_G(y_v,v) \geq \max_{x\in N\setminus\{x_v,y_v\}}\{\delta_x - d_G(x,v)\}$. Let Ψ_v be the event that $(\delta_{x_v} - d_G(x_v,v)) - (\delta_{y_v} - d_G(y_v,v)) < 4\frac{\Delta}{\beta}$. Recall that the event that the ball of radius $2\frac{\Delta}{\beta}$ around v is cut contained in Ψ_v. Following the analysis of Theorem 4, $\Pr[\Psi_v] \leq 1 - e^{-O(\mathrm{ddim}\cdot 4\cdot\frac{\Delta}{\beta}/\Delta)} = 1 - 2^{-\mathrm{ddim}/t}$, where the equality follows by an appropriate choice of α.

Let $\hat{x}$ be the closest center to v. It holds that $\delta_{\hat{x}} - d_G(\hat{x},v) \geq -\Delta$, while for every center x at distance larger that 3Δ it holds that $\delta_x - d_G(x,v) \leq -2\Delta$. Therefore Ψ_v depends only on centers at distance at most 3Δ. In particular, by triangle inequality, if v and u are farther away than 6Δ, Ψ_v and Ψ_u are independent.

We take $m = \alpha_m \cdot 2^{\frac{\mathrm{ddim}}{t}} \cdot \mathrm{ddim} \cdot \log t$ independent partitions of X using Corollary 9, for $\alpha_m = \Theta(1)$ to be determined later. Denote by Ψ_v^i the event representing Ψ_v in the i'th partition. Let $\Phi_v = \bigwedge_{i=1}^m \Psi_v^i$ be the event that v "failed" in all the partitions. It holds that

$$\Pr[\Phi_v] \leq \left(1 - 2^{-\mathrm{ddim}/t}\right)^m \leq e^{-2^{-\mathrm{ddim}/t}\cdot m} = e^{-\alpha_m\cdot\mathrm{ddim}\cdot\log t} .$$

Note that if Ψ_v did not occurred, then the ball of radius $2\frac{\Delta}{\beta}$ around v was contained in a single cluster in at least one partition.

Let Y be an $\frac{\Delta}{\beta}$-net of X. Set $\mathcal{A} = \{\Phi_v\}_{v\in Y}$, to be a set of events determined by $\{\delta_x^i\}_{x\in N, 1\leq i\leq m}$ (δ_x^i denotes δ_x in the i'th partition). Each event Φ_v might depend only on events Φ_u corresponding to vertices u at distance at most 6Δ from v. By Lemma 1, Φ_v is independent of all, but $\Gamma(\Phi_v) \leq \left(\frac{12\Delta}{\Delta/\beta}\right)^{O(\mathrm{ddim})} = 2^{O(\mathrm{ddim}\cdot\log t)}$ events. For every $\Phi_v \in \mathcal{A}$, set $x(\Phi_v) = p = 2^{-O(\mathrm{ddim}\cdot\log t)}$, such that $\max_{v\in Y}|\Gamma(v)| \leq \frac{1}{2p}$. Then, for every $\Phi_v \in \mathcal{A}$ it holds that,

$$x(\Phi_v)\cdot\Pi_{B\in\Gamma(\Phi_v)}(1-x(B)) = p\cdot(1-p)^{|\Gamma(\Phi_v)|} \geq p\cdot(1-p)^{\frac{1}{2p}} \geq \frac{p}{e} \geq \Pr(\Phi_v) ,$$

where the last inequality holds for large enough α_m. By Lemma 11 we can efficiently find an assignment to $\{\delta_x^i\}_{x\in N,1\leq i\leq m}$ such that none of the events $\{\Phi_v\}_{v\in Y}$ occurred. Under this assumption, we argue that our sparse cover has the padding property. Consider some vertex $v \in V$. There is a net point $u \in Y$ at distance at most $\frac{\Delta}{\beta}$ from v. As the event Φ_u did not occur, there is some cluster C in the cover in which u is padded. In particular $B_G(v,\gamma\Delta) \subseteq B_G(u,2\gamma\Delta) \subseteq C$ as required.

Suppose that $|V| = n$, then the running time is $|Y|\cdot\frac{p}{1-p}\cdot\mathrm{poly}\left(|Y|+|Y|\right) = \mathrm{poly}(n)$. ◀

5 Minor Free Graphs

Our clustering algorithm is based on the clustering algorithm of [5], with a small modification. The clustering of [5] has two steps. In the first step the graph is partitioned into *r-Core* clusters (see Definition 12 bellow). While r-core clusters do not have bounded diameter, they do have a simple geometric structure. Moreover, this clustering also has the padding property for small balls. In the second step, each r-core cluster is partitioned into bounded diameter sub-clusters using Theorem 4.

▶ **Definition 12** (r-Core). *Given a weighted graph $G = (V, E, w)$, we say that G has an* r-core *with radius Δ, if there is a set of at most r shortest paths $\mathcal{I}_1, \ldots, \mathcal{I}_{r'}$ such that for every $v \in V$, $d_G(v, \cup_i \mathcal{I}_i) \leq \Delta$.*

Given a cluster $C \subseteq G$, we say that C is an r-core cluster with radius Δ, if $G[C]$ has an r-core *with radius Δ. Given a partition $\mathcal{P}$ of G, we say that it is an r-core partition with radius Δ if each cluster $C \in \mathcal{P}$, is an r-core cluster with radius Δ.*

The following theorem was proved implicitly in [5].

▶ **Lemma 13** (Core Clustering [5]). *Given a weighted graph $G = (V, E, w)$ that excludes K_r as a minor and a parameter $\Delta > 0$, there is a distribution $\mathcal{D}$ over r-core partitions with radius Δ, such that for every vertex $v \in V$ and $\gamma \in (0, \Omega(\frac{1}{r}))$ it holds that*

$$\Pr\left[B_G(v, \gamma\Delta) \subseteq P(v)\right] \geq e^{-O(r \cdot \gamma)} .$$

Even though we will not provide full details of the proof of Lemma 13, we will describe the algorithm itself and provide some intuition for the core clustering in Section 5.2. Our clustering algorithm will be executed in two steps: first we partition the graph into core clustering (Lemma 13) and then we partition each r-core cluster using Theorem 4.

Some historical notes: [5] presented two different algorithms for strong and weak padded decompositions. Each of these algorithms consisted of two steps. For weak decompositions, essentially they first partitioning the graph using r-core clustering. Secondly, instead of partition further each cluster, they pick a net from the r-cores in all the clusters, and iteratively grow balls around net points, ending with weak diameter guarantee. For strong decompositions, they partition the graph into 1-core clusters (instead of r-core), ending with a probability of only $e^{-O(r^2 \cdot \gamma)}$ for a vertex x to be γ-padded.

5.1 Strong Padded Partitions for K_r Minor Free Graphs

▶ **Lemma 14.** *Let $G = (V, E, w)$ be a weighted graph that has an r-core with radius Δ. Then G admits a strong $(O(\log r), \Omega(1), \Delta)$-padded decomposition.*

Proof. Let $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_{r'}$ be the r-core of G. For each i, let N_i be a $\frac{\Delta}{8}$-net of $\mathcal{I}_i$. Set $N = \cup_i N_i$. Every vertex $v \in V$ has some vertex in N at distance at most $\frac{\Delta}{4}$. Indeed, by definition of r-core, there is $x \in \mathcal{I}_i$ such that $d_G(v, x) \leq \frac{\Delta}{8}$. Furthermore, there is a net point $y \in N_i$ at distance at most $\frac{\Delta}{8}$ from x. By triangle inequality $d_G(v, y) \leq \frac{\Delta}{4}$. As $\mathcal{I}_i$ is a shortest path and N_i is a $\frac{\Delta}{8}$-net, there are at most $O(1)$ net points at distance $\frac{3}{4}\Delta$ from v in N_i. We conclude that in N there are at most $O(r)$ net points at distance $\frac{3}{4}\Delta$ from v. The lemma now follows by Theorem 4. ◀

▶ **Theorem 15.** *Let $G = (V, E, w)$ be a weighted graph that excludes K_r as a minor. Then G admits a strong $\left(O(r), \Omega(\frac{1}{r})\right)$-padded decomposition scheme.*

Proof. Let $\Delta > 0$ be some parameter. We construct the decomposition in two steps. First we sample an r-core partition $\mathcal{P}$ with radius parameter Δ using Lemma 13. Next, for every cluster $C \in \mathcal{P}$, we create a partition $\mathcal{P}_C$ using Lemma 14. The final partition is simply $\cup_{C \in \mathcal{P}} \mathcal{P}_C$, the union of all the clusters in all the created partitions. It is straightforward that the created partition has strong diameter Δ. To analyze the padding, consider a vertex $v \in V$ and parameter $0 < \gamma \leq \Omega(\frac{1}{r})$. Denote by C_v the cluster containing v in $\mathcal{P}$, and by $P(v)$ the cluster of v in the final partition. Then,

$$\begin{aligned}\Pr\left[B_G(v, \gamma\Delta) \subseteq P(v)\right] &= \Pr\left[B_G(v, \gamma\Delta) \subseteq P(v) \mid B_G(v, \gamma\Delta) \subseteq C_v\right] \cdot \Pr\left[B_G(v, \gamma\Delta) \subseteq C_v\right] \\ &\geq e^{-O(\gamma \cdot r)} \cdot e^{-O(\gamma \cdot \log r)} = e^{-O(\gamma \cdot r)} ,\end{aligned}$$

where we used the fact that conditioning on $B_G(v, \gamma\Delta) \subseteq C_v$, it holds that $B_G(v, \gamma\Delta) = B_{G[C_v]}(v, \gamma\Delta)$. ◀

5.2 The Core Clustering Algorithm

In this section we describe the construction of the partition from Lemma 13. Afterwards, we will provide some intuition regarding the proof. For full details, we refer to [5]. Given two disjoint subsets $A, B \subseteq V$, we write $A \sim B$ if there exists an edge from a vertex in A to some vertex in B.

We denote the partition created by the algorithm by $\mathcal{S}$, and the clusters by $\{S_1, S_2, \dots\}$. The clusters are constructed iteratively. Initially $G_1 = G$. At step i, $G_i = G \setminus \cup_{j=1}^{i-1} S_j$. For a connected component $C \in G_i$, let $\mathcal{K}_{|C} = \{S_j \mid j < i \wedge C \sim S_j\}$ be the set of previously created clusters with a neighbor in C_i. To create S_i, pick arbitrary connected component C_i in G_i, and a vertex $x_i \in C_i$. For every neighboring cluster $S_j \in \mathcal{K}_{|C_i}$, pick arbitrary vertex $u_j \in C_i$ such that u_j has a neighbor in S_j. For each such u_j, let $\mathcal{I}_j$ be the shortest path in G_i from x_i to u_j. Let T_i be the tree created by the union of $\{\mathcal{I}_j\}_{S_j \in \mathcal{K}_{|C_i}}$ [6]. Sample a radius parameter R_i using truncated exponential distribution $\mathsf{Texp}_{[0,1]}(2r)$. The cluster S_i defined as $B_{G_i}(T_i, R_i\Delta)$, the set of all vertices at distance at most $R_i\Delta$ from T_i w.r.t. d_{G_i}. This finishes the construction of S_i. The algorithm halts when all the vertices are clustered. See pseudo-code in Algorithm 1. See also Figure 1 for illustration of the algorithm.

Algorithm 1 `Core-Partition`(G,Δ,r).

1: Let $G_1 \leftarrow G$, $i \leftarrow 1$.
2: Let $\mathcal{S} \leftarrow \emptyset$.
3: **while** G_i is non-empty **do**
4: Let C_i be a connected component of G_i.
5: Pick arbitrary $x_i \in C_i$. For each $S_j \in \mathcal{K}_{|C_i}$, let $u_j \in C_i$ be some vertex with a neighbor in S_j.
6: Let T_i be a tree rooted at x_i, consisting of shortest paths towards $\{u_j \mid S_j \in \mathcal{K}_{|C_i}\}$.
7: Let R_i be a random variable drawn independently from the distribution $\mathsf{Texp}_{[0,1]}(2r)$.
8: Let $S_i \leftarrow B_{G_i}(T_i, R_i\Delta)$.
9: Add S_i to $\mathcal{S}$.
10: $G_{i+1} \leftarrow G_i \setminus S_i$.
11: $i \leftarrow i + 1$.
12: **end while**
13: **return** $\mathcal{S}$.

Provided that the graph G excludes K_r as a minor, for every C_i it holds that $\left|\mathcal{K}_{|C_i}\right| \leq r-2$. Indeed, by induction for every $S_j, S_{j'} \in \mathcal{K}_{|C_i}$, there is an edge between S_j to $S_{j'}$ [7]. Assume for contradiction that $\left|\mathcal{K}_{|C_i}\right| \geq r-1$. By contracting all the internal edges in C_i and in the clusters in $\mathcal{K}_{|C_i}$ we will obtain K_r as a minor, a contradiction. It follows that for every i, T_i is an r-core of S_i. In particular, Algorithm 1 indeed produces an r-core partitions with radius Δ.

Abraham et al. [5] called the core T_i of each cluster a *skeleton*. Their algorithm induce an iterative process that creates "skeletons" and removes their R_i neighborhoods (a buffer) from the graph. R_i was sampled according to truncated exponential distribution. They called such an algorithm a *threatening* skeleton-process. In general, they consider such a process where each R_i is drawn according to $\mathsf{Texp}_{[l,u]}(\frac{b}{u-l})$, for $0 = l < u \leq 1$.

[6] Note that there is always a way to pick $\{\mathcal{I}_j\}_{S_j \in \mathcal{K}_{|C_i}}$ such that T_i will indeed be a tree.
[7] To see this note that there is a path between u_j to $u_{j'}$ in C_i. Therefore, when creating $S_{j'}$ (assuming $j < j'$), it was the case that $S_j \in \mathcal{K}_{|C_{j'}}$. In particular $T_{j'}$ contains a vertex with neighbor in S_j.

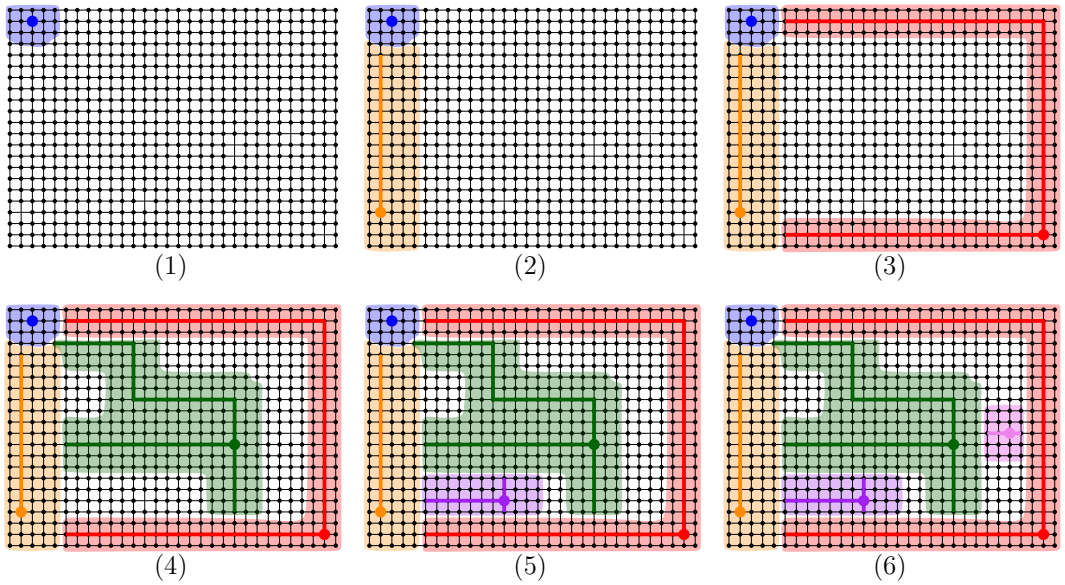


Figure 1 *The figure illustrates the 6 first steps in Algorithm 1. Here G is the (weighted) grid graph. Note that G excludes K_5 as a minor. In step (4), G_4 is the graph induced by all the vertices not colored in blue, orange or red. G_4 has a single connected component C_4. The green vertex defined as x_4. $\mathcal{K}_{|C_i}$ consist of 3 clusters S_1, S_2, S_3 colored respectively by blue, orange and red. T_4 is a tree rooted in x_4 colored in bold green, that consist of 3 shortest paths. Each of S_1, S_2, S_3 has a leaf of T_4 as a neighbor. R_4 is chosen according to $\mathsf{Texp}_{[0,1]}(10)$. The new cluster S_4, colored in green, consist of all vertices in C_4 at distance at most $R_4\Delta$ from T_4 w.r.t. d_{G_4}.*

Let $\gamma > 0$ be a padding parameter, fix some vertex $z \in V$ and set $B_z = B_G(z, \gamma\Delta)$. We say that a skeleton T_i threatens z if $d_{G_i}(z, T_i) \leq (u+\gamma)\Delta$, in other words, if there is a positive probability that some vertex of B_z joins C_i. Let $\mathcal{J}_z = \{T_i \mid d_{G_i}(z, T_i) \leq (u+\gamma)\Delta\}$ be the set of threatening skeletons. To bound the probability that B_z is cut, [5] first bound the expected number of threatening skeletons. A key lemma in [5] is that if we guaranteed that for every i, $|\mathcal{K}_{|C_i}| \leq s$, and sample each radius R_i from $\mathsf{Texp}_{[l,u]}(\frac{b}{u-l})$ for $b = 2s$, it holds that

$$\mathbb{E}[|\mathcal{J}_z|] \leq 3e^{(2s+1)\cdot(1+\gamma/u)} .$$

In a second key lemma, [5] argued that the probability that B_z is cut by a threatening skeleton-process, provided that $\tau = \mathbb{E}[|\mathcal{J}_z|]$, is at most

$$(1 - e^{-2b\gamma/(u-l)}) \left(1 + \frac{\tau}{e^b - 1}\right) .$$

In our case, as G is K_r free, thus we can pick $s = r - 2$. In Algorithm 1 we used the parameters $l = 0$, $u = 1$ and $b = 2r$. Therefore $\mathbb{E}[|\mathcal{J}_z|] \leq 3e^{(2r+1)\cdot(1+\gamma)}$. Assuming that $\gamma = O(\frac{1}{r})$, we conclude that the probability that B_z is cut is at most

$$\left(1 - e^{-4r\gamma}\right) \left(1 + \frac{3e^{(2r+1)\cdot(1+\gamma)}}{e^{2r} - 1}\right) = O(r\gamma) .$$

In particular, the probability that B_z is padded is at least $1 - O(r\gamma) = e^{-O(r\gamma)}$.

6 Applications

In this section we present some applications of stochastic decompositions. Some applications are using a weaker type of decomposition called *separating* decompositions. The difference being that padding decompositions bound the probability for a ball to be cut, while separating decompositions bound the probability of an edge to be cut.

▶ **Definition 16** (Separating Decomposition)**.** *A distribution $\mathcal{D}$ over partitions of a graph G is strongly (resp. weakly) (β, Δ)-separating decomposition if every $\mathcal{P} \in \text{supp}(\mathcal{D})$ is strongly (resp. weakly) Δ-bounded and for every pair $u, v \in V$, $\Pr[P(v) \neq P(u)] \leq \beta \cdot \frac{d_G(u,v)}{\Delta}$.*

Note that in contrast to padding decomposition, there is no upper bound δ on the distance between u to v. Nevertheless, we argue that padded decompositions imply separating ones.

▶ **Lemma 17.** *Let $G = (V, E, w)$ be a weighted graph with a strongly (β, δ, Δ)-padded decomposition $\mathcal{D}$ such that $\delta \geq \frac{1}{\beta}$. Then $\mathcal{D}$ is also a strongly (β, Δ)-separating decomposition.*

Proof. Let $v, u \in V$ be a pair of vertices. If $d_G(u, v) \geq \frac{\Delta}{\beta}$, then obviously $\Pr[P(v) \neq P(u)] \leq 1 \leq \beta \cdot \frac{d_G(u,v)}{\Delta}$. Thus we can assume $d_G(u, v) \leq \frac{\Delta}{\beta} \leq \delta\Delta$. Set $\gamma = \frac{d_G(u,v)}{\Delta}$. It holds that

$$\Pr[P(v) = P(u)] \geq \Pr\left[B_G\left(v, \gamma\Delta\right) \subseteq P(v)\right] \geq e^{-\beta\gamma} \geq 1 - \beta\gamma \ .$$

In particular, $\Pr[P(v) \neq P(u)] \leq \beta\gamma = \beta \cdot \frac{d_G(u,v)}{\Delta}$ as required. ◀

Applying Lemma 17 on Corollary 9 and Theorem 15 we conclude,

▶ **Corollary 18.** *Let G be a weighted graph and $\Delta > 0$ some parameter.*

- *If G excludes K_r as a minor, it admits an efficient strongly $(O(r), \Delta)$-separating decomposition.*
- *If G has doubling dimension* ddim*, it admits an efficient strongly $(O(\text{ddim}), \Delta)$-separating decomposition.*

6.1 Approximation for Unique Games on Minor Free Graphs

In the *Unique Games* problem we are give a graph $G = (V, E)$, an integer $k \geq 1$ and a set of permutations $\Pi = \{\pi_{uv}\}_{uv \in E}$ on $[k]$ satisfying $\pi_{uv} = \pi_{vu}^{-1}$. Given an assignment $x : V \to [k]$, the edge $uv \in E$ is satisfied if $\pi_{uv}(x(u)) = x(v)$. The problem is to find an assignment that maximizes the number of satisfied edges. The Unique Games Conjecture of Khot [32] postulates that it is NP-hard to distinguish whether a given instance of unique games is almost satisfiable or almost unsatisfiable. The unique games conjecture was thoroughly studied. The conjecture has numerous implications.

Alev and Lau [7] studied a special case of the unique games problem, where the graph G is K_r free. Given an instance (G, Π) where the optimal assignment violates ϵ-fraction of the edge constrains, Alev and Lau used an LP-based approach to efficiently find an assignment that violates at most $O(\sqrt{\epsilon} \cdot r)$-fraction. Specifically, in the rounding step of their LP, they used strong diameter separating decompositions with parameter $O(r^2)$. Using instead our decompositions from Corollary 18 with parameter $O(r)$ we obtain a quadratic improvement in the dependence on r.

▶ **Theorem 19.** *Consider an instance (G, Π) of the unique games problem, where the graph G is K_r free. Suppose that the optimal assignment violates at most an ϵ-fraction of the edge constrains. There is an efficient algorithm that find an assignment that violates at most an $O(\sqrt{\epsilon \cdot r})$-fraction.*

6.2 Spanner for Graphs with Moderate Doubling Dimension

Given a weighted graph $G = (V, E, w)$, a weighted graph $H = (V, E_H, w_H)$ is a *t-spanner* of G, if for every pair of vertices $v, u \in V$, $d_G(v, u) \leq d_H(v, u) \leq t \cdot d_X(v, u)$. If in addition H is a subgraph of G (that is $E_H \subseteq E$ and w_H agrees with w on E_H) then H is a *graph spanner*. The factor t is called the *stretch* of the spanner. The number of edges $|E_H|$ is the *sparsity* of the spanner. The weight of H is $w_H(H) = \sum_{e \in E_H} w_H(e)$ the sum of its edge weights. The *lightness* of H is $\frac{w_H(H)}{w(\text{MST}(G))}$ the ratio between the weight of the spanner to the wight of the MST of G. The tradeoff between stretch and sparsity/lightness of spanners had been the focus of an intensive research effort, and low stretch graph spanners were used in a plethora of applications.

There is an extensive study of spanners for doubling metrics. Recently, for an n-vertex graph with doubling dimension ddim, Borradaile, Le and Wulff-Nilsen [13] contrasted a graph spanner with $1+\epsilon$ stretch, $\epsilon^{-O(\text{ddim})}$ lightness and $n \cdot \epsilon^{-O(\text{ddim})}$ sparsity (improving [43, 28, 27]). This result is also asymptotically tight. Note that the dependency on ddim is exponential, which is unavoidable for small, $1 + \epsilon$ stretch. In cases where ddim is moderately large (say $\sqrt{\log n}$), it might be preferable to accept larger stretch in order to obtain reasonable lightness.

In a recent work, Filtser and Neiman [26], for every stretch parameter $t \geq 1$, constructed a spanner with stretch $O(t)$, lightness $O(2^{\frac{\text{ddim}}{t}} \cdot t \cdot \log^2 n)$ and $O(n \cdot 2^{\frac{\text{ddim}}{t}} \cdot \log n \cdot \log t)$ edges. However, this spanner was not a subgraph. Most applications require a graphic spanner. It is possible to transform [26] into a graphic spanner, but the number of edges becomes unbounded. The spanner construction of [26] is based on a variant of separating decompositions, where they used a weak-diameter version. If we replaced this with our strongly padded decompositions Corollary 9, and plug this into Theorem 3 from [26], we obtain a spanner with the same stretch to lightness ratio, but also with an additional sparsity guarantee.

▶ **Corollary 20.** *Let $G = (V, E, w)$ be an n vertex graph, with doubling dimension* ddim *and aspect ratio $\Lambda = \frac{\max_{e \in E} w(e)}{\min_{e \in E} w(e)}$. Then for every parameter $t > 1$ there is an graph-spanner of G with stretch $O(t)$, lightness $O(2^{\frac{\text{ddim}}{t}} \cdot t \cdot \log^2 n)$ and $O(n \cdot 2^{\frac{\text{ddim}}{t}} \cdot \log n \cdot \log \Lambda)$ edges.*

7 Conclusion and Open Problems

In this paper we closed the gap left in [5] between the padding parameters of strong and weak padded decompositions for minor free graphs. Our second contribution is tight strong padded decomposition scheme for graphs with doubling dimension ddim, which we also use to create sparse cover schemes. Some open questions remain:

1. Prove/disprove that K_r free graphs admit strong/weak decompositions with padding parameter $O(\log r)$, as conjectured by [5].
2. The question above is already open for the more restricted family of treewidth r graphs.
3. The δ parameter: [5] constructed weak $(O(r), \Omega(1))$-padded decomposition scheme, while we constructed strong $\left(O(r), \Omega(\frac{1}{r})\right)$-padded decomposition scheme. It will be nice to construct strong $(O(r), \Omega(1))$-padded decomposition scheme. Such a decomposition will imply a reacher spectrum of sparse covers (with $o(r)$ stretch).
4. Sparse covers for K_r free graphs: [6] constructed $(O(r^2), 2^r(r+1)!)$-sparse cover scheme, while [15] constructed $(4, f(r) \cdot \log n)$-sparse cover scheme. An interesting open question is to create additional sparse cover schemes. Specifically, our padded decompositions suggest that an $(O(r), g(r))$-sparse cover scheme for some function g independent of n, should be possible. Currently it is unclear how to construct such a cover. Optimally, we would like to construct $(O(1), g(r))$-sparse cover scheme.

References

1. I. Abraham and O. Neiman. Using Petal-Decompositions to Build a Low Stretch Spanning Tree. *SIAM Journal on Computing*, 48(2):227–248, 2019. doi:10.1137/17M1115575.
2. Ittai Abraham, Yair Bartal, and Ofer Neiman. Advances in metric embedding theory. *Advances in Mathematics*, 228(6):3026–3126, 2011. doi:10.1016/j.aim.2011.08.003.
3. Ittai Abraham, Shiri Chechik, Michael Elkin, Arnold Filtser, and Ofer Neiman. Ramsey Spanning Trees and their Applications. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1650–1664, 2018. doi:10.1137/1.9781611975031.108.
4. Ittai Abraham, Cyril Gavoille, Andrew V. Goldberg, and Dahlia Malkhi. Routing in Networks with Low Doubling Dimension. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS 2006), 4-7 July 2006, Lisboa, Portugal*, page 75, 2006. doi:10.1109/ICDCS.2006.72.
5. Ittai Abraham, Cyril Gavoille, Anupam Gupta, Ofer Neiman, and Kunal Talwar. Cops, robbers, and threatening skeletons: padded decomposition for minor-free graphs. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 79–88, 2014. doi:10.1145/2591796.2591849.
6. Ittai Abraham, Cyril Gavoille, Dahlia Malkhi, and Udi Wieder. Strong-Diameter Decompositions of Minor Free Graphs. *Theory Comput. Syst.*, 47(4):837–855, 2010. doi:10.1007/s00224-010-9283-6.
7. Vedat Levi Alev and Lap Chi Lau. Approximating Unique Games Using Low Diameter Graph Decomposition. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, pages 18:1–18:15, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.18.
8. Baruch Awerbuch. Complexity of Network Synchronization. *J. ACM*, 32(4):804–823, 1985. doi:10.1145/4221.4227.
9. Baruch Awerbuch and David Peleg. Sparse Partitions (Extended Abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 503–513, 1990. doi:10.1109/FSCS.1990.89571.
10. Yair Bartal. Probabilistic Approximations of Metric Spaces and Its Algorithmic Applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193, 1996. doi:10.1109/SFCS.1996.548477.
11. Yair Bartal, Lee-Ad Gottlieb, Tsvi Kopelowitz, Moshe Lewenstein, and Liam Roditty. Fast, precise and dynamic distance queries. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 840–853, 2011. doi:10.1137/1.9781611973082.66.
12. Guy E. Blelloch, Anupam Gupta, Ioannis Koutis, Gary L. Miller, Richard Peng, and Kanat Tangwongsan. Nearly-Linear Work Parallel SDD Solvers, Low-Diameter Decomposition, and Low-Stretch Subgraphs. *Theory Comput. Syst.*, 55(3):521–554, 2014. doi:10.1007/s00224-013-9444-5.
13. Glencora Borradaile, Hung Le, and Christian Wulff-Nilsen. Greedy spanners are optimal in doubling metrics. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2371–2379, 2019. doi:10.1137/1.9781611975482.145.
14. Costas Busch, Chinmoy Dutta, Jaikumar Radhakrishnan, Rajmohan Rajaraman, and Srinivasagopalan Srivathsan. Split and Join: Strong Partitions and Universal Steiner Trees for Graphs. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 81–90, 2012. doi:10.1109/FOCS.2012.45.
15. Costas Busch, Ryan LaFortune, and Srikanta Tirthapura. Sparse Covers for Planar Graphs and Graphs that Exclude a Fixed Minor. *Algorithmica*, 69(3):658–684, 2014. doi:10.1007/s00453-013-9757-4.

16 Gruia Călinescu, Howard J. Karloff, and Yuval Rabani. Approximation Algorithms for the 0-Extension Problem. *SIAM J. Comput.*, 34(2):358–372, 2004. doi:10.1137/S0097539701395978.

17 Michael Elkin, Yuval Emek, Daniel A. Spielman, and Shang-Hua Teng. Lower-Stretch Spanning Trees. *SIAM J. Comput.*, 38(2):608–628, 2008. doi:10.1137/050641661.

18 Michael Elkin and Ofer Neiman. Efficient Algorithms for Constructing Very Sparse Spanners and Emulators. *ACM Trans. Algorithms*, 15(1):4:1–4:29, November 2018. doi:10.1145/3274651.

19 Michael Elkin, Ofer Neiman, and Christian Wulff-Nilsen. Space-efficient path-reporting approximate distance oracles. *Theor. Comput. Sci.*, 651:1–10, 2016. doi:10.1016/j.tcs.2016.07.038.

20 Michael Elkin and Seth Pettie. A Linear-Size Logarithmic Stretch Path-Reporting Distance Oracle for General Graphs. *ACM Trans. Algorithms*, 12(4):50:1–50:31, 2016. doi:10.1145/2888397.

21 Matthias Englert, Anupam Gupta, Robert Krauthgamer, Harald Räcke, Inbal Talgam-Cohen, and Kunal Talwar. Vertex Sparsifiers: New Results from Old Techniques. *SIAM J. Comput.*, 43(4):1239–1262, 2014. doi:10.1137/130908440.

22 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004. doi:10.1016/j.jcss.2004.04.011.

23 Jittat Fakcharoenphol and Kunal Talwar. An Improved Decomposition Theorem for Graphs Excluding a Fixed Minor. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques, 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2003 and 7th International Workshop on Randomization and Approximation Techniques in Computer Science, RANDOM 2003, Princeton, NJ, USA, August 24-26, 2003, Proceedings*, pages 36–46, 2003. doi:10.1007/978-3-540-45198-3_4.

24 Uriel Feige, MohammadTaghi Hajiaghayi, and James R. Lee. Improved Approximation Algorithms for Minimum Weight Vertex Separators. *SIAM J. Comput.*, 38(2):629–657, 2008. doi:10.1137/05064299X.

25 Arnold Filtser, Robert Krauthgamer, and Ohad Trabelsi. Relaxed Voronoi: A Simple Framework for Terminal-Clustering Problems. In *2nd Symposium on Simplicity in Algorithms, SOSA@SODA 2019, January 8-9, 2019 - San Diego, CA, USA*, pages 10:1–10:14, 2019. doi:10.4230/OASIcs.SOSA.2019.10.

26 Arnold Filtser and Ofer Neiman. Light Spanners for High Dimensional Norms via Stochastic Decompositions. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, pages 29:1–29:15, 2018. doi:10.4230/LIPIcs.ESA.2018.29.

27 Arnold Filtser and Shay Solomon. The Greedy Spanner is Existentially Optimal. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing, PODC 2016, Chicago, IL, USA, July 25-28, 2016*, pages 9–17, 2016. doi:10.1145/2933057.2933114.

28 Lee-Ad Gottlieb. A Light Metric Spanner. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 759–772, 2015. doi:10.1109/FOCS.2015.52.

29 Anupam Gupta, Robert Krauthgamer, and James R. Lee. Bounded Geometries, Fractals, and Low-Distortion Embeddings. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 534–543, 2003. doi:10.1109/SFCS.2003.1238226.

30 David S. Johnson. The NP-Completeness Column: An Ongoing Guide. *J. Algorithms*, 8(2):285–303, 1987. doi:10.1016/0196-6774(87)90043-5.

31 Lior Kamma and Robert Krauthgamer. Metric Decompositions of Path-Separable Graphs. *Algorithmica*, 79(3):645–653, 2017. doi:10.1007/s00453-016-0213-0.

32 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 767–775, 2002. doi:10.1145/509907.510017.

33 Philip N. Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*, pages 682–690, 1993. doi:10.1145/167088.167261.

34 Robert Krauthgamer, James R. Lee, Manor Mendel, and Assaf Naor. Measured Descent: A New Embedding Method for Finite Metrics. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 434–443, 2004. doi:10.1109/FOCS.2004.41.

35 James R. Lee and Anastasios Sidiropoulos. Genus and the Geometry of the Cut Graph. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 193–201, 2010. doi:10.1137/1.9781611973075.18.

36 Nathan Linial and Michael E. Saks. Low diameter graph decompositions. *Combinatorica*, 13(4):441–454, 1993. doi:10.1007/BF01303516.

37 Jiri Matoušek. *Lectures on discrete geometry*. Springer-Verlag, New York, 2002. doi:10.1007/978-1-4613-0039-7.

38 Gary L. Miller, Richard Peng, Adrian Vladu, and Shen Chen Xu. Improved Parallel Algorithms for Spanners and Hopsets. In *Proceedings of the 27th ACM on Symposium on Parallelism in Algorithms and Architectures, SPAA 2015, Portland, OR, USA, June 13-15, 2015*, pages 192–201, 2015. doi:10.1145/2755573.2755574.

39 Robin A. Moser and Gábor Tardos. A constructive proof of the general lovász local lemma. *J. ACM*, 57(2):11:1–11:15, 2010. doi:10.1145/1667053.1667060.

40 Yuri Rabinovich. On average distortion of embedding metrics into the line and into L1. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 456–462, 2003. doi:10.1145/780542.780609.

41 Satish Rao. Small distortion and volume preserving embeddings for planar and Euclidean metrics. In *Proceedings of the fifteenth annual symposium on Computational geometry*, SCG '99, pages 300–306, New York, NY, USA, 1999. ACM. doi:10.1145/304893.304983.

42 Neil Robertson and Paul D. Seymour. Graph Minors. XVI. Excluding a non-planar graph. *J. Comb. Theory, Ser. B*, 89(1):43–76, 2003. doi:10.1016/S0095-8956(03)00042-X.

43 Michiel H. M. Smid. The Weak Gap Property in Metric Spaces of Bounded Doubling Dimension. In *Efficient Algorithms, Essays Dedicated to Kurt Mehlhorn on the Occasion of His 60th Birthday*, pages 275–289, 2009. doi:10.1007/978-3-642-03456-5_19.

A Path Reporting Distance Oracles

Given a weighted graph $G = (V, E, w)$, a *distance oracle* is a data structure that supports distance queries between pairs $u, v \in V$. The distance oracle has stretch t, if for every query $\{u, v\}$, the estimated distance $\mathrm{est}(u, v)$ is within $d_G(u, v)$ and $t \cdot d_G(u, v)$. The studied objects are stretch, size the query time. An additional requirement that been recently studied [20] is *path reporting*: in addition to distance estimation, the distance oracle should also return a path of the promised length. In this case, we say that distance oracle has query time q, if answering a query when the reported path has m edges, takes $q + O(m)$ time.

Path reporting distance oracles were studied for general graphs [20, 19]. For the special case of graphs excluding K_r as a minor, Elkin, Neiman and Wulff-Nilsen [19] constructed a path reporting distance oracles with stretch $O(r^2)$, space $O(n \cdot \log \Lambda \cdot \log n)$ and query time $O(\log \log \Lambda)$, where $\Lambda = \frac{\max_{u,v} d_G(u,v)}{\min_{u,v} d_G(u,v)}$ is the aspect ratio. For this construction they used the strongly padded decomposition of [5] (in fact strong-diameter sparse covers). Implicitly, given

a graph G that admits a strong (β, s)-sparse cover scheme, [19] constructs a path reporting distance oracle with stretch β, size $O(n \cdot s \cdot \log_\beta \Lambda)$ and query time $O(\log\log\Lambda)$. Following similar arguments to [19] [8], our padded decompositions from Theorem 15 implies that every K_r free graph admits a strong $(O(r), O(\log n))$-sparse cover scheme. We conclude:

▶ **Corollary 21.** *Given an n-vertex weighted graph $G = (V, E, w)$ which excludes K_r as a minor, with aspect ratio Λ , there is a path reporting distance oracle with stretch $O(r)$, space $O(n \cdot \log_r \Lambda \cdot \log n)$ and query time $O(\log\log\Lambda)$.*

It is interesting to mention that Busch et al. [15] constructed a $(4, O(f(r)\log n))$ sparse cover scheme for K_r free graphs, where $f(r)$ is an extremely large function of r. Using the framework of [19], it will imply a path reporting distance oracle with stretch 4, space $O(n \cdot \log\Lambda \cdot f(r))$ and query time $O(\log\log\Lambda)$. The value of $f(r)$ is larger that a square of the constant from the Robertson and Seymour structure theorem [42]. In particular, an estimation by Johnson [30] implies that $f(r)$ is larger than $2 \Uparrow (2 \Uparrow (2 \Uparrow (r/2)) + 3)$ [9]. This value is so big, that the [15]-based oracle is completely impractical already for quite small values of r.

For the case of graphs with doubling dimension ddim, we constructed the first strong-diameter sparse covers. Plugging our Theorem 10 into the framework of [19], we obtain the first path reporting distance oracle for doubling graphs. The only relevant previous distance oracle for doubling metrics is by Bartal et al. [11]. However, they focused on the $1+\epsilon$-stretch regime, where inherently the oracle size has exponential dependency on ddim.

▶ **Corollary 22.** *Given an n-vertex weighted graph $G = (V, E, w)$ with doubling dimension* ddim *and aspect ratio Λ, for every parameter $t \geq \Omega(1)$, there is a path reporting distance oracle with stretch $O(t)$, space $O(n \cdot 2^{\mathrm{ddim}/t} \cdot \mathrm{ddim} \cdot \log\Lambda)$ [10] and query time $O(\log\log\Lambda)$.*

In particular, there is a path reporting distance oracle with stretch $O(\mathrm{ddim})$, space $O(n \cdot \mathrm{ddim} \cdot \log\Lambda)$ and query time $O(\log\log\Lambda)$.

B Proof of Theorem 4 using Cones

We will prove a Theorem 4 with slightly weaker parameters. Specifically we will construct a strongly $\left(O(\ln\tau), \frac{1}{32}, 4\Delta\right)$-padded decomposition.

Order the vertices in $N = \{x_1, x_2, \dots\}$ arbitrarily. For every center $x_i \in N$, sample $\delta_i \in [0,1]$ according to $\mathsf{Texp}(\lambda)$ truncated exponential distribution with parameter $\lambda = 2 + 2\ln\tau$. Set $R_i = \delta_i \cdot \Delta \in [0, \Delta]$. The clustering algorithm is executed in an iterative manner. We denote by S the set of unclustered vertices, which are also called *active* vertex. Initially $S = V$. As long as there is an active center $S \cap N \neq \emptyset$, pick active center $x_i \in N$ with minimal index and create the cluster

$$C_i = \left\{v \in S \mid d_{G[S]}(v, x_i) - d_{G[S]}(v, N \cap S) \leq R_i\right\} .$$

This procedure halts when all the centers are clustered. See Algorithm 2 for pseudo code.

▷ Claim 23. For a vertex $v \in G$ let $x_v \in N$ be the closest center, and let $\mathcal{I}_v$ be the shortest path from v to x_v. Then if some vertex of $\mathcal{I}_v$ is clustered, so do v.

[8] Taking $O(\log n)$ independent copies and using union bound,

[9] $2 \Uparrow t$ denotes an exponential tower of t 2's. That is $2 \Uparrow 0 = 1$ and $2 \Uparrow t = 2^{2 \Uparrow (t-1)}$.

[10] This is assuming $\Lambda > \log t$, otherwise simply using an arbitrary shortest path tree will provide a distance oracle with stretch $O(\log t)$.

Algorithm 2 `Partition-To-Cones`$(G=(V,E,w),N,\Delta,\tau)$.

```
Let S ← V, 𝒮 ← ∅.
Order the vertices in N = x_1, x_2, ... arbitrarily.
for i = 1 to |N| do
  if x_i ∈ S then
    Sample R_i independently from the distribution Texp(2 + 2 ln τ).
    C_i ← ∅
    for all v ∈ S do
      if d_{G[S]}(v, x_i) − d_{G[S]}(v, N ∩ S) ≤ R_i then
        Add v to C_i.
      end if
    end for
    S ← S \ C_i
    Add C_i to 𝒮.
  end if
end for
return 𝒮.
```

Proof. Suppose that $u \in \mathcal{I}_v$ joined the cluster of x_j while the set of active vertices were S (in particular $\mathcal{I}_v \subseteq S$). Then

$$\begin{aligned} d_{G[S]}(v,x_j) &\leq d_{G[S]}(v,u) + d_{G[S]}(u,x_j) \\ &\leq d_{G[S]}(v,u) + d_{G[S]}(u,x_v) + R_j = d_{G[S]}(v,x_v) + R_j \ . \end{aligned}$$ ◁

▶ **Corollary 24.** *All vertices are clustered.*

Proof. The vertex v will be clustered at the first time some vertex from $\mathcal{I}_v$ is clustered. As x_v itself necessarily clustered, the corollary follows. ◀

▷ Claim 25. Every cluster has strong diameter 4Δ.

Proof. Suppose that at the time we constructed C_i the set of active vertices was S. Let $v \in C_i$, and $x_v \in N$ the closest center to v. As v joined C_i and was active, all the vertices in $\mathcal{I}_v$ the shortest path from v to x_v were active as well. Therefore,

$$d_{G[S]}(v,x_i) \leq d_{G[S]}(v,x_v) + R_i \leq 2\Delta \ .$$

Let $\mathcal{I}$ be the shortest path from v to x_i in $G[S]$. We argue that all the vertices on $\mathcal{I}$ also joined C_i. Indeed, consider $u \in \mathcal{I}$. Then

$$\begin{aligned} d_{G[S]}(u,x_i) &= d_{G[S]}(v,x_i) - d_{G[S]}(v,u) \\ &\leq d_{G[S]}(v, N\cap S) + R_i - d_{G[S]}(v,u) \leq d_{G[S]}(u, N\cap S) + R_i \ . \end{aligned}$$

It follows that $d_{G[C_i]}(v,x_i) \leq 2\Delta$. In particular C_i has strong diameter bounded by 4Δ. ◁

Consider some vertex $v \in V$ and parameter $\gamma \leq \frac{1}{8}$. We will argue that the ball $B = B_G(v, \gamma\Delta)$ is fully contained in $P(v)$ with probability at least $2^{-O(\gamma \log \tau)}$, in other words that v is $\frac{\gamma}{4}$-padded. Let N_v be the set of centers x_i for which there is a non zero probability that C_i intersects B. Following the calculation in Claim 25, each vertex joins the cluster of a center at distance at most 2Δ. By triangle inequality, all the centers in N_v are at distance at most $(2+\gamma)\Delta \leq 3\Delta$ from v. In particular $|N_v| \leq \tau$.

For x_i, denote by $\mathcal{F}_i$ the event that some vertex of B joins the cluster C_i for the first time. I.e. $B \cap C_i \neq \emptyset$ and for all $j < i$, $B \cap C_j = \emptyset$. Denote by $\mathcal{C}_i$ the event that $\mathcal{F}_i$ occurred and B is cut by C_i. Note that for every $x_i \notin N_v$, $\mathcal{F}_i = \mathcal{C}_i = \emptyset$. To prove the theorem, it is enough to show that $\Pr[\cup_i \mathcal{C}_i] \leq 1 - e^{-O(\gamma \cdot \lambda)}$. Set $\alpha = e^{-4\gamma \cdot \lambda}$.

▷ Claim 26. For every i, $\Pr[\mathcal{C}_i] \leq (1-\alpha)\left(\Pr[\mathcal{F}_i] + \frac{1}{e^\lambda - 1}\right)$.

Proof. Let $S \subset V$ be the set of active vertices at the beginning of round i. If $B \cup \{x_i\} \not\subseteq S$ then $\Pr[\mathcal{C}_i] = 0$ and we are done. Let ρ_S be the minimal value of δ_i such that if $\delta_i \geq \rho_S$, some vertex of B joins C_i. Formally $\rho_S = \frac{1}{\Delta} \cdot \min_{u \in B}\{d_{G[S]}(u, x_i) - d_{G[S]}(u, N \cap S)\}$. If $\rho_S > 1$, then $\Pr[\mathcal{C}_i] = 0$ and we are done, thus we assume $\rho_S \leq 1$. Conditioning on S, it holds that

$$\Pr[\mathcal{F}_i \mid S] = \Pr[\delta_i \geq \rho_S] = \int_{\rho_S}^{1} \frac{\lambda \cdot e^{-\lambda y}}{1 - e^{-\lambda}} dy = \frac{e^{-\rho_S \cdot \lambda} - e^{-\lambda}}{1 - e^{-\lambda}}$$

Let $v' \in B$ some vertex that joins C_i if $\delta_i = \rho_S$. Then for every $u \in B$ it holds that

$$\begin{aligned} d_{G[S]}(u, x_i) &\leq d_{G[S]}(v', x_i) + 2\gamma\Delta \leq d_{G[S]}(v', N \cap S) + \rho_S \cdot \Delta + 2\gamma\Delta \\ &\leq d_{G[S]}(u, N \cap S) + (\rho_S + 4\gamma) \cdot \Delta \ . \end{aligned}$$

Therefore, if $\delta_i \geq \rho_S + 4\gamma$, the entire ball B will be contained in C_i. We conclude,

$$\begin{aligned} \Pr[\mathcal{C}_i \mid S] &\leq \Pr[\rho_S \leq \delta_i < \rho_S + 4\gamma] \\ &= \int_{\rho_S}^{\max\{1, \rho_S + 4\gamma\}} \frac{\lambda \cdot e^{-\lambda y}}{1 - e^{-\lambda}} dy \\ &\leq \frac{e^{-\rho_S \cdot \lambda} - e^{-(\rho_S + 4\gamma) \cdot \lambda}}{1 - e^{-\lambda}} \\ &= \left(1 - e^{-4\gamma \cdot \lambda}\right) \cdot \frac{e^{-\rho_S \cdot \lambda}}{1 - e^{-\lambda}} \\ &= (1 - \alpha) \cdot \left(\Pr[\mathcal{F}_i \mid S] + \frac{1}{e^\lambda - 1}\right) \ . \end{aligned}$$

By the low of total probability, we can remove the conditioning on S. Denote by f the density function of the distribution over all possible choices of S. It holds that,

$$\begin{aligned} \Pr[\mathcal{C}_i] &= \int_S \Pr[\mathcal{C}_i \mid S] \cdot f(S)\ dS \\ &\leq (1 - \alpha) \cdot \int_S \left(\Pr[\mathcal{F}_i \mid S] + \frac{1}{e^\lambda - 1}\right) \cdot f(S)\ dS \\ &= (1 - \alpha) \cdot \left(\Pr[\mathcal{F}_i] + \frac{1}{e^\lambda - 1}\right) \ . \end{aligned}$$

◁

We bound the probability that the ball B is cut,

$$\begin{aligned} \Pr[\cup_i \mathcal{C}_i] = \sum_{x_i \in N_v} \Pr[\mathcal{C}_i] &\leq (1 - \alpha) \cdot \sum_{x_i \in N_v} \left(\Pr[\mathcal{F}_i] + \frac{1}{e^\lambda - 1}\right) \\ &\leq \left(1 - e^{-4\gamma \cdot \lambda}\right) \cdot \left(1 + \frac{\tau}{e^\lambda - 1}\right) \\ &\leq \left(1 - e^{-4\gamma \cdot \lambda}\right) \cdot \left(1 + e^{-4\gamma \cdot \lambda}\right) = 1 - e^{-8\gamma \cdot \lambda} \ , \end{aligned}$$

where the last inequality follows as $e^{-4\gamma\lambda} = \frac{e^{-4\gamma\lambda}\left(e^\lambda - 1\right)}{e^\lambda - 1} \geq \frac{e^{-4\gamma\lambda} \cdot e^{\lambda - 1}}{e^\lambda - 1} \geq \frac{e^{\frac{\lambda}{2} - 1}}{e^\lambda - 1} = \frac{\tau}{e^\lambda - 1}$.

Max-Min Greedy Matching

Alon Eden
Tel Aviv University, Israel
alonarden@gmail.com

Uriel Feige
Weizmann Institute of Science, Rehovot, Israel
uriel.feige@weizmann.ac.il

Michal Feldman
Tel Aviv University, Israel
Microsoft Research, Herzlyia, Israel
michal.feldman@cs.tau.ac.il

Abstract

A bipartite graph $G(U, V; E)$ that admits a perfect matching is given. One player imposes a permutation π over V, the other player imposes a permutation σ over U. In the greedy matching algorithm, vertices of U arrive in order σ and each vertex is matched to the highest (under π) yet unmatched neighbor in V (or left unmatched, if all its neighbors are already matched). The obtained matching is maximal, thus matches at least a half of the vertices. The max-min greedy matching problem asks: suppose the first (max) player reveals π, and the second (min) player responds with the worst possible σ for π, does there exist a permutation π ensuring to match strictly more than a half of the vertices? Can such a permutation be computed in polynomial time?

The main result of this paper is an affirmative answer for these questions: we show that there exists a polytime algorithm to compute π for which for every σ at least $\rho > 0.51$ fraction of the vertices of V are matched. We provide additional lower and upper bounds for special families of graphs, including regular and Hamiltonian graphs. Our solution solves an open problem regarding the welfare guarantees attainable by pricing in sequential markets with binary unit-demand valuations.

2012 ACM Subject Classification Theory of computation → Computational pricing and auctions; Mathematics of computing → Matchings and factors

Keywords and phrases Online matching, Pricing mechanism, Markets

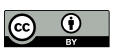
Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.7

Category APPROX

Related Version `https://arxiv.org/pdf/1803.05501.pdf`

Funding *Alon Eden*: Partially supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement number 337122, the Israel Science Foundation (grant number 317/17), and the Blavatnik family foundation.
Uriel Feige: Partially supported by the Israel Science Foundation (grant No. 1388/16).
Michal Feldman: Partially supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement number 337122, the Israel Science Foundation (grant number 317/17), and the Blavatnik family foundation.

Acknowledgements A substantial part of this work was conducted in Microsoft Research, Herzlyia. We are grateful to Amos Fiat and Sella Nevo for numerous discussions that contributed significantly to the ideas presented in this paper. We also thank Robert Kleinberg for helpful discussions.



© Alon Eden, Uriel Feige, and Michal Feldman;
licensed under Creative Commons License CC-BY



1 Introduction

Given a bipartite graph $G(U, V; E)$, where U and V are the sets of vertices and $E \in U \times V$ is the set of edges, a matching $M \subset E$ is a set of edges such that every vertex is incident with at most one edge of M. For simplicity of notation, for every n we shall only consider the following class of bipartite graphs, that we shall refer to as G_n. For every $G(U, V; E) \in G_n$ it holds that $|U| = |V| = n$ and that E contains a matching of size n (and hence G has a perfect matching). All results that we will state for G_n hold without change for all bipartite graphs that have a matching of size n (and arbitrarily many vertices).

Karp, Vazirani and Vazirani [12] introduced the *online bipartite matching* problem. This setting can be viewed as a game between two players: a maximizing player who wishes the resulting matching to be as large as possible, and a minimizing player who wishes the matching to be as small as possible. First, the minimizing player chooses $G(U, V; E)$ in private (without the maximizing player seeing E), subject to $G \in G_n$. Thereafter, the structure of G is revealed to the maximizing player in n steps, where at step j (for $1 \leq j \leq n$) the set $N(u_j) \subset V$ of vertices adjacent to u_j is revealed. At every step j, upon seeing $N(u_j)$ (and based on all edges previously seen and all previous matching decisions made), the maximizing player needs to irrevocably either match u_j to a currently unmatched vertex in $N(u_j)$, or leave u_j unmatched.

There is much recent interest in the online bipartite matching problem and variations and generalizations of it, as such models have applications for allocation problems in certain economic settings, in which buyers (vertices of U) arrive online and are interested in purchasing various items (vertices of V). A prominent example of such an application is online advertising; for more details, see for example the survey by Mehta [17]. The new problems are both theoretically elegant and practically relevant.

Max-Min Greedy Matching

We study a setting related to online bipartite matching, that we call *Max-Min Greedy matching*. Our setting is also a game between a maximizing player and a minimizing player. The bipartite graph $G(U, V; E) \in G_n$ is given upfront. Upon seeing G the maximizing player chooses a permutation π over V. Upon seeing G and π, the minimizing player chooses a permutation σ over U. The combination of G, π and σ define a unique matching $M_G[\sigma, \pi]$ that we refer to as the greedy matching. It is the matching produced by the greedy matching algorithm in which vertices of U arrive in order σ and each vertex $u \in U$ is matched to the highest (under π) yet unmatched $v \in N(u)$ (or left unmatched, if all $N(u)$ is already matched).

The matching $M_G[\sigma, \pi]$ has several additional equivalent definitions. For example, $M_G[\sigma, \pi]$ is the matching produced by the greedy matching algorithm in which vertices of V arrive in order π and each vertex $v \in V$ is matched to the highest (earliest in arrival order under σ) yet unmatched $u \in N(v)$ (or left unmatched, if all $N(v)$ is already matched). Also, $M_G[\sigma, \pi]$ is the unique stable matching in G (in the sense of [9]), if the preference order of every vertex $u \in U$ over its neighbors is consistent with π, and the preference order of every vertex $v \in V$ over its neighbors is consistent with σ.

Let $\rho[G] = \frac{1}{n} \max_\pi \min_\sigma [|M_G[\sigma, \pi]|]$, and let $\rho = \min_{G \in G_n} [\rho[G]]$. It is easy to see that $\rho \geq \frac{1}{2}$. In fact, to ensure a matching of size $n/2$, the max player need not work hard. Since every greedy matching is a *maximal* matching, for *every* permutation π the obtained matching is of size at least $n/2$. The question we study in this work is whether the max player can ensure a matching of size strictly greater than $n/2$; that is, whether ρ is strictly greater than $\frac{1}{2}$.

For an upper bound on ρ, it was observed by Cohen Addad et al. [4] that $\rho \leq 2/3$. To show this, they observe that in the 6-cycle graph, depicted in Figure 1, no permutation π can guarantee to match more than two vertices in the worst case. Indeed, suppose (without loss of generality) that $\pi = (v_3, v_2, v_1)$. For $\sigma = (u_1, u_3, u_2)$, u_1 is matched to v_2, u_3 is matched to v_3, and u_2 is left unmatched, resulting in a matching of size 2.

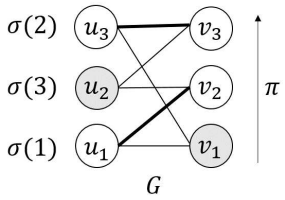


Figure 1 For every permutation π there exists a permutation σ that matches only 2 of the 3 vertices. Thick edges are in the matching; gray vertices are unmatched.

1.1 Our Results

Our main result resolves the open problem in the affirmative:

▶ **Theorem** (main theorem). *It holds that* $\rho \geq \frac{1}{2} + \frac{1}{86} > 0.51$. *Moreover, there is a polynomial time algorithm that given* $G(U, V; E)$ *produces a permutation* π *over* V *satisfying the above bound.*

The significance of this result is that 1/2 is not the optimal answer. We believe that further improvements are possible. In fact, for Hamiltonian graphs we show that $\rho \geq \frac{5}{9}$ (see Section 6).

The proof method is quite involved; it is natural to ask whether simpler approaches may work. In what follows we specify three natural attempts that all fail.

Failed attempt 1: random permutation

A first attempt would be to check whether a random permutation π obtains the desired result (in expectation). The performance of a random permutation is interesting for an additional reason: it is the performance in scenarios where the graph structure is unknown to the designer. Unfortunately, there exists a bipartite graph G, even one where all vertices have high degree, for which a random permutation matches no more than a fraction $1/2 + o(1)$ of the vertices (see Section 4).

In contrast, we show that in the case of Hamiltonian graphs a random permutation guarantees a competitive ratio strictly greater than 1/2 (Section 4). A similar proof approach applies to regular graphs as well.

▶ **Theorem** (random permutation). *There is some* $\rho_0 > \frac{1}{2}$ *such that for every Hamiltonian graph* $G \in G_n$, *regardless of* n, *a random permutation* π *results in* $\rho \geq \rho_0$. *Similarly, there is some constant* $\rho_0 > \frac{1}{2}$ *such that for every* d*-regular graph* G, *regardless of* d, n, *a random permutation* π *results in* $\rho \geq \rho_0$.

Failed attempt 2: iterative upgrading

A second attempt would be to iteratively "upgrade" unmatched vertices, with the hope that the iterative process will reach a state where many vertices will be matched. That is, in every iteration consider the worst order σ for the current permutation π and move all unmatched vertices (in the matching induced by (π, σ)) to be ranked higher in π. This algorithm is similar to the *k-pass Category Advice* algorithm of [1], but with the difference that in [1] σ remains unchanged throughout the k iterations. In [1] it was shown that in their setting, the fraction of matched vertices approaches $\frac{2}{1+\sqrt{5}} \simeq 0.619$ as k grows. In contrast, in Appendix B we show that in our setting this process can go on for $\log n$ iterations before reaching a permutation that matches more than a half of the vertices. This fact gives some indication that establishing a proof using this operator might be difficult.

Failed attempt 3: degree-based ranking

A third attempt would be to give preference to vertices with lower degrees, as they would have fewer opportunities to be matched to incoming vertices of U. Consider a graph with multiple copies of the subgraph $(u_1, v_1), (u_2, v_2), (u_1, v_2)$ along with two additional vertices u_a, u_b (and their partners v_a, v_b). If we connect all vertices of type v_1 to u_a and u_b, we get that their degree is 3, while the degree of vertices of type v_2 is 2. If π is chosen according to the degree, vertices of type v_2 will be ranked higher than vertices of type v_1. In this case, if σ orders the vertices of type u_1 first, they will be matched to vertices of type v_2, leaving the vertices of type v_1 unmatched. The resulting matching will therefore be of size $(1/2 + o(1))n$.

Why is this model interesting mathematically?

The setting of max-min greedy matching is easy to state. The problem of getting a ratio better than half turns out to be deceptively difficult. As discussed above, several natural approaches fail to achieve this. The problem remained open for quite some time, despite attempts to solve it. Indeed, the solution that we find is not simple; it involves taking the best of four algorithms. However, these algorithms are not unrelated. They all share a unifying theme that involves a clean combinatorial property, referred to as a maximal path cover (see Section 2). This theme enabled us to break the barrier of half, but interesting problems remain open, such as whether the bound of 2/3 can be achieved. We hope that the progress made in this work will motivate and enable further improvement in this interesting problem.

1.2 Additional Results

We further establish lower and upper bounds for regular graphs.

▶ **Theorem** (regular graphs). *For d-regulars bipartite graphs, $\rho \geq \frac{5}{9} - O(\frac{1}{\sqrt{d}})$. On the other hand, for every integer $d \geq 1$, there is a regular graph G_d of even degree $2d$ such that $\rho(G_d) \leq \frac{8}{9}$.*

An additional natural problem is to find the best permutation π, given a graph G. We suspect that this is a difficult computational problem. However, the special case of determining whether there is a *perfect* π (a permutation on V that for every permutation σ leads to a perfect matching) does have a polynomial time algorithm (proof appears in Section 5).

▶ **Proposition 1.** *There is a polynomial time algorithm that given a graph $G \in G_n$ determines whether G has a* perfect *π, and if so, outputs a perfect π.*

1.3 Application to Resource Allocation and Pricing

Various problems related to online bipartite matching are closely related to problems that attract attention in the algorithms community. Gaining better understanding of the max-min greedy matching problem sheds light on more general problems, some of which are still open. In what follows we elaborate on an application to a pricing problem.

Feldman et al. [8] study the design of pricing mechanisms for allocation of items in markets. The basic setting is a matching market, where v_{ij} is the value of agent i for item j, and every agent can receive at most a single item. The seller assigns prices to items, and agents arrive in an adversarial order (after observing the prices), each purchasing an (arbitrary) item that maximizes their utility (defined as value minus price). It is shown that, given a weighted bipartite graph (with agents on one side, items of the other side, and weight v_{ij} for the edge between agent i and item j), one can set item prices that guarantee at least half of the optimal welfare for any arrival order. The last result holds in much more general settings, namely settings where buyers have submodular valuations over bundles of items [1], and even in a Bayesain settings, where the seller knows only the (product) distribution from which agent valuations are drawn, but not their realizations.

In the Bayesian setting no item prices can guarantee better than half of the optimal welfare in the worst case. A natural question is whether this ratio can be improved in scenarios where the designer knows the realized values of the buyers from the outset [2]. Concretely, do there exist item prices that guarantee strictly more than half the optimal welfare, for any arrival order σ? Not only has this question been open for general combinatorial auctions with submodular valuations, it has been open even for unit-demand buyers, and even if all individual values are in $\{0, 1\}$ (henceforth referred to as binary unit demand valuations). In the latter setting, pricing is equivalent to imposing a permutation over the items, hence the max-min greedy matching is a precise formulation for the pricing problem in binary unit-demand settings.

An equivalent scenario is one where in each step the "items player" offers an item, and the "buyers player", upon seeing the item, allocates the item to one of the buyers who wants the item (if there is any), and that buyer leaves. The items player is non-adaptive (plays blindfolded, without seeing which buyers remain[3]). The size of the matching that can be guaranteed by the items player is equivalent to the max-min greedy matching problem.

Yet an additional equivalent formulation of the problem is one where the permutation π is imposed over the buyers rather than over the items. The buyers then arrive in the order of π, each taking an arbitrary item she wants. One can verify that the size of the matching that can be guaranteed by an ordering over the buyers is equivalent to the max-min greedy matching problem.

[1] A valuation is said to be submodular if for every two sets S, T, $v(S) + v(T) \geq v(S \cup T) + v(S \cap T)$,.

[2] The full information assumption is sensible in repeated markets or in markets where the stakes are high and the designer may invest in learning the demand in the market before setting prices.

[3] We note that when the items player is adaptive (chooses the next item based on what happened in the past), the items player can ensure a perfect matching. This is done as follows: in each step, find a minimal tight set of items, and offer an arbitrary item from that set. Here, a set of items is tight if the number of buyers that want items in the set is equal to the size of the set.

1.4 Relation to Previous Work

Relation to online bipartite matching

The Max-Min Greedy Matching problem is a nonstandard version of online problems. In the standard online matching problem [12], the algorithm designer has control over the matching algorithm, but has no control over the arrival order of clients (vertices). Our setting can model a situation in which the designer (the maximizing player) has full control over the arrival order of clients (it knows which "items" in U each "client" in V wants, and it chooses π based on this knowledge), but no control over the matching algorithm (the minimizing player can choose the worst possible match in every step, effectively resulting in a permutation σ).

Karp et al. [12] introduced the *Ranking* algorithm which has a $1 - 1/e$ competitive ratio in the online bipartite matching setting. Translating this algorithm to our Max-Min Greedy setting, it amounts to simply selecting π at random, and then the minimizing player selects σ after seeing π. We show that there are bipartite graphs $G \in G_n$ for which with high probability over the random choice of π, there is a choice of σ resulting in $M_G[\sigma, \pi] \leq 1/2 + o(1)$. Karp et al. [12] also showed that no algorithm for online bipartite matching has a competitive ratio better than $1 - 1/e + o(1)$. This was shown by exhibiting a distribution over "difficult" graphs. Each graph in the support of this distribution has a unique perfect matching, and consequently (see Proposition 3), there is a permutation π in the Max-Min Greedy setting that ensures that all vertices are matched (regardless of σ). Hence neither the lower bounds nor the upper bounds known for the online matching model give useful bounds in the Max-Min Greedy model.

There are additional known results for online bipartite matching. For d-regular graphs, Cohen and Wajc [3] present a random algorithm that obtains $1-O(\sqrt{\log d}/\sqrt{d})$ in expectation, and a lower bound of $1 - O(1/\sqrt{d})$. This is in contrast to our Theorem 12 that shows that ρ is bounded away from 1 even when d is arbitrarily large. For general bipartite graphs, under random (rather than adversarial) arrival order, the deterministic greedy algorithm gives $1 - 1/e$, and no deterministic algorithm can obtain more than $3/4$ [10]. *Ranking* (which is a randomized algorithm) obtains at least 0.696 of the optimal matching [15] and at most 0.727 [11]. No random algorithm can obtain more than 0.823 [16].

Relation to pricing mechanisms

Our work is also related to the recent body of literature on pricing mechanisms. Motivated by the fact that in real-life situations one is often willing to trade optimality for simplicity, the study of simple mechanisms has gained a lot of interest in the literature on algorithmic mechanism design. One of the simplest forms of mechanisms is that of posted price mechanisms, where prices are associated with items and agents buy their most preferred bundles as they arrive. Pricing mechanisms have many advantages: they are simple, straightforward, and allow for asynchronous arrival and departure of buyers. Various forms of posted price mechanisms for welfare maximization have been proposed for various combinatorial settings [8, 5, 14, 6]. These mechanisms are divided along several axes, such as item vs. bundle pricing, static vs. dynamic pricing, and anonymous vs. personalized pricing. For any market with submodular valuations, one can obtain $1/2$ of the optimal welfare by static, anonymous item prices [8]. Until the present paper, no better results than $1/2$ were known even for markets with unit-demand valuations with $\{0, 1\}$ individual values. For a market with m identical items, there exists a pricing scheme that obtains at least $5/7 - 1/m$ of the optimal welfare for submodular valuations [6].

2 Proof of Main Result

The graph $G(U, V; E)$ with $|U| = |V| = n$ has a perfect matching M in which $u_i \in U$ is matched with $v_i \in V$ for every $1 \leq i \leq n$. For a given i, we refer to u_i and v_i as *partners* of each other. Given a set $S \subset V$, the set of neighbors of S is denoted as $N(S)$ (where necessarily $N(S) \subset U$). In this section we prove our main result.

▶ **Theorem 2.** *Given a bipartite graph $G(U, V; E)$ with a perfect matching $\{(u_i, v_i)\}$, there exists a permutation π that guarantees that the greedy matching will be of size at least $\frac{22}{43}n$, regardless of σ. Moreover, there is a polynomial time algorithm that chooses π with such a guarantee.*

Our proof approach is as follows. We shall first associate with G an auxiliary directed graph that we refer to as the spoiling graph $H(V, D)$. This notion by itself is not new – similar notions appeared also in previous related work. The new aspect related to the spoiling graph and the key to our approach is a notion of a *maximal path cover*. Given a maximal path cover of the spoiling graph (which as we show in Proposition 4, can be found in polynomial time), we partition the set V of vertices into four classes, depending on their roles in the maximal path cover. The classes are V_1 (singleton vertices), S (start vertices of paths), T (end vertices of paths), and I (intermediate vertices of paths). By considering several carefully chosen orders among the classes of vertices, and also of vertices within the classes, we obtain four possible candidate permutations for π, denoted $\pi_1, \pi_2, \pi_3, \pi_4$. We show that for every bipartite graph with a perfect matching, at least one of these permutations, if used as π, guarantees that the greedy matching will be of size at least $\frac{22}{43}n$, for every σ. Put in other words, if for each of $\{\pi_1, \pi_2, \pi_3, \pi_4\}$ there is a permutation over U for which the greedy matching is smaller than $\frac{22}{43}n$, this would imply (using properties listed in Lemma 5) that the path cover giving rise to these permutations was not maximal.

We now proceed to define the spoiling graph. Given $G(U, V; E)$, consider a directed graph $H(V, D)$ whose vertices are the set V, and whose set D of directed edges (arcs) is defined as follows: $(v_i, v_j) \in D$ iff $(u_j, v_i) \in E$. We refer to $H(V, D)$ as the *spoiling graph* for G, because arc $(v_i, v_j) \in D$ allows for the possibility that edge $(u_j, v_i) \in E$ is chosen into a matching M' in G, spoiling for v_j the possibility (offered by M) of being matched to u_j. Note that this spoiling effect may materialize in a (σ, π) matching only if v_i is ranked higher than v_j in π. Hence the spoiling graph conveys information that may be relevant to the choice of π.

As an example of the information that can be derived from the spoiling graph, consider the following proposition (whose proof can be also obtained as a special case of a result given in [4] and [13] for the more general case of Gross Substitutes valuations).

▶ **Proposition 3.** *If G has a unique perfect matching, then $\rho(G) = 1$.*

Proof. Let $u_i \in U$ and $v_i \in V$ be partners in the unique perfect matching M of G. We claim that the spoiling graph H of G is a *directed acyclic graph* (DAG). Suppose toward contradiction that H contains a simple directed cycle $v_{i_1}, v_{i_2}, \ldots, v_{i_\ell}, v_{i_1}$. This directed cycle corresponds to the cycle $u_{i_1}, v_{i_1}, u_{i_2}, v_{i_2}, \ldots, u_{i_\ell}, v_{i_\ell}, u_{i_1}$ in G. But removing the edges (u_{i_j}, v_{i_j}), $1 \leq j \leq \ell$ from M and adding the edges $(u_{i_j}, v_{i_{j+1}})$ to M (where $\ell + 1 = 1$) yields a different perfect matching, contradicting the uniqueness of M.

Since H is a DAG, we can topologically sort its vertices and choose a permutation π such that earlier vertices in the topological order have a lower rank in π. This ensures that for every directed edge (v, w) in H, v's partner will never prefer w over v. Thus, every vertex chooses its partner in M upon arrival. ◀

We now proceed to define the notion of a maximal path cover. A *directed path* P (whose length is denoted by $|P|$) in H is a sequence of $|P|$ vertices (say, $v_1, \ldots, v_{|P|}$) such that $(v_i, v_{i+1}) \in D$ for all $1 \leq i \leq |P| - 1$. A single vertex is a path of length 1. A *path cover* of H is a collection of vertex disjoint directed paths that covers all vertices in V. We consider the following operations that can transform a given path cover to a different one:

1. *Path merging*: Two paths can be merged into one longer path if $H(V, D)$ has an arc from the end of one path to the start of the other path.
2. *Path unbalancing*: Consider any two paths P and P' with $1 < |P| \leq |P'|$, let v_s and v_t denote the first and last vertices of P, and let v'_s and v'_t denote the first and last vertices of P'. If $(v_s, v'_s) \in D$ we may remove v_s from P and append it at the beginning of P'. Likewise, if $(v'_t, v_t) \in D$ we may remove v_t from P and append it at the end of P'.
3. *Rotation*: if there is a path P (say, $v_s, \ldots, v_t$) such that $(v_t, v_s) \in D$, we may add the arc (v_t, v_s) to P (obtaining a cycle), and then remove any other single arc from the resulting cycle to get a path P'. Observe that P' and P have the same vertex set, but they differ in their starting vertex along the cycle $v_s, \ldots, v_t, v_s$.

A path cover is *maximal* if no path merging operation and no path unbalancing can be applied to it, and furthermore, this continues to hold even after performing any single rotation operation.

▶ **Proposition 4.** *Given a bipartite graph $G(U, V; E)$ with a perfect matching $\{(u_i, v_i)\}$, a maximal path cover in the associated spoiling graph $H(V, D)$ can be found in $O(n^2)$ time.*

Proof. Start with the trivial path cover in which each vertex of V forms a path of length 1, and perform arbitrary path merging and path unbalancing operations (some of which are preceded by a single rotation operation) until no longer possible. The process must end within $O(n^2)$ operations, because each path merging and each path unbalancing operation increases the sum of squares of the lengths of the paths, and the sum of squares of the lengths is at most n^2. ◀

Given a maximal path cover of H (where p denotes the number of paths in the path cover), sort the paths in order of increasing lengths, breaking ties arbitrarily. Hence $1 \leq |P_1| \leq |P_2| \leq \ldots \leq |P_p|$. We consider the following classes of vertices of V:

1. *Singleton vertices* V_1. These are the vertices that belong to paths of length 1 in the given maximal path cover. Let $k = |V_1|$ denote the number of singleton vertices. Observe that $|P_k| = 1$ and $|P_{k+1}| > 1$.
2. *Other vertices* $V_2 = V \setminus V_1$. We partition V_2 into three subclasses of vertices:
   a. *Start vertices* S. These are the starting vertices of those paths that have length larger than 1. The start vertex of path j, for $k < j \leq p$, is denoted by s_j.
   b. *End vertices* T. These are the end vertices of those paths that have length larger than 1. The end vertex of path j, for $k < j \leq p$, is denoted by t_j.
   c. *Intermediate vertices* $I = V_2 \setminus (S \cup T)$.

▶ **Lemma 5.** *The classes of vertices listed above have the following properties:*
1. *There are no arcs in H between vertices of V_1. Hence no vertex of V_1 can be a spoiler vertex for another vertex in V_1.*
2. *There are no arcs in H from vertices of V_1 to vertices in S. Hence no vertex of V_1 can be a spoiler vertex for a vertex in S.*

3. *There are no arcs in H from vertices of T to vertices in V_1. Hence no vertex of T can be a spoiler vertex for a vertex in V_1.*
4. *For $i \neq j$, there are no arcs in H from any vertex $t_i \in T$ to any vertex $s_j \in S$. Hence no vertex of T can be a spoiler vertex for a vertex in S, unless they both belong to the same path in the given maximal path cover.*
5. *$(s_i, s_j) \notin D$ for any $s_i, s_j \in S$ with $i < j$. Hence s_i cannot be a spoiler vertex for s_j if $i < j$.*
6. *$(t_j, t_i) \notin D$ for any $t_i, t_j \in S$ with $i < j$. Hence t_j cannot be a spoiler vertex for t_i if $i < j$.*
7. *If for some $s_j \in S$ and $t_j \in T$ (where s_j and t_j are start and end vertices of the same path P_j) it holds that $(t_j, s_j) \in D$, then there are no arcs from $(T \setminus \{t_j\}) \cup V_1$ to any of the vertices of P_j, and likewise, no arcs from $s_{k+1}, \ldots, s_{j-1}$ to any of the vertices of P_j.*

Proof. All properties follow from the maximality of the path cover. Properties 1,2,3 and 4 hold because otherwise one could perform a path merging operation. Properties 5 and 6 hold because otherwise one could perform a path unbalancing operation. Property 7 holds because otherwise one could perform a rotation operation for path P_j, followed either by a path merging operation (if there is an arc from $(T \setminus \{t_j\}) \cup V_1$ to any of the vertices of P_j) or a path unbalancing operation (if there is an arc from $s_{k+1}, \ldots, s_{j-1}$ to any of the vertices of P_j). ◀

We now introduce additional notation. Considering only the arcs in D leading from V_2 to V_1, we let M_{21} denote the maximum matching among these arcs. In our analysis we shall consider three parameters:

1. ϵ_1: its value is such that $k = \left(\frac{1}{2} - \epsilon_1\right) n$ (recall that $k = |V_1|$ is the number of singleton paths in the maximal path cover). Observe that ϵ_1 might be negative.
2. ϵ_2: its value is such that $p = k + \epsilon_2 n = \left(\frac{1}{2} - \epsilon_1 + \epsilon_2\right) n$ (recall that p is the total number of paths in the maximal path cover). Necessarily, $\epsilon_2 \geq 0$.
3. ϵ_3: its value is such that $|M_{21}| = \left(\frac{1}{2} - \epsilon_3\right) n$. Necessarily, $\epsilon_3 \geq 0$.

Given the above classes of vertices, we consider four possible candidate permutations for π (denoted $\pi_1, \pi_2, \pi_3, \pi_4$, see below for details). Given some permutation π, we shall use the notation $\rho(\pi)$ to denote the fraction of vertices guaranteed to be matched under π. This fraction will be expressed as a function of the parameters ϵ_1, ϵ_2 and ϵ_3, and we will show that regardless of the value of these parameters, there must be some π with $\rho(\pi) \geq \frac{22}{43}$.

The following four lemmas present the four candidate permutations for π along with their corresponding guarantees. Whenever unspecified, the order within a set of vertices can be arbitrary; e.g., for two sets of vertices X, Y, $\pi = X, Y$ means that the set X precedes Y and the order within X, as well as the order within Y, is arbitrary.

▶ **Lemma 6.** *For G and $\pi_1 = V_2, V_1$,*

$$\rho(\pi_1) \geq \frac{1}{n}\left(|V_1| + \frac{|V_2|}{2} - \frac{|M_{21}|}{2}\right) = \frac{1}{2} - \frac{\epsilon_1}{2} + \frac{\epsilon_3}{2}.$$

Proof. Let σ be an arbitrary permutation over U. Let m denote the number of vertices in V_2 that are matched to vertices in U_1, where U_1 is the set of partners of V_1. Then $m \leq |M_{21}|$. Of the $|V_2| - m$ vertices of V_2 not matched to vertices in U_1, at least half are matched (because for every unmatched vertex from this set, its partner must be matched to a different vertex from this set). In addition, all those vertices of V_1 whose partner is not matched to V_2 are matched, because of property 1 of Lemma 5. Hence the total number of vertices matched is at least $m + \frac{|V_2|-m}{2} + |V_1| - m \geq |V_1| + \frac{|V_2|}{2} - \frac{|M_{21}|}{2}$, as desired. ◀

▶ **Lemma 7.** *For G and $\pi_2 = V_1, V_2$,*

$$\rho(\pi_2) \geq \frac{2}{3} - \frac{1}{3}(\epsilon_1 + \epsilon_3).$$

Proof. Let σ be an arbitrary permutation over U. All vertices in V_1 are matched because of property 1 of Lemma 5. As to the vertices in V_2, observe that $|N(V_2)| \geq |V_2| + |M_{21}|$, as the set $N(V_2)$ includes the $|V_2|$ partners of V_2, plus at least $|M_{21}|$ additional neighbors in U_1 (due to the matching M_{21}). Moreover, if x vertices are removed from V_2, the number of remaining neighbors is at least $|V_2| + |M_{21}| - 2x$, because each vertex of V_2 contributed at most two neighbors to the lower bound that we provided on the number of neighbors.

Let x denote the number of vertices in V_2 matched under (π, σ). Then the size of the matching is $|V_1| + x$, the number of unmatched vertices in V_2 is $|V_2| - x$, and they have at least $|V_2| + |M_{21}| - 2x$ neighbors which have to be matched. Since the number of matched vertices at each side is the same, we have that $|V_1| + x \geq |V_2| + |M_{21}| - 2x$.

We get that

$$\begin{aligned} 3x \;\geq\; |V_2| + |M_{21}| - |V_1| &= n\left(\frac{1}{2} + \epsilon_1\right) + n\left(\frac{1}{2} - \epsilon_3\right) - n\left(\frac{1}{2} - \epsilon_1\right) \\ &= \left(\frac{1}{2} + 2\epsilon_1 - \epsilon_3\right) n. \end{aligned}$$

Therefore, the size of the matching is at least

$$|V_1| + x \geq \left(\frac{1}{2} - \epsilon_1\right) n + \left(\frac{1}{6} + \frac{2\epsilon_1}{3} - \frac{\epsilon_3}{3}\right) n = \left(\frac{2}{3} - \frac{1}{3}(\epsilon_1 + \epsilon_3)\right) n. \qquad \blacktriangleleft$$

▶ **Lemma 8.** *For G and $\pi_3 = t_p, \ldots, t_{k+1}, V_1, s_{k+1}, \ldots, s_p, I$,*

$$\rho(\pi_3) \geq \frac{2p - k}{n} = \frac{1}{2} - \epsilon_1 + 2\epsilon_2.$$

Proof. In π_3, we refer to the vertices of $T \cup V_1 \cup S$ as the *prefix* of π_3, and to the vertices of I as the *suffix*. Lemma 5 implies that within the prefix, the only arcs of H that go from an earlier vertex to a later vertex are of the form (t_j, s_j) (for a path P_j that can undergo a rotation). We claim that regardless of σ, all the prefix will be matched. As the length of this prefix is $2p - k$, this will prove the lemma.

It remains to prove the claim. Suppose first that in the above prefix there are no arcs of H that go from an earlier vertex to a later vertex. Then earlier vertices in this prefix cannot be spoiling vertices for later vertices. Hence indeed, regardless of σ, all the prefix will be matched.

Suppose now that in the prefix of π_3 there are arcs of H that go from an earlier vertex to a later vertex. As noted above, such an arc would be of the form (t_j, s_j). We need to show that even if t_j acts as a spoiling vertex for s_j under π_3 and σ, the vertex s_j will still be matched. Consider the path P_j, and let us rename its vertices as $x_1, \ldots, x_\ell$ (where previously we used $s_j = x_1$ and $t_j = x_\ell$). We wish to show the x_1 would be matched even if x_ℓ is matched to the partner of x_1. The path P_j implies that the partner of x_2 is a neighbor of x_1 in G. Hence x_1 will be matched if no vertex preceding $x_1 = s_j$ in π_3 is matched to the partner of x_2. By property 7 of Lemma 5, there is no arc in H from any of the vertices $T \cup V_1 \cap \{s_{k+1}, \ldots, s_{j-1}\} \setminus \{t_j\}$ to x_2, and consequently none of them can be matched to the partner of x_2. As to $t_j = x_\ell$, it might be a neighbor of the partner of x_2 (in fact, it could be that $\ell = 2$), but we already assumed that t_j is matched to the partner of x_1, and hence it is not matched to the partner of x_2. Hence no vertex preceding $x_1 = s_j$ in π_3 is matched to the partner of x_2, and hence s_j will be matched. $\blacktriangleleft$

Let V_e (V_o, respectively) denote those vertices of $S \cup I$ that are at even (odd, respectively) distance from the beginning of their respective path. Observe that $S \subset V_e$.

▶ **Lemma 9.** *For G and $\pi_4 = t_p, \ldots, t_{k+1}, V_1, V_o, V_e$,*

$$\rho(\pi_4) \geq \frac{5}{9} - \frac{p}{9n} = \frac{1}{2} + \frac{\epsilon_1}{9} - \frac{\epsilon_2}{9}.$$

Proof. Observe that $|V_e| \geq |V_o|$, because in every path (of length above 1) the vertices alternate in entering V_e and V_o, and start with V_e. Observe also that every vertex $v \in V_e$ contributes two distinct neighbors to $N(V_e)$: the partner of v, and the partner of the vertex that follows v on its path (note that the vertex that follows v is not in V_e). Likewise, every vertex $v \in V_o$ contributes two distinct neighbors to $N(V_o)$.

Regardless of σ, all p vertices of T and V_1 are matched, as in Lemma 8. For a given σ, let n_o be the number of vertices matched in V_o and let n_e be the number of vertices matched in V_e. Then, $|V_o| - n_o$, the number of unmatched vertices in V_o, satisfies $2(|V_o| - n_o) \leq p + n_o$, because the neighbors of the unmatched vertices in V_o need to be matched to earlier vertices in $T \cup V_1 \cup V_o$. Likewise, $|V_e| - n_e$, the number of unmatched vertices in V_e, satisfies $2(|V_e| - n_e) \leq p + n_o + n_e$. Adding two times the first inequality and three times the second we get that $4|V_o| + 6|V_e| - 4n_o - 6n_e \leq 5p + 5n_o + 3n_e$. Using $|V_o| + |V_e| = n - p$ and $|V_e| \geq |V_o|$, we can replace $4|V_o| + 6|V_e|$ by $5(n-p)$, implying that $9(p + n_o + n_e) \geq 5n - p$, as desired. ◀

We can now prove Theorem 2.

Proof. Observe that $\rho(G) \geq \max_{i \in [1,4]}[\rho(\pi_i)]$.

By Lemma 6 we have: $\rho(\pi_1) \geq \frac{1}{2} - \frac{\epsilon_1}{2} + \frac{\epsilon_3}{2}$.

By Lemma 7 we have: $\rho(\pi_2) \geq \frac{2}{3} - \frac{1}{3}(\epsilon_1 + \epsilon_3)$.

By Lemma 8 we have: $\rho(\pi_3) \geq \frac{1}{2} - \epsilon_1 + 2\epsilon_2$.

By Lemma 9 we have: $\rho(\pi_4) \geq \frac{1}{2} + \frac{\epsilon_1}{9} - \frac{\epsilon_2}{9}$.

Taking a weighted average of the lower bounds provided by the four lemmas, with weights $\frac{2}{43}$, $\frac{3}{43}$, $\frac{2}{43}$, $\frac{36}{43}$, respectively, results in a weighted average of $\frac{22}{43}$. Hence regardless, of the values of ϵ_1, ϵ_2 and ϵ_3, at least one of the lemmas gives $\rho(G) \geq \frac{22}{43}$. For $\epsilon_1 = \frac{19}{86}$, $\epsilon_2 = \frac{10}{86}$ and $\epsilon_3 = \frac{21}{86}$, none of the lemmas implies a bound better than $\frac{1}{2} + \frac{1}{86} = \frac{22}{43}$.

The above analysis leads to a polynomial time algorithm for finding π that ensures $\rho(G) \geq \frac{22}{43}$. A maximal path cover of $H(V, D)$ can be found in polynomial time by Proposition 4. Thereafter, the sets V_1, S, T, V_e and V_o can easily be determined, and likewise, the values of ϵ_1 and ϵ_2 can be easily computed. A maximum matching M_{21} (from V_2 to V_1 in H) can be computed in polynomial time using any standard algorithm for maximum bipartite matching. Thereafter, ϵ_3 can be easily computed. Given the values ϵ_1, ϵ_2 and ϵ_3, one can determine which of π_1, π_2, π_3 or π_4 provides a higher lower bound on ρ, and use that permutation as π. ◀

3 Regular Graphs

In this section we consider the case where $G(U, V; E)$ is a d-regular bipartite graph with $2n$ vertices. Given that such graphs have d edge disjoint perfect matchings, one can hope to achieve higher values for ρ for these graphs.

3.1 Positive Result

The following known proposition (see for example [18]) establishes a lower bound on ρ, as a function of d. A proof is provided for completeness.

▶ **Proposition 10.** *For every d-regular graph $G \in G_n$, it holds that $\rho[G] \geq \frac{d}{2d-1}$.*

Proof. Since the greedy algorithm produces a maximal matching, it suffices to show that every maximal matching in a d-regular graph has size at least $\frac{d}{2d-1}n$. To see this, let $S \subset U$ and $T \subset V$ be the sets of unmatched nodes in an arbitrary maximal matching, and suppose $|S| = |T| = (1-\alpha)n$. The nodes in S, T must form an independent set. Consider the size of the edge set connecting S and $V \setminus T$. On the one hand, this size equals $(1-\alpha)nd$ (since all edges from S go to $V \setminus T$); on the other hand, this size is at most $\alpha n(d-1)$ (since at least one edge from each node in $V \setminus T$ goes to $U \setminus S$). Thus, $(1-\alpha)nd \leq \alpha n(d-1)$, implying that $\alpha \geq d/(2d-1)$. Hence we have that $|M_G[\sigma, \pi]| \geq \frac{d}{2d-1}n$, for every π. ◀

Remark: For every d there exists a d-regular graph with a perfect matching that admits a maximal matching of size $\frac{d}{2d-1}n$. Suppose that $n = 2d-1$, and consider a d-regular graph where $|S| = |T| = d-1$ for some $S \subset U, T \subset V$, every node in $U \setminus S$ is connected to a single, different node in $V \setminus T$, and to all $d-1$ nodes in T, and every node in $V \setminus T$ is connected to a single, different node in $U \setminus S$, and to all $d-1$ nodes in S. The perfect matching between $U \setminus S$ and $V \setminus T$ is a maximal matching of size $\frac{d}{2d-1}n$.

The lower bound of Proposition 10 approaches $\frac{1}{2}$ from above as d grows. The following theorem shows that there exists some permutation π that ensures that the fraction of matched vertices approaches $5/9$. This is a direct corollary from Lemma 9 and a theorem in [7].

▶ **Corollary 11.** *For d-regulars bipartite graphs, $\rho \geq \frac{5}{9} - O(\frac{1}{\sqrt{d}})$.*

Proof. Theorem 3 in [7] shows that every n-vertex d-regular graph has a path cover (referred to as a *linear forest*) with $p = O(\frac{n}{\sqrt{d}})$ paths. By Lemma 9, $\rho(G) \geq \frac{5}{9} - O(\frac{1}{\sqrt{d}})$. ◀

Remarks.

1. For small d, the bound of $\rho \geq \frac{d}{2d-1}$ which holds for every maximal matching is stronger than the bound in Corollary 11.
2. The proof of Corollary 11 extends to graphs that are nearly d-regular, by using Theorem 5 from [7].
3. For d-regular graphs, conjectures mentioned in [7] combined with our proof approach suggest that $\rho \geq \frac{5}{9} - O(\frac{1}{d})$.

3.2 Negative Result

The following example shows that even in a regular graph with arbitrarily high degree, there may be no permutation π that ensures to match more than a fraction $8/9$ of the vertices.

▶ **Theorem 12.** *For every integers $d, t \geq 1$, there is a regular bipartite graph $G_{d,t}$ of even degree $2d$ and $n = 3dt$ vertices on each side such that $\rho(G_{d,t}) \leq \frac{8}{9}$.*

Proof. Consider a regular bipartite graph $G(U, V; E)$ with even degree $2d$, and $3d$ vertices on each side. To define the edge set, let $U = U_1 \cup U_2 \cup U_3$ with each U_i of cardinality d, and similarly $V = V_1 \cup V_2 \cup V_3$ with each V_i of cardinality d. For every $i \neq j$, we have a complete bipartite graph between U_i and V_j, and for every i, there are no edges between U_i and V_i.

Let π be an arbitrary permutation over V, let S be the first $2d$ vertices in π, and let T be the last d vertices. Let i be such that $|V_i \cap T|$ is largest (breaking ties arbitrarily). Without loss of generality we may assume that $i = 3$, and then $|V_3 \cap T| \geq d/3$. Hall's condition implies that there is a perfect matching between $U_1 \cup U_2$ and S (and more generally, between $U_1 \cup U_2$ and any $2d$ vertices from V). Hence one can choose a permutation σ over U whose first $2d$ vertices are $U_1 \cup U_2$ that will match the vertices of S one by one. Thereafter, the vertices of $T \cap V_3$ will remain unmatched.

To get the graph $G_{d,t}$ claimed in the theorem, take t disjoint copies of $G(U, V; E)$ above. ◀

4 Random Permutation

In this section we consider scenarios in which the maximizing player is unaware of the graph structure. In such scenarios, the best she can do is impose a random permutation over the vertices in V.

We first show that there exists a graph $G \in G_n$ for which a random permutation does not match significantly more than a half of the vertices, even if every vertex has a high degree.

▶ **Proposition 13.** *There exists a bipartite graph $G(U, V; E) \in G_n$ such that a random permutation gets $\rho(G) = \frac{1}{2} + o(1)$ almost surely.*

Proof. Consider the graph $G(U, V; E)$, where $U = (U_1, U_2)$, $V = (V_1, V_2)$, and each of U_1, U_2, V_1, V_2 is of size $n/2$. The set of edges constitutes of a perfect matching between U_1 and V_1, a perfect matching between U_2 and V_2, and a bi-clique between U_1 and V_2. Let π be a random permutation. With high probability, for each vertex $v_1 \in V_1$, except for $\sim \sqrt{n}$ such vertices, we can associate a unique vertex $v_2 \in V_2$ that precedes v_1 in π. Let $S \subseteq V_1$ denote this set. Consider an arrival order σ in which agents in U_1 arrive first, with a vertex u_{1j} preceding a vertex $u_{1j'}$ if $\pi(v_{2j}) < \pi(v_{2j'})$. Every vertex in U_1 such that its neighbor in V_1 (according to the perfect matching) belongs to S will be matched to the corresponding vertex in V_2. Therefore, all but $\sim \sqrt{n}$ vertices of V_1 remain unmatched, and the size of the matching is $n(1/2 + o(1))$, whereas $OPT = n$. ◀

In the above example, if the vertices of V with degree 1 are placed in the prefix of π, then the obtained matching is optimal. This might suggest that prioritizing low degree vertices in π (and randomizing within sets of vertices of comparable degrees) leads to good performance. However, the example above can be transformed into one where all vertices in V have the same degree. To see this, consider a graph where vertices are partitioned into sets of perfect matchings of size $\sqrt{n}$, $\{(U_{11}, V_{11}), \ldots, (U_{1\sqrt{n}}, V_{1\sqrt{n}}), (U_{21}, V_{21}), \ldots, (U_{2\sqrt{n}}, V_{2\sqrt{n}})\}$. Each V_{1i} is also connected in a bi-clique to U_{2i}, and in addition, there are sets U', V' of size $\sqrt{n}$ each connected to the vertices of the other side to balance out the degrees. A similar argument shows that in this graph, a random permutation performs badly as well.

In contrast to the last examples, in some classes of graphs, a random permutation guarantees to match a fraction of the vertices that is bounded away from a half. This is the case, for example, in Hamiltonian graphs. The formal statement and proof are deferred to Section 6.

5 Finding a perfect π

A permutation π over V is said to be *perfect* if for every permutation σ over U, $|M_G[\sigma, \pi]| = n$. A vertex $v \in V$ is *good* with respect to a set of vertices $S \subset V$ if there is no matching between $N(v)$ and S. Given permutation π over V, let $\overline{\pi}(v)$ be the set of vertices preceding v in π.

▶ **Observation 14.** *π is perfect if and only if every vertex $v \in V$ is good with respect to $\overline{\pi}(v)$.*

Proof. Suppose there exists a vertex $v \in V$ that is not good. Then, consider a permutation σ where the vertices in $N(v)$ are placed first, in an order corresponding to the rank (according to π) of their partners in the perfect matching between $N(v)$ and $\overline{\pi}(v)$. In such a σ, v will not be matched. Now suppose that all vertices in V are good. Then, for every σ, for every $v \in V$, there exists a vertex $u \in U$ that is not matched to a vertex in $\overline{\pi}(v)$; therefore v will surely be matched. ◀

We present an algorithm that finds a perfect π if one exists, and claims that no such π exists otherwise.

Checking whether v_i is good can be done in polynomial time by running a maximum matching algorithm on $N(v_i)$ and S_i.

An algorithm for finding a perfect π.

1. Let $S_1 = V$.
2. In iteration $i = 1, \ldots, n$:
   a. Find an arbitrary good vertex $v_i \in S_i$ with respect to $S_i \setminus \{v_i\}$, and place it in rank $n - i + 1$ in π.
   b. Set $S_{i+1} = S_i \setminus \{v_i\}$.

▶ **Lemma 15.** *If there exists a perfect π, then the algorithm is guaranteed to find a good v_i in every iteration i.*

Proof. Consider some perfect permutation π (not necessarily the one produced by our algorithm), and the suffix $V_{i-1} = v_{i-1}, v_{i-2}, \ldots, v_1$ of vertices chosen in the first $i-1$ iterations of the algorithm (of course, there must be a good v_1 at the first iteration, otherwise there is no perfect π). Let π' be the permutation that places $v_{i-1}, v_{i-2}, \ldots, v_1$ as the lowest ranked vertices in the same order as the algorithm picked them, and places all other vertices of $V \setminus V_{i-1}$ in a higher rank than V_{i-1} according to their internal order in π.

Since every v_j, $1 \leq j \leq i-1$, is good with respect to $V \setminus \{v_1, \ldots, v_j\}$, then clearly v_j is good with respect to $\overline{\pi'}(v_j)$ (since $\overline{\pi'}(v_j) = V \setminus \{v_1, \ldots, v_j\}$). Now consider a vertex $v \in V \setminus V_{i-1}$. This vertex is good with respect to $\overline{\pi}(v)$, and since $\overline{\pi'}(v) \subseteq \overline{\pi}(v)$, it is clear that v is good with respect to $\overline{\pi'}(v)$. It follows that π' is perfect as well.

Let v'_i be the vertex ranked in position $n - i + 1$ in π'. Since π' is perfect, this vertex is good with respect to $\overline{\pi'}(v'_i)$. But since $\overline{\pi'}(v'_i) \cup \{v'_i\}$ is exactly the set S_i in iteration i, it is guaranteed that the algorithm can find a good v_i in this iteration. ◀

Let π be the permutation computed by the algorithm. Since every vertex v is good with respect to $\overline{\pi}(v)$, it follows from Observation 14 that π is perfect, and Proposition 1 follows.

6 Hamiltonian Bipartite Graphs

In this section we establish two results about Hamiltonian graphs. First, we show that $\rho \geq \frac{5}{9}$. Note that, since for the case of a Hamiltonian graph there exists a path cover using only a single path (*i.e.*, $p = 1$), Lemma 9 directly implies that $\rho \geq \frac{5}{9} - \frac{1}{9n}$. Theorem 16 improves this bound to 5/9. Second, we show that for Hamiltonian graphs, even a random permutation π ensures a ratio that is bounded away from $\frac{1}{2}$ (this is in contrast to general graphs, see Section 4).

▶ **Theorem 16.** *For every Hamiltonian graph G, it holds that $\rho \geq \frac{5}{9}$.*

Proof. Consider a Hamiltonian graph G and a Hamiltonian cycle $u_1, v_1, u_2, v_2, \ldots, u_n, v_n, u_1$ that traverses through its vertices. Let $V_o = \{v_i \ : \ i = 2\ell + 1, \ \ell \in \mathbb{N}, \ i \leq n\}$ be the set of odd labeled vertices of the cycle, and $V_e = V \setminus V_o$.

We first claim that if the number of vertices is even ($|V_o| = |V_e| = \frac{n}{2}$), then $\pi = V_e, V_o$ (and in fact, also $\pi = V_o, V_e$) ensures that $\rho \geq 5/9$. Let n_e (respectively, n_o) be the number of vertices of V_e (respectively, V_o) matched in $M_G[\sigma, \pi]$ defined using $\pi = (V_e, V_o)$ and an arbitrary σ. Similar to the proof of Lemma 9, it is easy to see that each vertex in V_e contributes two distinct neighbors to $N(V_e)$, and each vertex in V_o contributes two distinct neighbors to $N(V_o)$ (the difference from the proof of Lemma 9 is that this property also holds for $v_1 \in V_o$ and $v_n \in V_e$, and this follows because v_1 and v_n contribute u_1 to $N(V_o)$ and $N(V_e)$, respectively). The number of unmatched vertices in V_o, namely $|V_o| - n_o$, satisfies

$$2(|V_o| - n_o) \leq n_o,$$

because the neighbors of the unmatched vertices in V_o must be matched to vertices in V_o, as they precede the vertices in V_e in π. Likewise, the number of unmatched vertices in N_e, namely $|V_e| - n_e$, satisfies

$$2(|V_e| - n_e) \leq n_e + n_o.$$

Adding two times the first inequality and three times the second, we get

$$4|V_o| + 6|V_e| \leq 9(n_o + n_e) \ \Rightarrow \ \frac{5}{9} \cdot n \leq n_o + n_e.$$

As $|V| = n$ and $|M_G[\sigma, \pi]| = n_o + n_e$, this implies that $\rho \geq \frac{5}{9}$.

We now handle the case where n is odd. Lemma 9 ensures that $\frac{5}{9} \cdot n - \frac{1}{9}$ of the vertices are matched by π_4 when the path cover is of a single path. If $\frac{5}{9} \cdot n - \frac{1}{9}$ is not integral, then $\lceil \frac{5}{9} \cdot n - \frac{1}{9} \rceil$ is at least $\frac{5}{9} \cdot n$, thus $\rho \geq \frac{5}{9}$. Therefore, it only remains to handle the case where $\frac{5}{9} \cdot n - \frac{1}{9}$ is integral; namely where $n = 18\ell + 11$ for some integer ℓ. In this case, we show that $\pi = V_e, V_o$ ensures that $|M[\sigma, \pi]| > \frac{5}{9} \cdot n$ for every σ. Since $n = 18\ell + 11$, it holds that $|V_o| = 9\ell + 6$ and $|V_e| = 9\ell + 5$. As in the case where n is even, every vertex in V_e contributes two distinct neighbors to $N(V_e)$. As for V_o, every vertex in $V_o \setminus \{v_1, v_n\}$ also contributes two distinct neighbors to $N(V_o)$, and v_1 and v_n contribute (together) to $N(V_o)$ three additional distinct vertices (since they share a vertex along the Hamiltonian cycles). Using the same reasoning as before, it follows that

$$2(|V_e| - n_e) \leq n_e \Rightarrow 18\ell + 10 \leq 3 \cdot n_e \Rightarrow n_e \geq 6\ell + 3\frac{1}{3}.$$

Since n_e is integral, this implies that

$$n_e \geq 6\ell + 4. \tag{1}$$

Again, for $|V_o| - n_o$ we have $2(|V_o| - n_o) - 1 \leq n_o + n_e$. Rearranging gives us

$$n_e + 3n_o \geq 18\ell + 11.$$

Adding twice Inequality (1) to the last inequality yields

$$n_e + n_o \geq 10\ell + 6\frac{1}{3} > 10\ell + 6\frac{1}{9} = \frac{5}{9} \cdot |V|,$$

which implies $\rho > \frac{5}{9}$. This concludes the proof. ◀

Next, we show that for the case of a Hamiltonian graph, a random permutation yields a $\rho > 1/2$.

▶ **Theorem 17.** *Consider choosing a permutation π uniformly at random. For every Hamiltonian graph G, it holds that $E_\pi[\min_\sigma[|M_G[\sigma,\pi]|]] > 0.5012$.*

We explain the proof approach here, and present the full details in Appendix A.

Proof Approach

We first provide a high level overview of our proof approach.

A permutation π (over V) is said to be *safe* for a set $S \subset V$ if for every permutation σ (over U) the greedy process matches at least one vertex in S (i.e., no σ leaves all vertices in S unmatched). Fix some constant ϵ. In order to establish that $\rho \geq (1/2 + \epsilon)$, we need to show that there exists a permutation π that is safe for every set S of size $(1/2 - \epsilon)n$. Our proof approach is the following: we show that for a permutation π chosen uniformly at random, the expected number (expectation taken over choice of π) of sets of size $(1/2 - \epsilon)n$ for which pi is unsafe is smaller than 1. This implies that there exists a permutation π that is safe for all sets of size $(1/2 - \epsilon)n$, as desired.

First, we define a collection of sets that can potentially remain unmatched ("bad" sets). Let B_ϵ denote the set of all sets $S \subset U$ of size $(1/2 - \epsilon)n$ such that there exists a permutation π that is unsafe for S.

Second, for a given set S and permutation π we identify a sufficient condition for π to be safe for S. Let $S' \subset S$ be the lowest αn vertices in S (according to π), let v' be the last vertex in S' (i.e., the vertex with rank αn in S'), and let P be the set of vertices in $V - S'$ that precede v' in π. We claim that if the size of P is smaller than the size of $N(S')$ (the neighbors of S'), then π is safe for S. To see this, assume by way of contradiction that π is unsafe for S'. This implies that every vertex in $N(S')$ is matched to a vertex in $V - S'$. Since there are strictly less than $|N(S')|$ vertices in $V - S'$ that precede v', at least one of the vertices in $N(S')$ must be matched to a vertex higher than v'. But, this vertex has a neighbor in S' with lower rank, contradicting the greedy process.

We now proceed by establishing the following three lemmas:

- Few bad sets lemma: the size of B_ϵ is at most $n_B = n_B(\epsilon)$.
- *Expansion lemma:* given a set $S \subset V$ and parameters α, β, the probability (over a random choice of π) that the lowest αn vertices in S have less than βn neighbors is at most $p = p(\alpha, \beta)$.
- *Good order lemma:* given a set $S \subset V$ and parameters α, β, the probability (over a random choice of π) that the $(\alpha n)^{th}$ lowest vertex in S is higher than βn vertices in $V \setminus S$ is at most $q = q(\alpha, \beta)$.

The three lemmas are combined as follows. For a given set S, due to the sufficient condition identified above, it follows from the union bound that the probability that a uniformly random permutation π is unsafe for S is at most $p + q$. Applying the union bound once more over all bad sets (at most n_B sets, as implied by the few bad sets lemma), implies that the probability that a uniformly random permutation π is unsafe for some set of size $(1/2 - \epsilon)n$ is at most $n_B(p + q)$. Thus, to conclude the proof, it remains to find parameters such that $n_B(p + q) < 1$.

The good order lemma is independent of the graph structure. In contrast, the expansion lemma and the few bad sets lemma rely heavily on the structure of the graph. As it turns out, Hamiltonian graphs have properties that enable us to establish the two lemmas with good parameters.

References

1. Allan Borodin, Denis Pankratov, and Amirali Salehi-Abari. On Conceptually Simple Algorithms for Variants of Online Bipartite Matching. In *Approximation and Online Algorithms - 15th International Workshop, WAOA 2017, Vienna, Austria, September 7-8, 2017, Revised Selected Papers*, pages 253–268, 2017. doi:10.1007/978-3-319-89441-6_19.
2. Miroslav Chlebík and Janka Chlebíková. Approximation hardness of edge dominating set problems. *J. Comb. Optim.*, 11(3):279–290, 2006. doi:10.1007/s10878-006-7908-0.
3. Ilan Reuven Cohen and David Wajc. Randomized Online Matching in Regular Graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 960–979, 2018. doi:10.1137/1.9781611975031.62.
4. Vincent Cohen-Addad, Alon Eden, Michal Feldman, and Amos Fiat. The Invisible Hand of Dynamic Market Pricing. In *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, pages 383–400, 2016. doi:10.1145/2940716.2940730.
5. Paul Dütting, Michal Feldman, Thomas Kesselheim, and Brendan Lucier. Posted Prices, Smoothness, and Combinatorial Prophet Inequalities. *CoRR*, abs/1612.03161, 2016. arXiv:1612.03161.
6. Tomer Ezra, Michal Feldman, Tim Roughgarden, and Warut Suksompong. Pricing Identical Items. *CoRR*, abs/1705.06623, 2017. arXiv:1705.06623.
7. Uriel Feige, R. Ravi, and Mohit Singh. Short Tours through Large Linear Forests. In *Integer Programming and Combinatorial Optimization - 17th International Conference, IPCO 2014, Bonn, Germany, June 23-25, 2014. Proceedings*, pages 273–284, 2014. doi:10.1007/978-3-319-07557-0_23.
8. Michal Feldman, Nick Gravin, and Brendan Lucier. Combinatorial Auctions via Posted Prices. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 123–135, 2015. doi:10.1137/1.9781611973730.10.
9. David Gale and L. S. Shapley. College Admissions and the Stability of Marriage. *American Math. Monthly*, 69:9–15, 1962.
10. Gagan Goel and Aranyak Mehta. Online Budgeted Matching in Random Input Models with Applications to Adwords. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pages 982–991, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1347082.1347189.
11. Chinmay Karande, Aranyak Mehta, and Pushkar Tripathi. Online Bipartite Matching with Unknown Distributions. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 587–596, New York, NY, USA, 2011. ACM. doi:10.1145/1993636.1993715.
12. R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An Optimal Algorithm for On-line Bipartite Matching. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, STOC '90, pages 352–358, New York, NY, USA, 1990. ACM. doi:10.1145/100216.100262.
13. Renato Paes Leme and Sam Chiu-wai Wong. Computing Walrasian Equilibria: Fast Algorithms and Structural Properties. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 632–651, 2017. doi:10.1137/1.9781611974782.42.
14. Brendan Lucier. An economic view of prophet inequalities. *SIGecom Exchanges*, 16(1):24–47, 2017. doi:10.1145/3144722.3144725.
15. Mohammad Mahdian and Qiqi Yan. Online Bipartite Matching with Random Arrivals: An Approach Based on Strongly Factor-revealing LPs. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 597–606, New York, NY, USA, 2011. ACM. doi:10.1145/1993636.1993716.

16 Vahideh H. Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online Stochastic Matching: Online Actions Based on Offline Statistics. *Mathematics of Operations Research*, 37(4):559–573, 2012. URL: http://www.jstor.org/stable/23358636.

17 Aranyak Mehta. Online Matching and Ad Allocation. *Foundations and Trends in Theoretical Computer Science*, 8(4):265–368, 2013. doi:10.1561/0400000057.

18 Joseph Naor and David Wajc. Near-Optimum Online Ad Allocation for Targeted Advertising. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015*, pages 131–148, 2015. doi:10.1145/2764468.2764482.

A Full Proof of Theorem 17

Throughout this section we use $H(\cdot)$ to denote the binary entropy function; i.e., given a constant $p \in (0,1)$, $H(p) = -p\log_2 p - (1-p)\log_2(1-p)$.

▶ **Fact 18** (Stirling's Approximation). *As $n \to \infty$,*

$$n! = (1+o(1))\sqrt{2\pi n}\left(\frac{n}{e}\right)^n.$$

Using Stirling's Approximation, one can derive the following bound.

▶ **Fact 19.** *For n and $k = pn$ for some constant $p \in (0,1)$,*

$$\binom{n}{k} = 2^{(H(p)+o(1))n}, \tag{2}$$

where $H(p) = -p\log_2 p - (1-p)\log_2(1-p)$ is the binary entropy function.

We first establish the good order lemma, which is independent of the graph structure.

▶ **Lemma 20** (Good order lemma). *Let $\alpha < \beta < 1$, $\rho = \frac{1}{2} + \epsilon$ for some $\epsilon > 0$ and $\bar{\rho} = 1-\rho$ such that $\frac{\beta}{\alpha} > \frac{\rho}{\bar{\rho}}$. Let $S \subset V$ be a set of size $\bar{\rho}n$. The probability that in a random permutation π there are at least βn vertices of $V \setminus S$ before αn vertices from S is at most $2^{-(H(\alpha+\beta)-H(\frac{\alpha}{\bar{\rho}})\bar{\rho}-H(\frac{\beta}{\rho})\rho-o(1))n}$.*

Proof. We first analyze the case that in the first $(\alpha+\beta)n$ vertices in π there are *exactly* αn vertices from S. The number of possibilities for this case is $\binom{\bar{\rho}n}{\alpha n}\binom{\rho n}{\beta n}$.

Let $\beta' = \beta + x$ and $\alpha' = \alpha - x$. By the conditions on α, β and ϵ, we have that $\frac{\beta'}{\alpha'} \geq \frac{\beta}{\alpha} \geq \frac{\rho}{\bar{\rho}}$. Therefore,

$$\beta'\bar{\rho} \geq \alpha'\rho \Rightarrow \beta'\bar{\rho} - \alpha'\beta' \geq \alpha'\rho - \alpha'\beta' \Rightarrow \frac{(\bar{\rho}-\alpha')}{\alpha'} \cdot \frac{\beta'}{(\rho-\beta')} \geq 1$$

$$\Rightarrow \frac{(\bar{\rho}n - \alpha'n + 1)}{\alpha' n} \cdot \frac{\beta' n + 1}{(\rho n - \beta' n)} \geq 1 \iff \frac{\binom{\bar{\rho}n}{\alpha' n}\binom{\rho n}{\beta' n}}{\binom{\bar{\rho}n}{\alpha' n - 1}\binom{\rho n}{\beta' n + 1}} > 1.$$

It follows that $\binom{\bar{\rho}n}{\alpha n}\binom{\rho n}{\beta n} > \binom{\bar{\rho}n}{\alpha' n}\binom{\rho n}{\beta' n}$ for every $\alpha' < \alpha$ and $\beta' > \beta$ such that $\alpha + \beta = \alpha' + \beta'$. Therefore, the probability to have at most αn vertices from S in the first $(\alpha+\beta)n$ vertices in π is at most

$$\frac{\alpha n \cdot \binom{\bar{\rho}n}{\alpha n}\binom{\rho n}{\beta n}}{\binom{n}{(\alpha+\beta)n}} = \frac{2^{(H(\frac{\alpha}{\bar{\rho}})\bar{\rho}+H(\frac{\beta}{\rho})\rho+o(1))n}}{2^{(H(\alpha+\beta)+o(1))n}}$$

$$= 2^{-(H(\alpha+\beta)-H(\frac{\alpha}{\bar{\rho}})\bar{\rho}-H(\frac{\beta}{\rho})\rho-o(1))n},$$

where the first equality follows Fact 19. ◀

Let $\rho = \frac{1}{2} + \epsilon$ for some constant $\epsilon > 0$, $\bar{\rho} = 1 - \rho = \frac{1}{2} - \epsilon$. The next lemma will be used in order to prove the few bad sets lemma and the expansion lemma. It uses the existence a Hamiltonian cycle in the graph in order to claim that most sets will have a large number of neighbors. Therefore, a random set will have a large expansion. In addition, there will be few sets of size $(\frac{1}{2} - \epsilon)n$ with less than $(\frac{1}{2} + \epsilon)n$ neighbors (i.e., a few bad sets).

▶ **Lemma 21.** *Let $\alpha \in (0, 1/2)$ and $\beta \in (\alpha, 1)$ be two constants such that $\delta = \beta - \alpha < \alpha/2$. The number of sets S of size αn where $|N(S)| \leq \beta n$ is at most $2^{(\alpha H(\frac{\delta}{\alpha}) + (1-\alpha)H(\frac{\delta}{(1-\alpha)}) + o(1))n}$.*

Proof. Consider a Hamiltonian cycle that traverses through the graph's vertices $H = (v_1, u_1, v_2, u_2, \ldots, v_n, u_n, v_1)$, where $\{v_i\}_{i\in[n]} = V$ and $\{u_i\}_{i\in[n]} = U$. Let S be some set of vertices from V of cardinality ρn. Note that in the cycle H, each vertex v of S has two neighbors, where one of these neighbors is joined with an adjacent vertex from V in the cycle. Therefore, the number of neighbors of a sequence of k consecutive vertices of V in H is $k + 1$. Thus, the set $N(S)$ is of size αn plus the number of consecutive blocks of vertices from V chosen.

We bound the number of ways to pick at most δn consecutive blocks of vertices from V. We first bound the number of ways to pick *exactly* δn such blocks. In this case, the αn chosen elements have to be within δn blocks. The number of ways to partition αn elements to δn noen empty blocks is $\binom{\alpha n - 1}{\delta n - 1}$. After deciding the number of elements in each block, we need to figure out their location along the Hamiltonian cycle. $(1 - \alpha)n$ elements reside outside of the blocks of chosen αn elements. We need to chose the location of the first block in H (for which there are n possibilities), and then the number of element between each block, where two blocks are separated by at least one element. The latter is equivalent to splitting $(1-\alpha)n$ elements into δn non empty bins, for which there are $\binom{(1-\alpha)n-1}{\delta n-1}$ possibilities. Overall, there are $n\binom{\alpha n-1}{\delta n-1}\binom{(1-\alpha)n-1}{\delta n-1}$ such possibilities[4].

For $\delta' < \delta$, one can similarly devise the bound of $n\binom{\alpha n-1}{\delta' n-1}\binom{(1-\alpha)n-1}{\delta' n-1}$ which is smaller than $n\binom{\alpha n-1}{\delta n-1}\binom{(1-\alpha)n-1}{\delta n-1}$ by our conditions on α and δ. Overall, we can bound the number of ways to pick at most δn consecutive blocks of vertices from V by

$$\begin{aligned}\delta n^2 \binom{\alpha n - 1}{\delta n - 1}\binom{(1-\alpha)n - 1}{\delta n - 1} &< \delta n^2 \binom{\alpha n}{\delta n}\binom{(1-\alpha)n}{\delta n} \\ &= 2^{o(1)n} \cdot 2^{(H(\frac{\delta}{\alpha}) + o(1))\alpha n} \cdot 2^{(H(\frac{\delta}{(1-\alpha)}) + o(1))(1-\alpha)n} \\ &= 2^{(\alpha H(\frac{\delta}{\alpha}) + (1-\alpha)H(\frac{\delta}{(1-\alpha)}) + o(1))n},\end{aligned}$$

where the first equality follows Fact 19. ◀

The expansion and few bad sets lemmas are obtained as direct corollaries of Lemma 21.

▶ **Lemma 22** (Few bad sets Lemma for Hamiltonian graphs)**.** *Let ϵ be a constant such that $\epsilon < 0.1$. The number of bad sets in any Hamiltonian graph is at most*

$$|B_\epsilon| \leq 2^{\left(\bar{\rho}H(\frac{2\epsilon}{\bar{\rho}}) + \rho H(\frac{2\epsilon}{\rho}) + o(1)\right)n}.$$

[4] Notice there's some over-counting in this argument, but this bound suffices for our purpose.

Proof. Notice that if a set S of size $\bar{\rho}n = (\frac{1}{2} - \epsilon)n$ has more than ρn neighbors, it cannot be left unmatched, since at least one of it's neighbors will could not be matched to $V \setminus S$. A direct application of Lemma 21 yields that the number of such sets is at most $2^{\left(\bar{\rho}H(\frac{2\epsilon}{\bar{\rho}})+\rho H(\frac{2\epsilon}{\rho})+o(1)\right)n}$. ◀

We note that this lemma is not true for general graphs. An example of a graph that admits $2^{n/4}$ bad sets is given in the full version.

▶ **Lemma 23** (Expansion Lemma for Hamiltonian graphs). *Consider a set $S \subset V$ of size $\bar{\rho}n$ and parameters α, β. The probability that the lowest αn vertices in S have less than βn neighbors is at most*

$$2^{\left(-H(\frac{\alpha}{\bar{\rho}})+\alpha H(\frac{\delta}{\alpha})+(1-\alpha)H(\frac{\delta}{(1-\alpha)})+o(1)\right)\bar{\rho}n}.$$

Proof. Consider a set S of size $\bar{\rho}n$, and the first αn vertices in S in a random permutation. This set is just a random subset of S of size αn. The number of choices of such subset is

$$\binom{\bar{\rho}n}{\alpha n} = 2^{(H(\frac{\alpha}{\bar{\rho}})+o(1))\bar{\rho}n}.$$

Notice that we can apply Lemma 21 for with set S, even though S is just a subset of V, because the same proof applies only with respect to a subset of vertices in one side of a Hamiltonian graph. Therefore, the number of subsets of size αn of S with at most βn neighbors is at most

$$2^{(\alpha H(\frac{\delta}{\alpha})+(1-\alpha)H(\frac{\delta}{(1-\alpha)})+o(1))\bar{\rho}n}.$$

Combining the above, we get that the probability that a random set of αn vertices of S have at most βn neighbors is at most

$$2^{(-H(\frac{\alpha}{\bar{\rho}})+\alpha H(\frac{\delta}{\alpha})+(1-\alpha)H(\frac{\delta}{(1-\alpha)})+o(1))\bar{\rho}n}.$$ ◀

Now that we have established the three lemmas we are ready to prove Theorem 17.

Proof of Theorem 17. Setting $\epsilon = 0.0012$, $\alpha = 0.245$ and $\beta = 0.3675$ (and $\rho = \frac{1}{2} + \epsilon$, $\bar{\rho} = 1 - \rho$), we get that these parameters satisfy the conditions for Lemmas 20, 23 and 22.

Applying Lemma 22, we get that the size of B_ϵ is at most $n_B \leq 2^{0.044n}$. Applying Lemma 23, we get that the probability that the lowest αn vertices of a set of size $\bar{\rho}n$ have less than βn neighbors is at most $p \leq 2^{-0.86n}$. Applying Lemma 20, we get that the probability that for a set S of size $\bar{\rho}n$ the αnth vertex in a random π comes after βn vertices of $V - S$ is at most $q \leq 2^{-0.45n}$. Combining these three, we get that the probability there exists a set of size $\bar{\rho}n$ unmatched by a random π is at most $n_B(p + q) < 1$, therefore, there must be a π that matches at least one vertex in each set of size $\bar{\rho}n$, and the proof follows. ◀

This proof approach can be also used to show that a random permutation guarantees to match more than a half of the vertices in every regular graph. On the other hand, Theorem 24 in Section C shows that one cannot hope to get $\rho > 3/4$ with a random permutation in regular graphs.

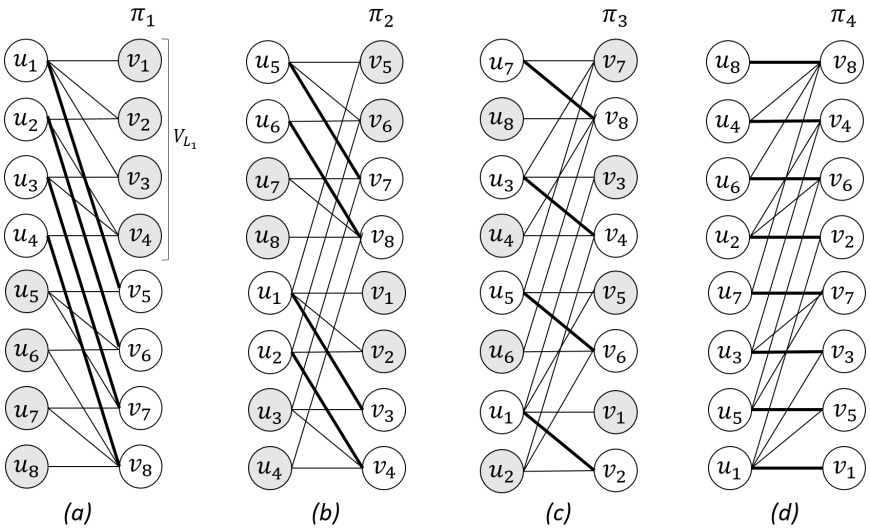


Figure 2 An iterative process where unmatched vertices are given priority. In every iteration thick edges are in the matching; gray vertices are unmatched.

B Iterative Process

A natural approach for establishing the existence of a good permutation π is the following iterative process of "upgrading" unmatched vertices. Given a permutation $\pi : V \to [n]$ and a permutation $\sigma : U \to [n]$, let $M[\pi, \sigma]$ be the result of the greedy matching where vertices in U arrive in order σ (from low to high) and each vertex $u \in U$ is matched to its lowest (under π) neighbor (or left unmatched if all its neighbors are already matched).

Fix an arbitrary permutation π_1 on V, and let σ_1 be a permutation on U minimizing the greedy matching[5]. Let $M_1 = M[\pi, \sigma]$ be the result of the greedy matching under permutations σ and π. If $|M_1|/n$ is some constant greater than $1/2$, then terminate with permutation π_1. Otherwise, partition V into the set V_{L_1} of unmatched vertices (L for *low*, as they will be placed low in the next iteration, and also for *losers*, or *leftovers*) and the set V_{H_1} of matched vertices (H for *high*, as they will be placed high in the next iteration, and also for *hitters*, or *happy*).

Consider now a permutation π_2 in which V_{L_1} precedes V_{H_1} (preserving the internal order between vertices in V_{L_1} and similarly between vertices in V_{H_1}), and let σ_2 be a permutation on U minimizing the resulting greedy matching. Let $M_2 = M[\pi_2, \sigma_2]$. If $|M_2|/n$ is some constant greater than $1/2$, then terminate with permutation π_2. Else, partition V into the set V_{L_2} of unmatched vertices and the set V_{H_2} of matched vertices, and consider a permutation π_3 in which V_{L_2} precedes V_{H_2} (preserving internal orders). Continue this iterative process until the obtained permutation π_k ensures a matching greater than a half.

The intuition behind this approach is that the unmatched vertices need some "help" in order to be matched, and we provide this help in the form of prioritizing them over their mates. One might hope that this process will reach a good permutation within a constant number of iterations. Unfortunately, we show an example where the process goes through $\log n$ iterations before it first obtains a permutation ensuring a matching that exceeds $n/2$.

[5] It is unclear whether σ_1 can be computed in polynomial time. The related problem of computing a minimum maximal matching in bipartite graphs is known to be NP-hard [2]. However, here we consider the existential problem.

The construction of the graph is inductive. The base is $G_0(U_0, V_0; E_0)$, with two vertices u, v and a single edge between them. For every $i = 1, 2, \ldots$, $G_i(U_i, V_i; E_i)$ is such that $|U_i| = |V_i| = 2^i$; it is obtained by taking two (disjoint) copies of G_{i-1}, with additional edges of the form (u_j, v_j) for every u_j from one copy of G_{i-1} to v_j in the second copy of G_{i-1}. An example of G_3 is presented in Figure 2(a). The iterative process is depicted in Figure 2(a)-(d). In all iterations preceding the last one, exactly $n/2$ vertices are matched in the worst σ.

C Additional Results

The following theorem shows that one cannot hope to get $\rho > 3/4$ with a random permutation in regular graphs.

▶ **Theorem 24.** *For every $\epsilon > 0$ and sufficiently large d, there are d-regular graphs G for which a random permutation π results in $\rho \leq \frac{3}{4} + \epsilon$.*

Proof. Consider a d-regular bipartite graph $G(U, V; E)$, where d is very large, there is a balanced bipartite independent set (S, T) of size $\frac{1-\epsilon}{2}n$, and conditioned on that, G is random. Let Q (a random variable) be the set of first $\frac{1+\epsilon}{2}n$ vertices under the random permutation π. Then, $E[|T \cap (V \setminus Q)|] = (\frac{1-\epsilon}{2})^2 n \simeq \frac{1}{4}n$. W.h.p. there will be a perfect matching between Q and $U \setminus S$. Hence one can choose a permutation σ over U that matches all of $U \setminus S$ to Q. But then the vertices $T \cap (V \setminus Q)$ will remain unmatched. ◀

We also establish a few impossibility results for regular graphs of low degree.

▶ **Theorem 25.** *The following hold:*

- *There exists a 3-regular bipartite graph G for which $\rho(G) = \frac{5}{7}$.*
- *There exists a 4-regular bipartite graph G for which $\rho(G) = \frac{10}{13}$.*

The proof relies on graphs induced by projective planes. A projective plane consists of a set of lines and a set of points, where (among other properties) every two lines intersect in a single point and every two points are incident to a single line. A projective plane induces a bipartite graph $G(U, V; E)$, where every vertex $u \in U$ corresponds to a point in the plane, every vertex $v \in V$ corresponds to a line, and there exists an edge between u and v if the point corresponding to u is incident to the line corresponding to v.

Proof. For the first result, we show that $\rho = \frac{5}{7}$ for the bipartite graph induced by the Fano plane. The Fano plane is a projective plane consisting of 7 points and 7 lines, with 3 points on every line and 3 lines through every point. Consider the 3-regular bipartite graph $G(U, V; E)$ induced by the Fano plane. Let $N(V')$ denote the neighbors of a set $V' \in V$. For every set $V' \in V$ such that $|V'| = 2$, it holds that $|N(V')| = 5$. We show below that for every such V' there exists a perfect matching between $N(V')$ and $V \setminus V'$. Hence one can choose a permutation σ over U whose first 5 vertices are $N(V')$ that will match the vertices of $V \setminus V'$ one by one. Thereafter, the vertices of V' will remain unmatched. By Hall's condition, it suffices to show that for every set $U' \subset N(V')$ such that $|U'| \leq 5$ it holds that $|N(U')| \geq |U'| + 2$ (so that Hall's condition applies with respect to the set $V \setminus V'$). Indeed, for every set U' of size 1, $|N(U')| = 3$, for every set U' of size ≥ 2, $|N(U')| \geq 6$, and for every set U' of size 5, $|N(U')| = 7$. It follows that $\rho(G) = 5/7$.

The second result follows a similar argument. It is known that there exists a projective plane consisting of 13 points and 13 lines, with 4 points on every line and 4 lines through every point. We claim that $\rho = \frac{10}{13}$ for the bipartite graph $G(U, V; E)$ induced by this projective plane. By the properties of a projective plane, for every set $V' \in V$ such that $|V'| = 3$, it

holds that $|N(V')| \in \{9, 10\}$. We show below that for every such V' there exists a perfect matching between $N(V')$ (and possibly an additional vertex u in case $|N(V')| = 9$) and $V \setminus V'$. Hence one can choose a permutation σ over U whose first 10 vertices are $N(V')$ (possibly with the additional vertex) that will match the vertices of $V \setminus V'$ one by one. Thereafter, the vertices of V' will remain unmatched. By Hall's condition, it suffices to show that for every set $U' \subset N(V')$ such that $|U'| \leq 10$ it holds that $|N(U')| \geq |U'| + 3$ (so that Hall's condition applies with respect to the set $V \setminus V'$). Indeed, for every set U' of size 1, $|N(U')| = 4$, for every set U' of size ≥ 2, $|N(U')| \geq 7$, for every set U' of size ≥ 5, $|N(U')| \geq 11$, and for every set U' of size ≥ 9, $|N(U')| = 13$, It follows that $\rho(G) = 10/13$. ◀

Hardy-Muckenhoupt Bounds for Laplacian Eigenvalues

Gary L. Miller
Carnegie Mellon University, Pittsburgh, PA, USA
glmiller@cs.cmu.edu

Noel J. Walkington
Carnegie Mellon University, Pittsburgh, PA, USA
noelw@cmu.edu

Alex L. Wang
Carnegie Mellon University, Pittsburgh, PA, USA
alw1@cs.cmu.edu

Abstract

We present two graph quantities $\Psi(G, S)$ and $\Psi_2(G)$ which give constant factor estimates to the Dirichlet and Neumann eigenvalues, $\lambda(G, S)$ and $\lambda_2(G)$, respectively. Our techniques make use of a discrete Hardy-type inequality due to Muckenhoupt.

2012 ACM Subject Classification Mathematics of computing → Spectra of graphs; Mathematics of computing → Approximation algorithms; Mathematics of computing → Graph algorithms

Keywords and phrases Hardy, Muckenhoupt, Laplacian, eigenvalue, effective resistance

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.8

Category APPROX

Funding *Gary L. Miller*: Supported partially by NSF Grants CCF–1637523 AitF.
Noel J. Walkington: Supported by NSF Grants DMS–1418991 and DMREF–1729478.
Alex L. Wang: Supported partially by NSF Grants CCF–1637523 AitF.

Acknowledgements We would like to thank Timothy Chu for many helpful discussions.

1 Introduction

Possibly one of the most important constants of a graph is λ_2, the fundamental eigenvalue of its graph Laplacian. In computer science, this quantity is used to analyze the mixing time of random walks [14], Markov chains [16], the convergence of Laplacian solvers [11, 18, 20], the performance of spectral clustering [22] and more. The quantity λ_2 is also important in other domains: in quantum mechanics it is related to the uncertainty principles [13], and in numerical analysis arises in the analysis of partial differential equations [2]. As such, it is often necessary to give analytic estimates of this quantity.

In this paper we reexamine an inequality originating with the work of Hardy [9] and show its connection to the eigenvalues of the graph Laplacian. Using this tool, we provide an alternative to Cheeger's inequality and give a 4-approximation of λ_2 in a general setting.

Let $G = (V, E)$ be a connected graph and let $\mu \in \mathbb{R}^V_{>0}$ and $\kappa \in \mathbb{R}^E_{>0}$ be functions on the vertices and edges respectively. We will think of our graphs as spring mass systems where vertex v has mass μ_v and edge e has spring constant κ_e. The Laplacian matrix is defined as $L = D - A$ where D is the weighted diagonal degree matrix and A is the weighted


© Gary L. Miller, Noel J. Walkington, and Alex L. Wang;
licensed under Creative Commons License CC-BY



adjacency matrix. Let M be the diagonal mass matrix. Then, the generalized eigenvalues of L with respect to M have a nice interpretation. Specifically, solutions of the generalized eigenvalue problem

$$Lx = \lambda Mx$$

correspond to modes of vibration of the associated spring mass system. When the spring mass system is connected, λ_2 is the fundamental mode of vibration[1]. In this paper we will refer to λ_2 as the Neumann eigenvalue to emphasize the implicit boundary assumptions. For an introduction to spring mass systems and the Laplacian, see chapter 5 of [21].

Another interpretation of the weighted graph comes from electrical systems. In this interpretation, we will treat κ_e as the conductance of edge e and $\frac{1}{\kappa_e}$ as its resistances. In this paper we will go back and forth between these two interpretations and will refer to κ_e as either a conductance or a spring constant.

The following result, known as Cheeger's inequality, can be traced back to [1, 4, 5]. Define the isoperimetric constant[2] of G to be

$$\Phi(G) = \min_{A} \left\{ \frac{\sum_{e \in E(A,\bar{A})} \kappa_e}{\min(\mu(A), \mu(\bar{A}))} \,\middle|\, A, \bar{A} \neq \varnothing \right\}.$$

Here, and in the rest of the paper, $\bar{A}$ denotes the complement of A

Then in the case of the normalized Laplacian (i.e, when $\mu_v = d_v$, the degree of v), we can bound λ_2 by

$$\frac{\lambda_2}{2} \leq \Phi \leq \sqrt{2\lambda_2}, \qquad \text{or equivalently,} \qquad \frac{\Phi^2}{2} \leq \lambda_2 \leq 2\Phi.$$

It is well known that both sides of the bound are tight up to constants (see [6] for simple examples). Thus we see that Φ fails to give good control over λ_2 when both quantities are small.

In this paper, we introduce the Neumann content, $\Psi_2(G)$, of a graph G (see Section 6 for a formal definition).

$$\Psi_2(G) \approx \min_{A,B} \left\{ \frac{\kappa(A,B)}{\min(\mu(A), \mu(B))} \,\middle|\, A, B \neq \varnothing,\ A \cap B = \varnothing \right\}$$

where $\kappa(A, B)$ is the effective resistance between the sets A and B. When $B = \overline{A}$, it can be shown that $\kappa(A, \overline{A}) = \sum_{e \in E(A,\overline{A})} \kappa_e$, thus the Neumann content can be thought of as a relaxation of the isoperimetric constant. We will show that $\Psi_2(G)$ gives a constant factor estimate of λ_2 even in a much more general setting.

Along the way, we will consider another eigenvalue problem, which we refer to as the Dirichlet problem (see Section 2). This is a variant of the Laplacian eigenvalue problem where we hold a particular boundary set of vertices, S, to zero. In this setting, we will define the Dirichlet content, $\Psi(G, S)$, which allows us to estimate the Dirichlet eigenvalue.

Specifically, we will prove the following theorems.

---

[1] The quantity λ_2 is referred to in the literature under various names: the algebraic connectivity, the Fiedler value, the fundamental eigenvalue, etc.

[2] The quantity Φ is often referred to as the conductance of the graph or the Cheeger constant. In this paper we will refer to Φ as the isoperimetric constant and reserve the term conductance for the conductance of an edge.

▶ **Theorem 1.** *Let (G, S) be a nondegenerate weighted graph with boundary. Let $\lambda(G, S)$ be the Dirichlet eigenvalue and let $\Psi(G, S)$ be the Dirichlet content of (G, S). Then*

$$\frac{\Psi}{4} \leq \lambda \leq \Psi.$$

▶ **Theorem 2.** *Let G be a nondegenerate weighted graph. Let $\lambda_2(G)$ be the Neumann eigenvalue and let $\Psi_2(G)$ be the Neumann content of G. Then,*

$$\frac{\Psi_2}{4} \leq \lambda_2 \leq \Psi_2.$$

It can be shown that the constants in both of these theorems are optimal. In particular, there exist nondegenerate weighted graphs (with and without boundary) for which $\lambda(G, S) = \Psi(G, S)$ and $\lambda_2(G) = \Psi_2(G)$. This shows that the constant 1 in the upper bound is optimal. There also exist sequences of nondegenerate weighted graphs (with and without boundary) for which $\frac{\lambda(G,S)}{\Psi(G,S)} \to \frac{1}{4}$ and $\frac{\lambda_2(G)}{\Psi_2(G)} \to \frac{1}{4}$. This shows that the constant $\frac{1}{4}$ in the lower bound is optimal. See Appendex A for these constructions.

▶ Remark 3. The proof strategy we apply is general and the theorems can be extended to the p-Laplacian[3] for $1 < p < \infty$. The proofs for the case of a general p are almost identical to the proofs for the case of $p = 2$, which we present in this paper, and thus will be omitted. More specifically, with the appropriate definitions for the Dirichlet and Neumann p-contents, both theorem statements above will hold after replacing the 4 in the denominator of the lower bound with $pq^{p/q}$, where q is the Hölder dual of p. The constants in this setting are also optimal.

1.1 Related work

A very recent independent paper [19] introduced a quantity $\rho(G)$ specifically in the case of the normalized Laplacian, i.e., when $\mu_v = d_v$. In this setting, the Neumann content $\Psi_2(G)$ is equivalent to the definition of $\rho(G)$ up to constant factors: $\frac{\Psi_2}{2} \leq \rho \leq \Psi_2$. In [19], it is proved that

$$\frac{\rho}{25600} \leq \lambda_2 \leq 2\rho.$$

This is a special subcase of our Theorem 35 with weaker constants.

The application of the Hardy-Muckenhoupt inequality to estimating the Dirichlet eigenvalue was noted in [15]. In that paper, the authors showed how to bound the Dirichlet eigenvalue on an infinite path graph by the (infinite path analogue of) Ψ. Specifically,

$$\frac{\Psi}{4} \leq \lambda \leq 2\Psi.$$

This is a special subcase of our Theorem 28 with weaker constants.

In contrast with the above related work, we can show that our constants are optimal (see Appendix A).

Other methods for estimating λ_2 have been proposed. A method for lower bounding λ_2 based on path embeddings is presented in [7, 8, 10]. In this method, a graph with known eigenstructure is embedded into a host graph. Then the fundamental eigenvalue of the host graph can be estimated in terms of the eigenstructure of the embedded graph and the "distortion" of the embedding. For a review of path embedding methods, see the introduction in [8].

[3] We refer the curious reader to [3] for basic background on this topic.

1.2 Roadmap

In section 2, we set notation and discuss background related to weighted graphs, Laplacians, the eigenvalue problems, interpreting graphs as electrical networks, and minimum energy extensions. In section 3, we introduce Muckenhoupt's weighted Hardy inequality. In section 4, we introduce the Hardy quantity and the Dirichlet content and show how Muckenhoupt's result can be used to bound the Dirichlet eigenvalue on a path graph. In section 5, we extend the bounds on the Dirichlet eigenvalue from path graphs to arbitrary graphs. Finally in section 6, we introduce the two-sided Hardy quantity and the Neumann content and extend the bounds on the Dirichlet eigenvalue on a graph to the Neumann eigenvalue on a graph.

2 Preliminaries

2.1 Miscellaneous notation

For $A \subseteq V$, we denote by $\overline{A}$ the complement of A in V.

2.2 Vertex and edge weighted graphs

We collect some definitions and notation we will use related to weighted graphs.

▶ **Definition 4.** *A weighted graph is $G = (V, E, \mu, \kappa)$ where (V, E) forms an undirected graph, $\mu \in \mathbb{R}^V_{\geq 0}$ and $\kappa \in \mathbb{R}^E_{>0}$. We call μ_v the mass of vertex v and κ_e the conductance[4] of edge e.*

▶ **Definition 5.** *A weighted graph with boundary is a pair (G, S) where G is a weighted graph and $S \subseteq V$ is a proper nonempty subset of the vertices.*

We will make the following assumptions on our graphs. This will ensure that the appropriate eigenvalue quantities exist and are nonzero.

▶ **Definition 6.** *A nondegenerate weighted graph is a weighted graph $G = (V, E, \mu, \kappa)$ where (V, E) forms a connected graph, $|V| \geq 2$, and $\mu \in \mathbb{R}^V_{>0}$.*

▶ **Definition 7.** *A nondegenerate weighted graph with boundary is a weighted graph with boundary (G, S) where every connected component of G contains some $s \in S$, $\left|\overline{S}\right| \geq 1$, and $\mu_v > 0$ for all $v \in \overline{S}$.*

For notational simplicity, we extend μ to subsets of vertices, $\mu(A) = \sum_{v \in A} \mu_v$.

2.3 Laplacians

▶ **Definition 8.** *Let G be a weighted graph. Define $d_v = \sum_{(u,v) \in E} \kappa_{(u,v)}$ to be the degree of vertex v. Let D be the diagonal degree matrix $D_{v,v} = d_v$. Let $A \in \mathbb{R}^{V \times V}$ be the adjacency matrix of G, i.e. $A_{u,v} = \kappa_{(u,v)}$ if $(u, v) \in E$ and 0 otherwise. Then the Laplacian matrix corresponding to G is*

$$L = D - A.$$

[4] As we are dealing with spring mass systems, perhaps it would be better to refer to these quantities as spring constants and compliances. Nonetheless, we have chosen to refer to these quantities as conductances and resistances as this is the terminology most commonly found in the spectral graph theory literature.

Note that the quadratic form associated with L is

$$x^\top Lx = \sum_{(u,v)\in E} \kappa_{(u,v)}(x_u - x_v)^2.$$

▶ **Definition 9.** *Let G be a weighted graph. The mass matrix corresponding to G is the diagonal matrix $M(G)$ where $M_{v,v} = \mu_v$.*

2.4 The generalized Laplacian eigenvalue problem

▶ **Definition 10.** *Let G be a nondegenerate weighted graph. Let $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_{|V|}$ be the generalized eigenvalues of L with respect to M. We refer to λ_2 as the Neumann eigenvalue of G, denoted $\lambda_2(G)$ and we refer to an associated eigenvector as a Neumann eigenvector.*

Nondegeneracy ensures that $\lambda_2(G)$ exists as $|V| \ge 2$ and $\lambda_2(G) > 0$ by connectivity.

We state a version of the Courant-Fischer min-max theorem. This will allow us to give variational characterizations of eigenvalues.

▶ **Theorem 11** (Courant-Fischer). *Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric matrices and suppose $B \succ 0$. Let $\lambda_1 \le \cdots \le \lambda_n$ be the ordered generalized eigenvalues of A with respect to B. Let $k \in [n]$ and let S denote a subspace of $\mathbb{R}^n$. Then,*

$$\lambda_k = \min_S \max_x \left\{ \frac{x^\top Ax}{x^\top Bx} \,\middle|\, \dim(S) = k,\, x \in S,\, x \neq 0 \right\}.$$

Furthermore, suppose $v_1, \ldots, v_{k-1}$ are orthogonal eigenvectors corresponding to $\lambda_1, \ldots, \lambda_{k-1}$ then

$$\lambda_k = \min_x \left\{ \frac{x^\top Ax}{x^\top Bx} \,\middle|\, \begin{array}{l} x^\top Bv_i = 0,\, \forall i \in [k-1] \\ x \neq 0 \end{array} \right\}$$

and x is a generalized eigenvector with eigenvalue λ_k if and only if x is a minimizer of this second expression.

Noting that 1, the all ones vector, is a generalized eigenvector of L with respect to M with eigenvalue 0, we may apply the Courant-Fischer theorem to get a variational characterization of the Neumann eigenvalue and its eigenvectors.

▶ **Lemma 12.** *Let G be a nondegenerate weighted graph. Then*

$$\lambda_2(G) = \min_{x\in\mathbb{R}^V} \left\{ \frac{x^\top Lx}{x^\top Mx} \,\middle|\, x^\top M1 = 0,\, x \neq 0 \right\}.$$

Furthermore, x is a Neumann eigenvector of G if and only if x is a minimizer in this optimization problem.

The expression $\frac{x^\top Lx}{x^\top Mx}$ plays a large role in our analysis. This quantity is known as the Rayleigh quotient.

We will also consider the Laplacian eigenvalue problem on weighted graphs with boundaries. This corresponds to fixing the value of x at the boundary to zero.

▶ **Definition 13.** *Let (G, S) be a nondegenerate weighted graph with boundary. Let $L_{\overline{S}}$ be the submatrix of L associated with the complement of S and let $M_{\overline{S}}$ be the corresponding submatrix of M. Let $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_{|\overline{S}|}$ be the generalized eigenvalues of*

$$L_{\overline{S}}x = \lambda M_{\overline{S}}x.$$

We refer to λ_1 as the Dirichlet eigenvalue of (G, S), denoted $\lambda(G, S)$. Let $x \in \mathbb{R}^{\overline{S}}$ be an associated eigenvector and extend it to $\mathbb{R}^V$ by zeros. We will refer to $x \in \mathbb{R}^V$ as a Dirichlet eigenvector.

Nondegeneracy ensures that $\lambda(G, S)$ exists as $\left|\overline{S}\right| \geq 1$ and $\lambda(G, S) > 0$ by connectivity.

Again by Courant-Fischer, we can give a variational characterization of the Dirichlet eigenvalue.

▶ **Lemma 14.** *Let (G, S) be a nondegenerate weighted graph with boundary. Then,*

$$\lambda(G, S) = \min_{x \in \mathbb{R}^V} \left\{ \frac{x^\top L x}{x^\top M x} \,\middle|\, x\restriction_S = 0,\ x \neq 0 \right\}.$$

Furthermore, x is a Dirichlet eigenvalue if and only if x is a minimizer in this optimization problem.

Note that the masses of vertices $s \in S$ play no role in either characterization. We will often neglect to assign masses to vertices in the boundary when convenient.

2.5 Graphs as electrical networks

Given a weighted graph G, we can think of its edges as electrical conductors where edge e has conductance κ_e. Thinking of $x \in \mathbb{R}^V$ as an assignment of voltages to the vertices of our electrical network, we have that

$$x^\top L x = \sum_{(u,v) \in E} \kappa_{(u,v)} (x_u - x_v)^2$$

is the power dissipated in our system. Drawing inspiration from physics, we define the effective resistance between two sets of vertices in terms of the minimum power required to maintain a unit voltage drop.

▶ **Definition 15.** *Let G be a weighted graph and let $A, B \subseteq V$ be disjoint nonempty sets such that there exists a path between a and b for some $a \in A$ and $b \in B$. The effective resistance between A and B, denoted $R(A, B)$, and the effective conductance between A and B, denoted $\kappa(A, B)$, are the quantities such that*

$$\frac{1}{R(A, B)} = \kappa(A, B) = \min_{x \in \mathbb{R}^V} \left\{ x^\top L x \,\middle|\, x\restriction_A = 0,\ x\restriction_B = 1 \right\}.$$

The quantity on the right is well-defined as $x^\top L x$ is continuous and without loss of generality, we may optimize over $x \in [0, 1]^V$, a compact set. Then, by the connectivity assumption, $\kappa(A, B) \in (0, \infty)$. Thus, $R(A, B)$ is also well-defined.

If $A = \{a\}$ is a single element, we will opt to write $R(a, B)$ instead of the more cumbersome $R(\{a\}, B)$. Similarly we will write $R(A, b)$ or $R(a, b)$ where appropriate.

▶ Remark 16. When $A = \{a\}$ and $B = \{b\}$ are singleton sets, this definition agrees with the standard definition $R(a, b) = \chi_{a,b} L^+ \chi_{a,b}$. In general, we can define $R(A, B)$ in a different way. Consider contracting all vertices in A to a single vertex v_A and all vertices to a single vertex v_B. Then $R(A, B)$ is the effective resistance between v_A and v_B in the new graph. This is the definition given in [19].

2.6 Splitting edges and minimum energy extensions

Let G be a weighted graph. At times, we will split edges using vertices with zero mass. This can be done without affecting the variational quantities[5].

▶ **Lemma 17.** *Let $\alpha_i > 0$ such that $\sum_{i=1}^{k} \alpha_i = 1$. Let $\kappa > 0$ and let $\kappa_i = \kappa/\alpha_i$. Let $y_0, y_k \in \mathbb{R}$ be fixed. Then*

$$\min_{y_1,\ldots,y_{k-1}} \sum_{i=1}^{k} \kappa_i (y_i - y_{i-1})^2 = \kappa (y_k - y_0)^2.$$

Furthermore, the unique optimum is achieved by $y_i^* = y_0 + \left(\sum_{j=1}^{i} \alpha_j\right)(y_k - y_0)$.

Proof. Note that $\sum_{i=1}^{k} \kappa_i (y_i - y_{i-1})^2$ is a strictly convex function as y_0 and y_k are fixed. Thus if suffices to show that y^* is a local optimum. Differentiating with respect to y_j and evaluating at y^*,

$$\begin{aligned}\frac{\partial}{\partial y_j}\left(\sum_{i=1}^{k} \kappa_i (y_i - y_{i-1})^2\right)_{y=y^*} &= 2\left(\kappa_j (y_j^* - y_{j-1}^*) - \kappa_{j+1}(y_{j+1}^* - y_j^*)\right) \\ &= 2\left(\frac{\kappa}{\alpha_j}\alpha_j - \frac{\kappa}{\alpha_{j+1}}\alpha_{j+1}\right)(y_k - y_0) = 0.\end{aligned}$$

Then y^* is the unique minimizer achieving objective value

$$\sum_{i=1}^{k} \kappa_i (y_i^* - y_{i-1}^*)^2 = \kappa (y_k - y_0)^2 \sum_{i=1}^{k} \alpha_i = \kappa (y_k - y_0)^2. \qquad \blacktriangleleft$$

Let G be a weighted graph and consider an edge (a, b) of conductance κ in G. Given $\alpha_i > 0$ summing to 1, we can split the edge (a, b) into k edges by inserting $k-1$ new vertices, removing the edge (a, b), and inserting edges $(a, c_1), (c_1, c_2), \ldots, (c_{k-1}, b)$ with conductances according to the lemma. We will assign $\mu'(v) = 0$ for all newly added vertices. Let this new weighted graph be $G' = (V', E', \mu', \kappa')$.

▶ **Definition 18.** *Let G be a weighted graph and let G' be a weighted graph constructed from G by splitting edges using the procedure described above. Let $x \in \mathbb{R}^V$. The minimum energy extension of x to V' is the vector y given by*

$$y = \operatorname*{arg\,min}_{y \in \mathbb{R}^{V'}} \left\{ y^\top L' y \,\middle|\, y\restriction_V = x \right\}.$$

Then by the above lemma it is immediate that $\min_{y \in \mathbb{R}^{V'}} \left\{ y^\top L' y \,\middle|\, y\restriction_V = x \right\} = x^\top L x$. Thus, as $\mu'(v) = 0$ for all $v \in V' \setminus V$, we have,

$$\begin{aligned}&\min_{y \in \mathbb{R}^{V'}} \left\{ \frac{y^\top L' y}{y^\top M' y} \,\middle|\, y^\top M 1 = 0,\, y \neq 0 \right\} \\ &\quad = \min_{x \in \mathbb{R}^V} \left\{ \frac{\min_{y \in \mathbb{R}^{V'}} \left\{ y^\top L' y \,\middle|\, y\restriction_V = x \right\}}{x^\top M x} \,\middle|\, x^\top M 1 = 0,\, x \neq 0 \right\} \\ &\quad = \min_{x \in \mathbb{R}^V} \left\{ \frac{x^\top L x}{x^\top M x} \,\middle|\, x^\top M 1 = 0,\, x \neq 0 \right\}.\end{aligned}$$

[5] In fact, this can be done without affecting the eigenvalue quantities provided they exist. However, proving this requires more set up than is given in this paper.

Similarly, if S is a proper nonempty subset of V, then

$$\begin{aligned}
&\min_{y\in\mathbb{R}^{V'}}\left\{\frac{y^\top L'y}{y^\top M'y}\,\middle|\, y\restriction_S = 0,\, y\neq 0\right\}\\
&\quad=\min_{x\in\mathbb{R}^V}\left\{\frac{\min_{y\in\mathbb{R}^{V'}}\left\{y^\top L'y\,\middle|\, y\restriction_V = x\right\}}{x^\top Mx}\,\middle|\, x\restriction_S = 0,\, x\neq 0\right\}\\
&\quad=\min_{x\in\mathbb{R}^V}\left\{\frac{x^\top Lx}{x^\top Mx}\,\middle|\, x\restriction_S = 0,\, x\neq 0\right\}.
\end{aligned}$$

▶ **Definition 19.** *Let G be a nondegenerate weighted graph and let G' be a weighted graph constructed from G using the procedure described above. The Neumann eigenvalue of G' is*

$$\lambda_2(G') = \min_{y\in\mathbb{R}^{V'}}\left\{\frac{y^\top L'y}{y^\top M'y}\,\middle|\, y^\top M1 = 0,\, y\neq 0\right\}.$$

A vector y is a Neumann eigenvector of G' if y is a minimizer of this optimization problem.

▶ **Definition 20.** *Let (G,S) be a nondegenerate weighted graph with boundary and let G' be a weighted graph constructed from G using the procedure described above. The Dirichlet eigenvalue of (G',S) is*

$$\lambda(G',S) = \min_{y\in\mathbb{R}^{V'}}\left\{\frac{y^\top L'y}{y^\top M'y}\,\middle|\, y\restriction_S = 0,\, y\neq 0\right\}$$

A vector y is a Dirichlet eigenvector of (G',S) if y is a minimizer of this optimization problem.

3 Muckenhoupt's weighted Hardy inequality

The following theorem, due[6] to Muckenhoupt [17], relates the $L^2(\mathbb{R}_{\geq 0},\kappa)$ norm of a function and the $L^2(\mathbb{R}_{\geq 0},\mu)$ norm of the "running integral" of the function.[7] In other words, this theorem characterizes the boundedness of the Hardy operator. In this paper we will refer to this inequality as Muckenhoupt's weighted Hardy inequality (see [12] for a more thorough account of the development and history of the Hardy inequality).

▶ **Theorem 21** (Muckenhoupt 1972)**.** *Let μ, κ be measurable functions from $\mathbb{R}_{\geq 0}$ to $\mathbb{R}_{>0}$. Let C be the smallest (possibly infinite) constant such that for all $f\in L^1_{loc}(\mathbb{R}_{\geq 0})$,*

$$\int_0^\infty \mu(x)\left(\int_0^x f(t)\,dt\right)^2 dx \leq C\int_0^\infty \kappa(x)f(x)^2\,dx.$$

Let

$$B = \sup_{r>0}\left(\int_r^\infty \mu(x)\,dx\right)\left(\int_0^r \frac{1}{\kappa(x)}\,dx\right).$$

Then $B\leq C\leq 4B$. In particular, C is finite if and only if B is finite.

---

[6] A similar theorem may have been known previous to Muckenhoupt. Indeed [17] cites the work of Tomaselli and Artola, however we were unable to obtain copies of these papers.

[7] The original theorem deals more generally with L_p norms and Borel measures – see [17].

We will state and prove a finite, discrete version of the above inequality in the following section. Our proof will be stated in the language of graph Laplacians but closely follows the structure of [15, 17] and is only included for completeness.

We first sketch, non-rigorously, why this theorem may be useful. Suppose we have a differentiable function g satisfying $g(0) = 0$. Then taking $f = \frac{d}{dx}g$, and rearranging the above theorem, we have that

$$\frac{1}{C} \leq \frac{\int_0^\infty \kappa(x)g'(x)^2\,dx}{\int_0^\infty \mu(x)g(x)^2\,dx} \approx \frac{\sum_{i=1}^\infty \kappa_i(g_i - g_{i-1})^2}{\sum_{i=1}^\infty \mu_i g_i^2}.$$

Note that the right hand side is the Rayleigh quotient of the Laplacian of a weighted infinite path graph (cf. Lemma 14 above). Then minimizing over g, we have that $1/C$ corresponds to a Dirichlet eigenvalue and the bound $B \leq C \leq 4B$ allows us to estimate this eigenvalue.

4 The Dirichlet problem on path graphs

Throughout this section, let $P = (V, E, \mu, \kappa)$ be a weighted path graph. Let the vertices be numbered $\{v_0, v_1, \ldots, v_N\}$ for some $N \geq 1$ and let the boundary set be $S = \{v_0\}$. Let the edges be $E = \{(v_i, v_{i-1}) \,|\, i \in [N]\}$ and let edge (v_i, v_{i-1}) have conductance $\kappa_i > 0$. Let vertex v_i have mass $\mu_i > 0$.

It is immediate that (G, S) is a nondegenerate weighted graph with boundary.

4.1 The Hardy quantity and the Dirichlet content

Let $A \subseteq V \setminus S$ be a set of vertices disjoint from the boundary. Consider the graph consisting of two vertices v_S, v_A. Let v_A have mass $\mu(A)$ and let the edge (v_S, v_A) has conductance $\kappa(S, A)$. Then the Dirichlet eigenvalue of this two node system with boundary set $\{v_S\}$ is given by $\frac{\kappa(S,A)}{\mu(A)}$. We will define the Dirichlet content of G, $\Psi(G)$, to be the minimum such quantity over choices of A and, for historical reasons, we will define the Hardy quantity to be $H = \Psi^{-1}$.

▶ **Definition 22.** *Let (G, S) be a nondegenerate weighted graph with boundary. The Dirichlet content of (G, S), denoted $\Psi(G, S)$, is*

$$\Psi(G, S) = \min_{A \subseteq V} \left\{ \frac{\kappa(S, A)}{\mu(A)} \,\middle|\, A \neq \varnothing,\, A \cap S = \varnothing \right\}.$$

▶ **Definition 23.** *Let (G, S) be a nondegenerate weighted graph with boundary. The Hardy quantity of (G, S), denoted $H(G, S)$, is $H(G, S) = \Psi(G, S)^{-1}$, i.e.*

$$H(G, S) = \max_{A \subseteq V} \{R(S, A)\mu(A) \,|\, A \neq \varnothing,\, A \cap S = \varnothing\}.$$

In a path graph, it suffices to optimize over tail sets. This gives us a second characterization of H (and thus Ψ) on path graphs.

▶ **Lemma 24.** *Let (P, v_0) be a nondegenerate weighted path graph with boundary. Let $A_k = \{v_i \,|\, i \geq k\}$ be the tail set beginning at v_k. Then*

$$H(P, v_0) = \max_{1 \leq k \leq N} R(v_0, A_k)\mu(A_k).$$

Proof. Let $A \subseteq V \setminus S$. Let $k = \min\{i \,|\, v_i \in A\}$ be the minimum element in A. Then $R(S, A) = R(S, A_k)$ and $\mu(A_k) \geq \mu(A)$ thus $R(S, A_k)\mu(A_k) \geq R(S, A)\mu(A)$. ◀

For a path graph, we have the following closed form expression for the resistance between v_0 and A_k. This is a consequence of Lemma 17.

▶ **Lemma 25.** *In a weighted path graph,* $R(v_0, A_k) = \sum_{i=1}^{k} \frac{1}{\kappa_i}$.

4.2 Bounding the Dirichlet eigenvalue

▶ **Theorem 26.** *Let* (P, v_0) *be a nondegenerate weighted path graph with boundary. Let* $\lambda(P, v_0)$ *be the Dirichlet eigenvalue and let* $H(P, v_0)$ *be the Hardy quantity of* (P, v_0)*. Then,*

$$\frac{1}{4H} \leq \lambda \leq \frac{1}{H}.$$

We reiterate that the below proof has been known since [17] and is included only for completeness.

Proof. We begin by proving the upper bound. Note that if $x\restriction_A = 1$, then $x^\top M x \geq \mu(A)$ for any $A \subseteq V \setminus S$. Applying this bound to λ, we note that the numerator of the Rayleigh quotient becomes an effective conductance term.

$$\begin{aligned}
\lambda &= \min_x \left\{ \frac{x^\top L x}{x^\top M x} \,\middle|\, x_0 = 0,\, x \neq 0 \right\} \\
&\leq \min_{1 \leq k \leq N} \min_x \left\{ \frac{x^\top L x}{x^\top M x} \,\middle|\, x_0 = 0,\, x\restriction_{A_k} = 1 \right\} \\
&\leq \min_{1 \leq k \leq N} \frac{1}{\mu(A_k)} \min_x \left\{ x^\top L x \,\middle|\, x_0 = 0,\, x\restriction_{A_k} = 1 \right\} \\
&= \min_{1 \leq k \leq N} \frac{\kappa(S, A_k)}{\mu(A_k)} \\
&= H^{-1}.
\end{aligned}$$

We turn to the lower bound. Begin by rewriting x_i as a sum of differences in the denominator. Let $\alpha_j > 0$ to be fixed later. We apply Cauchy-Schwarz,

$$\begin{aligned}
\sum_{i=1}^{n} \mu_i x_i^2 &= \sum_{i=1}^{n} \mu_i \left(\sum_{j=1}^{i} (x_j - x_{j-1}) \left(\frac{\kappa_j}{\alpha_j} \right)^{1/2} \left(\frac{\alpha_j}{\kappa_j} \right)^{1/2} \right)^2 \\
&\leq \sum_{i=1}^{n} \mu_i \left(\sum_{j=1}^{i} (x_j - x_{j-1})^2 \frac{\kappa_j}{\alpha_j} \right) \left(\sum_{j=1}^{i} \frac{\alpha_j}{\kappa_j} \right).
\end{aligned}$$

Let $y_j = \left(\sum_{i=1}^{j} \frac{1}{\kappa_i} \right)^{1/2}$ and $y_0 = 0$. We will pick[8] $\alpha_j = \kappa_j (y_j - y_{j-1})$. Thus, plugging in this choice of α_j, noticing the telescoping sum and reversing the order of summation,

[8] This choice ensures that Cauchy-Schwarz is tight when $x = y$ and corresponds to the intuition that the true eigenvector is "close to" y.

$$\begin{aligned}\dots &\leq \sum_{i=1}^{n} \mu_i \left(\sum_{j=1}^{i} (x_j - x_{j-1})^2 \frac{\kappa_j}{\alpha_j} \right) y_i \\ &= \sum_{j=1}^{n} \kappa_j (x_j - x_{j-1})^2 \frac{1}{\alpha_j} \sum_{i=j}^{n} \mu_i y_i \\ &\leq \left(\sum_{j=1}^{n} \kappa_j (x_j - x_{j-1})^2 \right) \left(\max_{1 \leq j \leq n} \frac{1}{\alpha_j} \sum_{i=j}^{n} \mu_i y_i \right).\end{aligned}$$

It remains to bound the final term. Note that $y_j = R(v_0, A_j)^{1/2} \leq H^{1/2} \mu(A_k)^{-1/2}$. Then,

$$\sum_{i=j}^{n} \mu_i y_i \leq H^{1/2} \sum_{i=j}^{n} \mu_i \left(\sum_{k=i}^{n} \mu_k \right)^{-1/2}.$$

Note that if $A, a \geq 0$, then $a(A+a)^{-1/2} \leq 2\left((A+a)^{1/2} - A^{1/2}\right)$. Indeed, this holds by noting that $(A+ta)^{1/2}$ is concave: $a(A+a)^{-1/2} = \frac{d}{dt}\left(2(A+ta)^{1/2}\right)_{t=1} \leq 2\left((A+a)^{1/2} - A^{1/2}\right)$.

Then, taking $A = \sum_{k=i+1}^{n} \mu_k$ and $a = \mu_i$ in this inequality, we get

$$\begin{aligned}\sum_{i=j}^{n} \mu_i y_i &\leq 2H^{1/2} \left(\sum_{i=j}^{n-1} \left(\mu(A_i)^{1/2} - \mu(A_{i+1})^{1/2} \right) + \mu(A_n)^{1/2} \right) \\ &= 2H^{1/2} \mu(A_j)^{1/2}.\end{aligned}$$

We will use the inequality once more. This time, take $A = \sum_{k=1}^{j-1} \frac{1}{\kappa_k}$ and $a = \frac{1}{\kappa_j}$. Then,

$$\begin{aligned}\alpha_j &= \kappa_j (y_j - y_{j-1}) \\ &= \kappa_j \left((A+a)^{1/2} - A^{1/2} \right) \\ &\geq \kappa_j \left(\frac{a}{2} (A+a)^{-1/2} \right) \\ &= \frac{1}{2} R(v_0, A_j)^{-1/2}.\end{aligned}$$

Finally,

$$\begin{aligned}\max_{1 \leq j \leq n} \frac{1}{\alpha_j} \sum_{i=j}^{n} \mu_i y_i &\leq 4H^{1/2} \max_{1 \leq j \leq n} \left(\mu(A_j) R(v_0, A_j) \right)^{1/2} \\ &\leq 4H.\end{aligned}$$

Rearranging completes the proof. ◀

The following theorem follows as a corollary.

▶ **Theorem 27.** *Let (P, v_0) be a nondegenerate weighted path graph with boundary. Let $\lambda(P, v_0)$ be the Dirichlet eigenvalue and let $\Psi(P, v_0)$ be the Dirichlet content of (P, v_0). Then,*

$$\frac{\Psi}{4} \leq \lambda \leq \Psi.$$

5 The Dirichlet problem on general graphs

5.1 Bounding the Dirichlet eigenvalue

▶ **Theorem 28.** *Let (G, S) be a nondegenerate weighted graph with boundary. Let $\lambda(G, S)$ be the Dirichlet eigenvalue and let $H(G, S)$ be the Hardy quantity of G. Then*

$$\frac{1}{4H} \leq \lambda \leq \frac{1}{H}.$$

The proof of the upper bound in the graph case is the same as the proof of the upper bound in the path case. To prove the lower bound, we split edges by inserting zero mass vertices. We then treat the new graph as a path graph.

Proof. The upper bound follows immediately.

$$\begin{aligned}
\lambda &= \min_x \left\{ \frac{x^\top L x}{x^\top M x} \,\middle|\, x\restriction_S = 0,\, x \neq 0 \right\} \\
&\leq \min_{A \subseteq V,\, x} \left\{ \frac{x^\top L x}{x^\top M x} \,\middle|\, A \neq \varnothing,\, A \cap S = \varnothing,\, x\restriction_S = 0,\, x\restriction_{A_k} = 1 \right\} \\
&\leq \min_{A \subseteq V} \left\{ \frac{\kappa(S, A)}{\mu(A)} \,\middle|\, A \neq \varnothing,\, A \cap S = \varnothing \right\} \\
&= H^{-1}.
\end{aligned}$$

We turn to the lower bound. We construct a new weighted graph $G' = (V', E', \mu', \kappa')$ from G as follows. Let x be a Dirichlet eigenvector corresponding to $\lambda(G, S)$. Without loss of generality x is nonnegative. Let $0 = l_0 < \cdots < l_N$ be the distinct values of x. For each edge $(a, b) \in E$ such that $x_a = l_i < l_{i+1} < l_j = x_b$, split e into $j - i$ segments such that in the minimum energy extension of x, the new vertices on e take on all intermediate values $l_{i+1}, \ldots, l_{j-1}$. This is possible by Lemma 17 and the discussion following it. Let y be the minimum energy extension of x.

Let $\tilde{v}_i = \{v \in V' \,|\, y_v = l_i\}$, let $\tilde{A}_k = \{v \in V' \,|\, y_v \geq l_k\}$. Let $\tilde{\kappa}_i = \sum_{u \in \tilde{v}_i,\, v \in \tilde{v}_{i-1}} \kappa'_{(u,v)}$ be the conductance between $\tilde{v}_i$ and $\tilde{v}_{i-1}$. Let $\tilde{\mu}_i = \mu'(\tilde{v}_i)$. Note that $S \subseteq \tilde{v}_0$. Then,

$$\begin{aligned}
\lambda(G, S) &= \lambda(G', S) \\
&= \min_{y \in \mathbb{R}^{V'}} \left\{ \frac{y^\top L' y}{y^\top M' y} \,\middle|\, y\restriction_S = 0,\, x \neq 0 \right\} \\
&= \min_{z \in \mathbb{R}^N} \left\{ \frac{\sum_{i=1}^N \tilde{\kappa}_i (z_i - z_{i-1})^2}{\sum_{i=1}^N \tilde{\mu}_i z_i^2} \,\middle|\, z_0 = 0,\, z \neq 0 \right\}.
\end{aligned}$$

Equality in the last line follows by taking $z_i = l_i$. Then note that the objective function in the final optimization problem is the Rayleigh quotient of a nondegenerate weighted path graph with boundary with vertices $\tilde{v}_i$, conductances $\tilde{\kappa}_i$, and boundary set $\tilde{v}_0$. Then applying the lower bound of Theorem 26.

$$\begin{aligned}
\lambda(G,S) &\geq \frac{1}{4}\min_{1\leq k\leq N}\frac{\min_{z\in\mathbb{R}^N}\left\{\sum_{i=1}^{n}\tilde{\kappa}_i(z_i-z_{i-1})^2 \,\middle|\, z_0=0,\, z\restriction_{\{k,\ldots,N\}}=1\right\}}{\sum_{i=k}^{N}\tilde{\mu}_i}\\
&\geq \frac{1}{4}\min_{1\leq k\leq N}\frac{\min_{y\in\mathbb{R}^{V'}}\left\{y^\top L' y \,\middle|\, y\restriction_S=0,\, y\restriction_{\tilde{A}_k}=1\right\}}{\mu'\left(\tilde{A}_k\right)}\\
&= \frac{1}{4}\min_{1\leq k\leq N}\frac{\kappa'\left(S,\tilde{A}_k\right)}{\mu'\left(\tilde{A}_k\right)}\\
&\geq \frac{1}{4}\min_{A'\subseteq V'}\left\{\frac{\kappa'(S,A')}{\mu'(A')}\,\middle|\, A'\neq\varnothing,\, A'\cap S=\varnothing\right\}.
\end{aligned}$$

Note that for any $A' \subseteq V'$, we can take $A = A' \cap V$. For this choice of A, we have $\mu(A) = \mu'(A')$ and $\kappa'(S, A') \geq \kappa(S, A)$. Thus,

$$\begin{aligned}
\lambda(G,S) &\geq \frac{1}{4}\min_{A\subseteq V}\left\{\frac{\kappa(S,A)}{\mu(A)}\,\middle|\, A\neq\varnothing,\, A\cap S=\varnothing\right\}\\
&= \frac{1}{4H}.
\end{aligned}$$

◀

The following theorem follows as a corollary.

▶ **Theorem 29.** *Let (G, S) be a nondegenerate weighted graph with boundary. Let $\lambda(G, S)$ be the Dirichlet eigenvalue and let $\Psi(G, S)$ be the Dirichlet content of G. Then*

$$\frac{\Psi}{4} \leq \lambda \leq \Psi.$$

6 The Neumann problem on general graphs

Throughout this section, let $G = (V, E, \mu, \kappa)$ be a nondegenerate weighted graph.

6.1 The two-sided Hardy quantity and the Neumann content

Let $A, B \subseteq V$ be disjoint nonempty sets. Consider the graph consisting of two vertices v_A, v_B where vertex v_A has mass $\mu(A)$, vertex v_B has mass $\mu(B)$ and the edge (v_A, v_B) has conductance $\kappa(A, B) > 0$. Then the Neumann eigenvalue of this two node system is given by $\frac{\kappa(A,B)}{(\mu(A)^{-1}+\mu(B)^{-1})^{-1}}$. We will define the Neumann content of G, $\Psi_2(G)$, to be the minimum such quantity over choices of A and B. For historical reasons, we will define the two-sided Hardy quantity to be $H_2 = \Psi_2^{-1}$.

▶ **Definition 30.** *Let G be a nondegenerate weighted graph. The Neumann content of G, denoted $\Psi_2(G)$, is*

$$\Psi_2(G) = \min_{A,B\subseteq V}\left\{\frac{\kappa(A,B)}{(\mu(A)^{-1}+\mu(B)^{-1})^{-1}}\,\middle|\, A,B\neq\varnothing,\, A\cap B=\varnothing\right\}.$$

▶ **Definition 31.** *Let G be a nondegenerate weighted graph. The two-sided Hardy quantity of G, denoted $H_2(G)$, is $H_2(G) = \Psi_2(G)^{-1}$, i.e.,*

$$H_2 = \max_{A,B\subseteq V}\left\{\frac{R(A,B)}{\mu(A)^{-1}+\mu(B)^{-1}}\,\middle|\, A,B\neq\varnothing,\, A\cap B=\varnothing\right\}.$$

6.2 Bounding the Neumann eigenvalue

In this section we show how to extend the bounds on the Dirichlet eigenvalue to the Neumann eigenvalue.

We will bound the Neumann eigenvalue by applying Courant-Fischer to a carefully chosen two-dimensional subspace. In particular, we will split our graph into two parts sharing a common boundary. We will then take our two-dimensional subspace to be the linear span of solutions to the Dirichlet problem on either side of this boundary.

Let $f \in \mathbb{R}^V$ such that f takes both positive and negative values. We will write this concisely as $\pm f \notin \mathbb{R}^V_{\geq 0}$. We will "pinch" the graph at the zero level set of f to create a new weighted graph $G' = (V', E', \mu', \kappa')$: for every edge $(u, v) \in E$ such that $f_u < 0 < f_v$, insert a new vertex s such that the minimum energy extension of f assigns $f(s) = 0$. Let $\mu'(s) = 0$.

Abusing notation we will also let $f \in \mathbb{R}^{V'}$ be the minimum energy extension of f to V'. Let $F_0 = \{v \in V' \,|\, f_v = 0\}$, let $F_{\geq 0} = \{v \in V' \,|\, f_v \geq 0\}$ and $F_{\leq 0} = \{v \in V' \,|\, f_v \leq 0\}$. Similarly define $F_{>0}, F_{<0}$ and note that G' has no edges between $F_{>0}$ and $F_{<0}$. For $A, B \subseteq V$, let $A <_f B$ if $f_a < f_b$ for all $a \in A$, $b \in B$.

We have the following lemma regarding the optimal "pinch."

▶ **Lemma 32.** *Let G be a nondegenerate weighted graph. Let $f \in \mathbb{R}^V$ take both positive and negative values. Then $(G', F_{\geq 0})$ and $(G', F_{\leq 0})$ are both nondegenerate weighted graphs with boundary and*

$$\lambda_2(G) = \min_f \left\{ \max\left(\lambda(G', F_{\geq 0}), \lambda(G', F_{\leq 0})\right) \,\middle|\, \pm f \notin \mathbb{R}^V_{\geq 0} \right\}.$$

Proof. Let $\mathcal{R}$ denote the quantity on the right hand side.

We begin by showing that $\lambda_2(G) \leq \mathcal{R}$. Let $f \in \mathbb{R}^V$ take both positive and negative values. Note that $\lambda_2(G) = \lambda_2(G')$. It is easy to see that $(G', F_{\geq 0})$ and $(G', F_{\leq 0})$ are both nondegenerate weighted graphs with boundaries. Let $y, z \in \mathbb{R}^{V'}$ be Dirichlet eigenvectors with Dirichlet eigenvalues $\lambda(G', F_{\geq 0})$ and $\lambda(G', F_{\leq 0})$ respectively. Note that $\operatorname{supp}(L'y) \subseteq F_{\leq 0}$ and that $z\restriction_{F_{\leq 0}} = 0$, thus $z^\top L' y = 0$. Noting that there exists some nonzero $x \in \operatorname{span}(y, z)$ such that $x^\top M' 1 = 0$,

$$\begin{aligned}
\lambda_2(G) &= \lambda_2(G') \\
&= \min_{x \in \mathbb{R}^{V'}} \left\{ \frac{x^\top L' x}{x^\top M' x} \,\middle|\, x^\top M' 1 = 0,\, x \neq 0 \right\} \\
&\leq \max_{x \in \operatorname{span}(y,z)} \frac{x^\top L' x}{x^\top M' x} \\
&= \max_{(\alpha,\beta) \neq 0} \frac{\alpha^2 y^\top L' y + \beta^2 z^\top L' z}{\alpha^2 y^\top M' y + \beta^2 z^\top M' z} \\
&= \max\left(\lambda(G', F_{\geq 0}), \lambda(G', F_{\leq 0})\right).
\end{aligned}$$

Next we show that $\mathcal{R} \leq \lambda_2(G)$. We will exhibit a choice of f taking both positive and negative values such that $\lambda(G', F_{\geq 0}), \lambda(G', F_{\leq 0}) \leq \lambda_2(G)$. This will additionally imply that the minimum is achieved.

Let x be a Neumann eigenvector of G. As $x \neq 0$ and $x^\top M 1 = 0$, it is clear that x takes both positive and negative values. We will pick $f = x$. Abusing notation, also let $x \in \mathbb{R}^{V'}$ be the minimum energy extension of x to V'. Note that $x\restriction_{F_0} = 0$. Let $y = \min(x, 0)$ and $z = \max(x, 0)$ where min and max are taken element wise. Note that $L'y$ agrees with $L'x = \lambda_2(G) M' x$ on the support of y and that y agrees with x on the support of y. Thus

$y^\top L'y = \lambda_2(G) y^\top M'x = \lambda_2(G) y^\top M'y$. Then,

$$\begin{aligned}\lambda(G', F_{\geq 0}) &\leq \frac{y^\top L'y}{y^\top M'y} \\ &= \lambda_2(G).\end{aligned}$$

Similarly, $\lambda(G', F_{\leq 0}) \leq \lambda_2(G)$. ◀

We will need the two following technical lemmas regarding summing resistances.

▶ **Lemma 33.** *Let G be a nondegenerate weighted graph. Let $A, B \subseteq V$ be disjoint nonempty subsets. Let $f \in \mathbb{R}^{V'}$ such that $A \subseteq F_{<0}$ and $B \subseteq F_{>0}$. Then*

$$R'(A, F_{\geq 0}) + R'(F_{\leq 0}, B) \leq R'(A, B).$$

Proof. Let

$$\kappa_A = \kappa'(A, F_{\geq 0}) = \min_y \left\{ y^\top L'y \,\middle|\, y\restriction_A = 1,\, y\restriction_{F_{\geq 0}} = 0 \right\}$$

and let y be the minimizer. Similarly define κ_B and let z be its minimizer. Note that $\operatorname{supp}(L'y) \subseteq F_{\leq 0}$ and $z\restriction_{F_{\leq 0}} = 0$, i.e, $z^\top L'y = 0$.

Let $\alpha = \frac{\kappa_A}{\kappa_A + \kappa_B}$. Note that $(1-\alpha)y - \alpha z$ assigns $1-\alpha$ to vertices in A and $-\alpha$ to vertices in B. Thus

$$\begin{aligned}\frac{1}{R'(A,B)} &\leq ((1-\alpha)y - \alpha z)^\top L'((1-\alpha)y - \alpha z) \\ &= (1-\alpha)^2 \kappa_A + \alpha^2 \kappa_B \\ &= \frac{\kappa_A \kappa_B}{\kappa_A + \kappa_B} \\ &= \frac{1}{R'(A, F_{\geq 0}) + R'(F_{\leq 0}, B)}.\end{aligned}$$

Rearranging terms completes the proof. ◀

▶ **Lemma 34.** *Let G be a nondegenerate weighted graph. Let $A, B \subseteq V$ be disjoint nonempty subsets. For any $\alpha \in (0,1)$, there exists some $f \in \mathbb{R}^V$ with $A \subseteq F_{<0}$ and $B \subseteq F_{>0}$ such that*

$$\kappa'(A, F_{\geq 0}) = \frac{\kappa(A,B)}{\alpha} \qquad \text{and} \qquad \kappa'(B, F_{\leq 0}) = \frac{\kappa(A,B)}{1-\alpha}.$$

Proof. Let

$$\kappa(A,B) = \min_x \left\{ x^\top L x \,\middle|\, x\restriction_A = 0,\, x\restriction_B = 1 \right\}$$

and let x be the minimizer. Define $f = x - \alpha 1$ and take $y = \min(f, 0)$, where the minimum and maximum is element-wise. Note that $L'y$ agrees with Lx on the support of y. By optimality of x, for $v \in A \setminus (A \cup B)$, we have $0 = \frac{\partial}{\partial x_v}(x^\top L x) = 2\,(Lx)_v$. Then,

$$\begin{aligned}\kappa'(A, F_{\geq 0}) &\leq \frac{y^\top L' y}{\alpha^2} \\ &= \frac{\sum_{v \in \operatorname{supp}(y)} y_v (Lx)_v}{\alpha^2} \\ &= \frac{\sum_{v \in A} (-Lx)_v}{\alpha} \\ &= \frac{x^\top L x}{\alpha} \\ &= \frac{\kappa(A,B)}{\alpha}.\end{aligned}$$

Similarly, $\kappa'(B, F_{\leq 0}) \leq \frac{\kappa(A,B)}{(1-\alpha)}$. Then both inequalities must hold with equality by Lemma 33. ◀

We are now ready to prove the following theorem.

▶ **Theorem 35.** *Let G be a nondegenerate weighted graph. Let $\lambda_2(G)$ be the Neumann eigenvalue and let $H_2(G)$ be the two-sided Hardy quantity of G. Then*

$$\frac{1}{4H_2} \leq \lambda_2 \leq \frac{1}{H_2}.$$

Proof. Both the upper and lower bound will follow the same template: we will apply the pinch point characterization, apply Theorem 28 to each Dirichlet problem, and reorder the minima.

The upper bound is,

$$\begin{aligned}\lambda_2(G) &= \min_f \left\{ \max\left(\lambda(G', F_{\geq 0}), \lambda(G', F_{\leq 0})\right) \,\middle|\, \pm f \notin \mathbb{R}^V_{\geq 0} \right\} \\ &\leq \min_f \min_{A,B} \left\{ \max\left(\frac{\kappa'(A, F_{\geq 0})}{\mu(A)}, \frac{\kappa'(B, F_{\leq 0})}{\mu(B)} \right) \,\middle|\, A <_f 0 <_f B,\ A, B \neq \varnothing \right\} \\ &= \min_{A,B} \min_f \left\{ \max\left(\frac{\kappa'(A, F_{\geq 0})}{\mu(A)}, \frac{\kappa'(B, F_{\leq 0})}{\mu(B)} \right) \,\middle|\, A, B \neq \varnothing,\ A <_f 0 <_f B \right\}.\end{aligned}$$

The lower bound is,

$$\begin{aligned}\lambda_2(G) &= \min_f \left\{ \max\left(\lambda(G', F_{\geq 0}), \lambda(G', F_{\leq 0})\right) \,\middle|\, \pm f \notin \mathbb{R}^V_{\geq 0} \right\} \\ &\geq \frac{1}{4} \min_f \min_{A,B} \left\{ \max\left(\frac{\kappa'(A, F_{\geq 0})}{\mu(A)}, \frac{\kappa'(B, F_{\leq 0})}{\mu(B)} \right) \,\middle|\, A <_f 0 <_f B,\ A, B \neq \varnothing \right\} \\ &= \frac{1}{4} \min_{A,B} \min_f \left\{ \max\left(\frac{\kappa'(A, F_{\geq 0})}{\mu(A)}, \frac{\kappa'(B, F_{\leq 0})}{\mu(B)} \right) \,\middle|\, A, B \neq \varnothing,\ A <_f 0 <_f B \right\}.\end{aligned}$$

It remains to understand the following quantity for disjoint nonempty $A, B \subseteq V$.

$$\min_f \left\{ \max\left(\frac{\kappa'(A, F_0)}{\mu(A)}, \frac{\kappa'(B, F_0)}{\mu(B)} \right) \,\middle|\, \pm f \notin \mathbb{R}^V_{\geq 0},\ A <_f 0 <_f B,\ A, B \neq \varnothing \right\} \tag{1}$$

Let $\alpha = \frac{\kappa'(A,B)}{\kappa(A,F_{\geq 0})}$. Then by lemma 33, for all f, we have $\kappa'(B, F_{\leq 0}) \geq \kappa(A,B)/(1-\alpha)$. On the other hand, by lemma 34, there exists some f for which we get equality. Thus,

$$\begin{aligned}(1) &= \kappa(A,B) \min_{\alpha \in (0,1)} \max\left(\frac{1}{\mu(A)\alpha}, \frac{1}{\mu(B)(1-\alpha)} \right) \\ &= \frac{\kappa(A,B)}{(\mu(A)^{-1} + \mu(B)^{-1})^{-1}}.\end{aligned}$$

Taking the minimum over A, B completes the proof. ◀

The following theorem follows as a corollary.

▶ **Theorem 36.** *Let G be a nondegenerate weighted graph. Let $\lambda_2(G)$ be the Neumann eigenvalue and let $\Psi_2(G)$ be the Neumann content of G. Then,*

$$\frac{\Psi_2}{4} \leq \lambda_2 \leq \Psi_2.$$

7 Conclusion and future work

In this paper we introduced the Dirichlet and Neumann contents for nondegenerate weighted graphs (with and without boundary) and showed that these quantities can be related to the Dirichlet and Neumann eigenvalues (Theorems 28 and 35). We believe that these quantities are natural as evidenced by the simplicity of the corresponding proofs. An open question is whether it is possible to develop approximation algorithms based on these new inequalities as opposed to Cheeger's inequality. Such algorithms would be able to exploit the tighter bounds provided by our theorems under a more general setting of weights. We are hopeful that this open question will be answered affirmatively.

References

1. Noga Alon and Vitali D Milman. λ_1, isoperimetric inequalities for graphs, and superconcentrators. *Journal of Combinatorial Theory, Series B*, 38(1):73–88, 1985.
2. Ivo Babuška and John E Osborn. Finite element-Galerkin approximation of the eigenvalues and eigenvectors of selfadjoint problems. *Mathematics of computation*, 52(186):275–297, 1989.
3. Thomas Bühler and Matthias Hein. Spectral clustering based on the graph p-Laplacian. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 81–88. ACM, 2009.
4. Jeff Cheeger. A lower bound for the smallest eigenvalue of the Laplacian. In *Proceedings of the Princeton conference in honor of Professor S. Bochner*, 1969.
5. Jozef Dodziuk. Difference equations, isoperimetric inequality and transience of certain random walks. *Transactions of the American Mathematical Society*, 284(2):787–794, 1984.
6. Shayan Oveis Gharan. Lecture 12: Introduction to Spectral Graph Theory, Cheeger's inequality. Lecture Notes, May 2016.
7. Stephen Guattery, F. Thomson Leighton, and Gary L. Miller. The Path Resistance Method For Bounding The Smallest Nontrivial Eigenvalue Of A Laplacian. *Combinatorics, Probability & Computing*, 8(5), 1999.
8. Stephen Guattery and Gary L. Miller. Graph Embedding and Laplacian Eigenvalues. *SIAM J. Matrix Anal. Appl.*, 21(3):703–723, 2000.
9. G.H. Hardy, Karreman Mathematics Research Collection, J.E. Littlewood, G. Pólya, D.E. Littlewood, and G. Pólya. *Inequalities*. Cambridge Mathematical Library. Cambridge University Press, 1952. URL: `https://books.google.com/books?id=t1RCSP8YKt8C`.
10. Nabil Kahale. A semidefinite bound for mixing rates of Markov chains. *Random Structures & Algorithms*, 11(4):299–313, 1997.
11. Ioannis Koutis, Gary L. Miller, and Richard Peng. A Nearly-$m \log n$ Time Solver for SDD Linear Systems. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 590–598, Washington, DC, USA, 2011. IEEE Computer Society. Available at `arXiv:1102.4842`. `doi:10.1109/FOCS.2011.85`.
12. Alois Kufner, Lech Maligranda, and Lars-Erik Persson. The prehistory of the Hardy inequality. *The American Mathematical Monthly*, 113(8):715–732, 2006.
13. Gyu Eun Lee. Stability of matter. GSO Seminar, UCLA, 2017.

14 László Lovász et al. Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46, 1993.

15 Laurent Miclo. An example of application of discrete Hardy's inequalities. *Markov Process. Related Fields*, 5(3):319–330, 1999.

16 Bojan Mohar. *Some applications of Laplace eigenvalues of graphs*, pages 225–275. Springer Netherlands, Dordrecht, 1997. doi:10.1007/978-94-015-8937-6_6.

17 Benjamin Muckenhoupt. Hardy's inequality with weights. *Studia Mathematica*, 44(1):31–38, 1972.

18 Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2003. Available at: http://www-users.cs.umn.edu/~saad/toc.pdf.

19 Aaron Schild. A Schur Complement Cheeger Inequality. *arXiv preprint*, 2018. arXiv:1811.10834.

20 D. Spielman and S. Teng. Nearly Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885, 2014. Available at arXiv:cs/0607105. doi:10.1137/090771430.

21 Gilbert Strang. *Linear algebra and its applications*. Thomson, Brooks/Cole, Belmont, CA, 2006. URL: http://www.amazon.com/Linear-Algebra-Its-Applications-Edition/dp/0030105676.

22 Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

A Constants in Theorems 28 and 35 are sharp

In this appendix we give constructions of nondegenerate weighted graphs (with and without boundary) that show that the constants in our Theorems 28 and 35 are optimal.

We begin with the Dirichlet case. It is clear that the upper bound is achieved by any nondegenerate weighted graph with boundary (G, S) such that $\left|\overline{S}\right| = 1$. In this case

$$\Psi(G,S) = \frac{\kappa(\overline{S},S)}{\mu(\overline{S})} = \lambda(G,S).$$

We turn to the lower bound. Let $n \in \mathbb{N}$ and let $N = ne^n$. Consider the path graph P with vertices $V = \{v_0, v_1, \ldots, v_N\}$ where v_i has mass

$$\mu_i = \begin{cases} \frac{1}{i(i+1)} & 1 \le i \le N-1 \\ \frac{1}{N} & i = N \end{cases}$$

and the usual path edges. Let $\kappa = 1$ for every edge and let v_0 be the boundary set. We compute the Dirichlet content of (P, v_0).

$$\begin{aligned} \Psi(P,v_0) &= \min_{1\le k\le N} \frac{\kappa(v_0,v_k)}{\sum_{i=k}^N \mu_i} \\ &= \min_{1\le k\le N} \frac{1/k}{1/k} \\ &= 1. \end{aligned}$$

We next show that $\lambda(P, v_0) \le \frac{1}{4} + o(1)$. Consider the assignment

$$x_i = \begin{cases} 0 & 0 \le i \le n-1, \\ \sqrt{i} - \sqrt{n-1} & n-1 \le i. \end{cases}$$

Then

$$\lambda(P, v_0) \leq \frac{x^\top L x}{x^\top M x}.$$

We can bound the numerator above by

$$\begin{aligned} x^\top L x &= \sum_{i=1}^{N} (x_i - x_{i-1})^2 \\ &= \sum_{i=n}^{N} (\sqrt{i} - \sqrt{i-1})^2 \\ &\leq \sum_{i=n}^{N} \left(\frac{1}{2\sqrt{i-1}} \right)^2 \\ &= \frac{1}{4} \sum_{i=n}^{N} \frac{1}{i-1} \\ &= \frac{1}{4} \left(\sum_{i=n}^{N} \frac{1}{i} \right) + O(1). \end{aligned}$$

We can bound the denominator below by

$$\begin{aligned} x^\top M x &= \sum_{i=1}^{N} \mu_i x_i^2 \\ &= \sum_{i=n}^{N} \mu_i \left(\sqrt{i} - \sqrt{n-1} \right)^2 \\ &= \sum_{i=n}^{N-1} \frac{1}{i(i+1)} \left(i + (n-1) - 2\sqrt{i(n-1)} \right) + \frac{1}{N} \left(\sqrt{N} - \sqrt{n-1} \right)^2 \\ &= \sum_{i=n}^{N-1} \frac{1}{i(i+1)} \left(i + (n-1) - 2\sqrt{i(n-1)} \right) + O(1) \\ &= \left(\sum_{i=n}^{N} \frac{1}{i} \right) + \left(\sum_{i=n}^{N-1} \frac{(n-1)}{i(i+1)} \right) - 2 \left(\sum_{i=n}^{N-1} \frac{\sqrt{n-1}}{\sqrt{i}(i+1)} \right) + O(1) \\ &= \left(\sum_{i=n}^{N} \frac{1}{i} \right) + O(1). \end{aligned}$$

Finally, noting that $\sum_{i=n}^{N} 1/i \geq \int_n^N t^{-1}\, dt = \ln(ne^n/n) = n$ diverges to infinity with n, we have that $\lambda(P, v_0) = \frac{1}{4} + o(1)$. We conclude that the constants in Theorem 28 are optimal.

The same construction and a simple symmetry argument shows that the constants in Theorem 35 are optimal.

Improved 3LIN Hardness via Linear Label Cover



Prahladh Harsha
School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India
http://www.tcs.tifr.res.in/~prahladh/
prahladh@tifr.res.in

Subhash Khot
Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, USA

Euiwoong Lee
Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, USA

Devanathan Thiruvenkatachari
Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, USA

Abstract

We prove that for every constant c and $\varepsilon = (\log n)^{-c}$, there is no polynomial time algorithm that when given an instance of 3-LIN with n variables where an $(1-\varepsilon)$-fraction of the clauses are satisfiable, finds an assignment that satisfies atleast $(\frac{1}{2}+\varepsilon)$-fraction of clauses unless $\mathbf{NP} \subseteq \mathbf{BPP}$. The previous best hardness using a *polynomial time* reduction achieves $\varepsilon = (\log \log n)^{-c}$, which is obtained by the Label Cover hardness of Moshkovitz and Raz [*J. ACM*, 57(5), 2010] followed by the reduction from Label Cover to 3-LIN of Håstad [*J. ACM*, 48(4):798–859, 2001].

Our main idea is to prove a hardness result for Label Cover similar to Moshkovitz and Raz where each projection has a *linear* structure. This linear structure of Label Cover allows us to use Hadamard codes instead of long codes, making the reduction more efficient. For the hardness of Linear Label Cover, we follow the work of Dinur and Harsha [*SIAM J. Comput.*, 42(6):2452–2486, 2013] that simplified the construction of Moshkovitz and Raz, and observe that running their reduction from a hardness of the problem LIN (of unbounded arity) instead of the more standard problem of solving quadratic equations ensures the linearity of the resultant Label Cover.

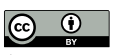
2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases probabilistically checkable proofs, PCP, composition, 3LIN, low soundness error

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.9

Category APPROX

Funding *Prahladh Harsha*: Supported in part by the DIMACS/Simons Collaboration in Cryptography through NSF grant #CNS-1523467 (while the author was visiting Rutgers University and DIMACS) and the Swarnajayanti Fellowship.
Subhash Khot: Supported by the NSF Award CCF-1422159, the Simons Collaboration on Algorithms and Geometry and the Simons Investigator Award.
Euiwoong Lee: Supported in part by the Simons Collaboration on Algorithms and Geometry.
Devanathan Thiruvenkatachari: Supported by same sources as Subhash Khot.


© Prahladh Harsha, Subhash Khot, Euiwoong Lee, and Devanathan Thiruvenkatachari;
licensed under Creative Commons License CC-BY

1 Introduction

In this paper, we study the 3-LIN problem. An instance of 3-LIN consists of a set of n variables over $\mathbb{F}_2$ and a set of m equations that contain at most three variables each, and the goal is to find an assignment to the n variables that satisfies the most number of equations.[1] If the given set of linear equations admits an assignment that satisfies every equation, then one such assignment can be found in polynomial time by Gaussian elimination. However, the general problem of finding the most of number of equations is **NP**-hard when the instance does not admit a satisfying assignment, and a large amount of research has been done on the limit of polynomial time approximation algorithms.

Assigning random values to variables satisfies exactly half the equations in expectation, giving a $1/2$-approximation algorithm. Håstad and Venkatesh [7] achieved an approximation factor of $1/2 + 1/O(\sqrt{m})$, which was improved by Khot and Naor [10] to $1/2 + O(\sqrt{\log n/n})$.

From the hardness side, there are strong hardness results even when the instance is *almost-satisfiable*. For $1 \geq c > s > 0$, let Gap 3-LIN(c, s) denote the problem of distinguishing whether the given instance of 3-LIN is at least c-satisfiable or at most s-satisfiable. Håstad's classic hardness results [6] show the following.

▶ **Theorem 1.1** ([6]). *The following hardness results for* Gap 3-LIN *hold.*

1. *For any constant $\varepsilon > 0$,* Gap 3-LIN$(1 - \varepsilon, 1/2 + \varepsilon)$ *is* **NP***-hard.*
2. *There exists a constant $c > 0$ such that for $\varepsilon = 1/(\log n)^c$, there is no polynomial time algorithm that solves* Gap 3-LIN$(1 - \varepsilon, 1/2 + \varepsilon)$ *unless* $\mathbf{NP} \subseteq \mathbf{DTIME}[n^{O(\log \log n)}]$.

Håstad's results are proved by giving the reduction from Label Cover to 3-LIN. Label Cover is a common starting point for hardness results, and we define the optimization problem below.

▶ **Definition 1.2** (Label Cover). *An instance of* Label Cover *contains a regular bipartite multi- graph $G = (A, B, E)$ and two finite sets Σ_A and Σ_B, where $|\Sigma_A| \geq |\Sigma_B|$. Every vertex in A is supposed to get a label in Σ_A, and every vertex in B is supposed to get a label in Σ_B. For each edge $e \in E$ there is a projection $\pi_e : \Sigma_A \to \Sigma_B$. Given a labeling to the vertices of the graph, i.e., functions $\phi_A : A \to \Sigma_A$ and $\phi_B : B \to \Sigma_B$, an edge $e = (a, b) \in E$ is said to be "satisfied" if $\pi_e(\phi_A(a)) = \phi_B(b)$. For $1 \geq c > s > 1$,* Gap Label Cover(c, s) *is the problem if distinguishing whether the given instance of* Label Cover *is at least c-satisfiable or at most s-satisfiable.*

Håstad's theorem can be stated in terms of reduction from Gap Label Cover$(1, \delta)$ as follows.

▶ **Theorem 1.3** ([6]). *For every $\varepsilon \in (0, 1)$ and positive integer ℓ, there exists a $\delta = \text{poly}(\varepsilon)$ and a $\text{poly}(n, 2^\ell, 2^{1/\varepsilon})$-time reduction from n-sized instances of* Gap Label Cover$(1, \delta)$ *with label size ℓ to* Gap 3-LIN$(1 - \varepsilon, 1/2 + \varepsilon)$.

When [6] was published, the hardness of Label Cover was achieved by the PCP theorem [2, 1] and parallel repetition [13]. More precisely, Gap Label Cover$(1, \varepsilon)$ with label size $\text{poly}(1/\delta)$ was **NP**-hard under $\text{poly}(n^{\log 1/\delta})$-time reductions. The two results of Håstad stated in Theorem 1.1 follow from this hardness of Gap Label Cover and Theorem 1.3 by setting δ to be an arbitrarily small constant and $1/\log n$ respectively. Since

[1] This maximization version is also known as Max 3-LIN in the literature.

achieving a subconstant soundness for Label Cover by parallel repetition requires a superpolynomial blowup in the instance size, $\varepsilon > 0$ could not be taken to subconstant under *polynomial time* reductions. Later in a celebrated paper, Moshkovitz and Raz [12] gave an improved hardness of Label Cover that achieves sub-constant error under polynomial time reductions. Their main result can be stated as follows.

▶ **Theorem 1.4** ([12, Theorem 11]). *For every n, and every $\delta > 0$ (that can be any function of n),* 3-SAT *on inputs of size n can be reduced to* Gap Label Cover$(1, \delta)$ *when* Label Cover *instance has $n^{1+o(1)} \cdot \mathrm{poly}(1/\delta)$ vertices and $|\Sigma_A| \leq \exp(\mathrm{poly}(1/\delta)), |\Sigma_B| \leq \mathrm{poly}(\log 1/\delta)$.*

A corollary of the above result, obtained by combining it with Håstad's reduction from Theorem 1.3, is that given a system of linear equations, it is **NP**-hard to distinguish between cases where $1 - o(1)$ fraction of equations are satisfied vs at most $1/2 + o(1)$ fraction are satisfied, where the $o(1)$ term is $1/(\log \log n)^{-\Omega(1)}$.

▶ **Theorem 1.5** ([12]). *There exists some constant $c > 0$ such that for $\varepsilon = 1/(\log \log n)^c$,* Gap 3-LIN$(1 - \varepsilon, 1/2 + \varepsilon)$ *is* **NP***-hard.*

Later, an improved parallel repetition by Dinur and Steurer [4] allowed c to be an arbitrary constant.

The above route prove hardness of 3-LIN is restricted by the large size of the alphabet in the resulting Label Cover instance in Theorem 1.4. Quantitatively, the alphabet size is exponential in $\mathrm{poly}(1/\varepsilon)$. The fact that the long code in Håstad's reduction has size exponential in the alphabet size restricts $\varepsilon = 1/(\log \log n)^{O(1)}$.

Our main contribution for 3-LIN is to bring ε in the above result down to $1/(\log n)^c$ for any constant c, while keeping the size of the reduced instance polynomial (albeit the reduction becomes randomized).

▶ **Theorem 1.6** (Main). *For any constant $c > 0$ and $\varepsilon = 1/(\log n)^c$, there is no polynomial time algorithm for* Gap 3-LIN$(1 - \varepsilon, 1/2 + \varepsilon)$ *unless* $\mathbf{NP} \subseteq \mathbf{BPP}$.

We get around the above alphabet barrier by starting with a reduction that would make the resulting Label Cover *linear*, and use Hadamard codes instead of long codes. Since the Hadamard code keeps the reduction size polynomial in the alphabet size, we can take $\varepsilon = 1/(\log n)^{\Omega(1)}$. A similar idea was previously used by Khot [8]. We define Linear Label Cover as follows.

▶ **Definition 1.7** (Linear Label Cover). *A* Linear Label Cover *is a special case of* Label Cover *where the alphabets are of the form $\Sigma_A = \mathbb{F}_2^a, \Sigma_B = \mathbb{F}_2^b$ where a, b are natural numbers. Each projection $\pi : \mathbb{F}_2^a \to \mathbb{F}_2^b$ is* affine *in the sense that $\pi(x) = \alpha x + \beta$ for some $\alpha \in \mathbb{F}_2^{b \times a}, \beta \in \mathbb{F}_2^b$. For $1 \geq c > s > 0$, the* Gap Linear Label Cover(c, s) *is defined similarly to* Gap Label Cover(c, s).

We prove the following hardness result for Linear Label Cover, which may be of independent interest.

▶ **Theorem 1.8** (Hardness of Linear Label Cover). *For any constant $c > 0$, for $\delta = 1/(\log n)^c$, there is no polynomial time algorithm for* Gap Linear Label Cover$(1 - \delta, \delta)$ *unless* $\mathbf{NP} \subseteq \mathbf{BPP}$, *when* Label Cover *instance has* $\mathrm{poly}(n)$ *vertices and $|\Sigma_A| = \mathrm{poly}(n), |\Sigma_B| = \mathrm{polylog}(n)$.*

We remark that if the above theorem can be further strengthened to obtain $\delta = 1/n^c$ (i.e., a linear version of the Sliding Scale conjecture), then this leads to near-optimal hardness of 3-LIN (i.e, Gap 3-LIN$(1 - \varepsilon, 1/2 + \varepsilon)$ is hard for $\varepsilon = 1/\mathrm{poly}(n)$) [11].

1.1 Proof Ideas

Our main technical contribution is Theorem 1.8 for Linear Label Cover, essentially proving a linear analogue of the Moshkovitz-Raz PCP [12] followed by the Dinur-Steurer parallel repetition [4]. The proof is given through a long sequence of reductions. We split them in 3 major steps.

1. Interestingly, the starting point of our reduction is again the hardness of (not necessarily linear) Label Cover proved by Moshkovitz and Raz [12] augmented by Dinur and Steurer [4], proving **NP**-hardness of Gap Label Cover$(1, 1/\log^c n)$ for any $c > 0$, while keeping the reduction size and the alphabet size polynomial. In Section 2, we give a *randomized* reduction from this Label Cover to Gap LIN$(1 - 1/\log^c n, 0.9)$. This style of reduction appeared previous from Label Cover to Closest Vector Problem [9]. Note that the standard proof of the PCP theorem encodes 3-SAT (or Circuit SAT) by solving quadratic equations over $\mathbb{F}_2$, and this is essentially the only place that needs where nonlinearity occurs. Our hardness result for solving linear equations with completeness very close to (but not exactly) 1 allows us to follow previous PCP constructions that will ensure linearity of the Label Cover instance in the subsequent steps.
2. To prove the hardness of Linear Label Cover given the above hardness of LIN, we closely follow the steps of Dinur and Harsha [3], who gave a simpler and modular proof of [12]. The two basic building blocks in their proof are robust PCPs and decodable PCPs. Robust PCPs are PCPs where in the soundness case, for any proof and most random choices of the verifier, not only are the local views non-accepting, but they are also very far from any accepting string. It is indeed equivalent to Label Cover. Using our previous hardness for LIN as the starting point and following the standard robust PCP construction (e.g., low-degree extension and sum-check protocol), we can prove a polynomial time reduction to Linear Label Cover$(1 - 1/\log^c n, 1/\log^c n)$ for any $c > 1$, but the alphabet size will be always $\exp(\log^{c_0} n)$ for some $c_0 > 1$, which is superpolynomial.
3. The second building block, decodable PCP, is similar to robust PCP with the additional requirement that the prover is given a position i in the original string and supposed to output the value of the ith position if the given proof is a honest encoding of a valid original string. The main idea of Dinur and Harsha [3] is to iteratively compose a robust PCP with a suitable decodable PCP, where the composed PCP is another robust PCP that consists of a decodable PCP for each constraint of the original robust PCP. This iteratively reduces the query complexity and the alphabet size of the robust PCP, which is related to the alphabet size of the equivalent Label Cover instance. This iterative composition is interleaved and preprocessed by technical operations that reduce the alphabet size of the robust PCP and make it regular.
   Once these two building blocks are linear, the operations of [3] can be used verbatim in our construction. Our main observation is that every step of this construction preserves (1) the robust completeness $1-\delta$ for some $\delta = 1/\mathrm{polylog}(n)$, and (2) the linearity, which were not issues in [3]. In Section 3, we introduce the basic building blocks and these operations, and show how they preserve robust completeness and linearity. These iterative operations will eventually reduce the alphabet size of the Linear Label Cover polynomial, proving Theorem 1.8.

After the hardness of Linear Label Cover is proved, we give a reduction from Linear Label Cover with the above parameters to 3-LIN with the required parameters. We do this by composing with the Hadamard Code to get a $(1 - \varepsilon)$ vs $(1/2 + \varepsilon)$ **NP**-hardness result for 3LIN. Similar PCP constructions based on Hadamard codes were presented in [8]. Details of this step can be found in Section 4.

2 Reduction to System of Linear Equations

In this section, we first prove the hardness of approximate solving linear equations over large fields, where each equation can involve as many variables as possible. It will serve as the starting point towards proving hardness of Linear Label Cover.

▶ **Theorem 2.1.** *For any constant $c > 0$, $\varepsilon = 1/(\log n)^c$, Gap LIN$(1 - 1/(\log n)^c, 0.9)$ is* **NP***-hard under polynomial time randomized reductions.*

Proof. The proof starts from the following hardness of Label Cover, which is obtained by combining the main result of Moshkovitz and Raz [12] with the parallel repetition of Dinur and Steurer [4].

▶ **Theorem 2.2** ([12, 4])**.** *For any constant $c > 0$, for $\delta = 1/(\log n)^c$, Gap Label Cover$(1, \delta)$ is* **NP***-hard when the* Label Cover *instance satisfies $|\Sigma_A|, |\Sigma_B| \leq |A| + |B|$.*

Let $G = (A, B, E)$, Σ_A, Σ_B, and $\{\pi_e\}_{e \in E}$ be an instance of Label Cover. We show a reduction to LIN over $\mathbb{F}_2$ where

- If all Label Cover edges are satisfiable, at least $(1 - \frac{1}{|\Sigma_A|})$ fraction of equations are satisfiable.
- If at most δ fraction of Label Cover edges are satisfiable, at most $(1 - \frac{1}{(\delta|\Sigma_A|)})$ fraction of equations are satisfiable.

For each vertex $v \in \Sigma_A \cup \Sigma_B$ and possible label ℓ on the Label Cover instance, we have a variable $x_{v,\ell}$ in the LIN instance. Let $n = |A||\Sigma_A| + |B||\Sigma_B| = \text{poly}(|A| + |B|)$ be the number of variables. Consider the following four kinds of equations. Recall that every arithmetic is performed over $\mathbb{F}_2$.

$$(1) \quad \sum_{\ell \in \Sigma_A} x_{v,\ell} = 1 \qquad \forall v \in A$$

$$(2) \quad \sum_{\ell \in \Sigma_B} x_{v,\ell} = 1 \qquad \forall v \in B$$

$$(3) \quad \sum_{r : \pi_{uv}(r) = \ell} x_{v,r} = x_{u,\ell} \qquad \forall (u,v) \in E, \forall \ell \in \Sigma_B$$

$$(4) \quad x_{v,\ell} = 0 \qquad \forall (v,\ell) \in A \times \Sigma_A$$

In our final LIN instance, we treat (1), (2), and (3) as *hard constraints* that need to be always satisfied, and find x that always satisfies all hard constraints and as many constraints in (4) as possible. Also note that in (4), we only consider vertices in A.

This is equivalent to the usual LIN problem with hard constraints by *folding*. Formally, let V be the set of assignments that satisfy (1), (2), and (3). If V is empty, we can conclude that the Label Cover instance is unsatisfiable. Otherwise, there exist $c \in \mathbb{N}$ and linearly independent vectors $y_0, \ldots, y_c \in \mathbb{F}_2^{(A \times \Sigma_A) \cup (B \times \Sigma_B)}$ such that $V = \{y_0 + \sum_{i=1}^{c} y_i z_i : z_1, \ldots, z_c \in \mathbb{F}_2\}$. This gives an one-to-one correspondence between $\mathbb{F}_2^c$ and V, so we can treat $z_1, \ldots, z_c$ as the variables of LIN and write the fourth constraints $x_{v,\ell} = 0$ in terms of z, which gives an instance of LIN without hard constraints.

Completeness

If the Label Cover instance is satisfiable, $x_{v,\ell} = 1$ if and only if v is assigned with ℓ gives an assignment that satisfies (1), (2), and (3), and violates one equation in (4) for each $v \in A$.

Soundness

Let x be an assignment that satisfies (1), (2), and (3). For $v \in A \cup B$, let $L_v := \{\ell : x_{v,\ell} = 1\}$. Since (1) and (2) require $\sum_\ell x_{v,\ell} = 1$ for every $v \in A \cup B$, L_v is not empty for every v.

Consider the randomized strategy for Label Cover where each $v \in A \cup B$ is assigned with a uniform random label from L_v independently. For $(u, v) \in E$ with $u \in A, v \in B$, by (3), $x_{v,\ell} = 1$ for some $\ell \in \Sigma_B$ implies that there exists $r \in \Sigma_A$ with $\pi_{uv}(r) = \ell$ such that $x_{u,r} = 1$. This implies (u, v) is satisfied with probability at least $\frac{1}{|L_u|}$ by the randomized strategy. Then the expected fraction of the Label Cover constraints satisfied by the strategy is at least

$$\mathbf{E}_{u\in A}\left[\frac{1}{|L_u|}\right] \geqslant \frac{1}{\mathbf{E}_{u\in A}[|L_u|]}.$$

Therefore, if at most δ fraction of Label Cover constraints are simultaneously satisfiable, we can conclude that

$$\delta \geqslant \frac{1}{\mathbf{E}_{u\in A}[|L_u|]} \quad \Leftrightarrow \quad \mathbf{E}_{u\in A}[|L_u|] \geqslant \frac{1}{\delta}.$$

So in total, at least $\frac{1}{(\delta|\Sigma_A|)}$ fraction of equations are violated.

Gap Amplification

We have a hardness of LIN over $\mathbb{F}_2$ where the completeness value is at least $1 - \frac{1}{|\Sigma_A|}$ and the soundness value is at most $1 - \frac{1}{(\delta|\Sigma_A|)}$. Consider a new system of linear equations where we sample m linear equations independently, where each new equation randomly chooses $\delta \cdot |\Sigma_A|$ old equations and takes a random linear combination of them. In the completeness case, at least an $(1 - O(\delta))$ fraction of new equations can be satisfied by a good assignment to old equations.

In the soundness case, fix an assignment to n possible variables. (There are 2^n of them.) It satisfies at most an $1 - \frac{1}{(\delta|\Sigma_A|)}$ fraction of old equations. Note that if a new equation chooses an old equation not satisfied by the assignment, it is satisfied with probability exactly $1/2$. Therefore, the expected number of new equations satisfied by this fixed assignment is at most

$$m \cdot \left(\left(1 - \frac{1}{(\delta|\Sigma_A|)}\right)^{\delta\cdot|\Sigma_A|} + \frac{1}{2}\right) \leq m \cdot \left(\frac{1}{e} + \frac{1}{2}\right) \leq 0.87m.$$

For a given $c \in \mathbb{N}$, let $\delta = 1/\log^c n$. By taking sufficiently large $m = O(n)$, we can apply the Chernoff and union bound to conclude that no assignment satisfies more than a 0.9 fraction of new equations. So we reduce from Label Cover to Gap LIN$(1 - O(\delta), 0.9)$, which finishes the proof. ◀

We remark that the sampling performed above is the only step in our reduction involving randomization.

3 Reduction to Linear Label Cover

In this section, we show for any $c > 0$, unless **NP** $\subseteq$ **BPP**, there is no polynomial time algorithm for Gap Linear Label Cover$(1 - \varepsilon, \varepsilon)$ with $\varepsilon = 1/(\log n)^c$, proving Theorem 1.8.

The construction we employ is almost identical to that of Dinur and Harsha [3], except that the basic building blocks (robust PCP and decodable PCP) try to prove (almost) satisfiability of linear equations instead of standard quadratic equations. They are introduced in Sections 3.1 and 3.2.

After constructing the building blocks, the result of [3] is proved by iterative composition of them followed by technical steps including alphabet and degree reduction. Our main observation in this part is that each of the steps in the construction preserves *linearity* so that the final Label Cover instance produced also has a liear structure. We present them in Section 3.3 and Section 3.4. Finally, Section 3.5 shows how to combine all these steps to prove Theorem 1.8.

3.1 Robust PCPs

In this subsection, we define robust PCPs. For two strings x, y of the same length n, let $\mathsf{agr}(x, y)$ denote the relative agreement of the strings x, y, defined as

$$\mathsf{agr}(x, y) := \Pr_{i \in [n]}[x_i = y_i]$$

If S is a set of strings, $\mathsf{agr}(x, S)$ is defined as $\max_{y \in S}\{\mathsf{agr}(x, y)\}$.

▶ **Definition 3.1** (Robust PCPs). *For functions $\mathsf{r}, \mathsf{q}, \mathsf{m}, \mathsf{a}, \mathsf{s} : \mathbb{N} \to \mathbb{N}$ and $c, \delta : \mathbb{N} \to [0, 1]$, a verifier V is a* robust probabilistically checkable proof (robust PCP) system *for a promise problem $L = (L_{\text{YES}}, L_{\text{NO}})$ with randomness complexity r, query complexity q, proof length m, alphabet size a, robust completeness c, and robust soundness error δ if V is a probabilistic polynomial-time algorithm that behaves as follows: On input x of length n and oracle access to a proof string $\pi \in \Sigma^{\mathsf{m}(n)}$ over the (proof) alphabet Σ where $|\Sigma| = \mathsf{a}(n)$, V reads the input x, tosses at most $\mathsf{r} = \mathsf{r}(n)$ random coins, and generates a sequence of locations $I = (i_1, \ldots, i_q) \in [\mathsf{m}]^{\mathsf{q}(n)}$ and a predicate $f : \Sigma^{\mathsf{q}} \to \{0, 1\}$, which satisfy the following properties.*

Robust Completeness. *If $x \in L_{\text{YES}}$ then there exists π such that*

$$\mathop{\boldsymbol{E}}_{(I,f)}[\mathsf{agr}(\pi_I, f^{-1}(1))] \geqslant c. \tag{1}$$

Robust Soundness. *If $x \in L_{\text{NO}}$ then for every π,*

$$\mathop{\boldsymbol{E}}_{(I,f)}[\mathsf{agr}(\pi_I, f^{-1}(1))] \leqslant \delta, \tag{2}$$

where the distribution over (I, f) is determined by x and the random coins of V.

We say that V is linear *if $\Sigma = \mathbb{F}_2^b$ for some b and for every f, the accepting sets of the predicate f, i.e., $f^{-1}(1)$, forms an affine subspace of $\Sigma^{\mathsf{q}} = \mathbb{F}_2^{b\mathsf{q}}$ over the field $\mathbb{F}_2$.*

Robust completeness and soundness must be contrasted with (regular) completeness and soundness of standard PCP verifiers in which the expression for completeness and soundness given in (1) and (2) respectively are replaced as follows:

$$\text{Completeness: } \Pr_{I,f}[f(\pi_I) = 1] \geqslant c,$$

$$\text{Soundness: } \Pr_{I,f}[f(\pi_I) = 1] \leqslant \delta.$$

In fact, this is the only difference between the above definition and the standard definition of a PCP system. The robust soundness states that not only does the local view violate the local predicate f, but in fact has very little agreement with any of the satisfying assignments of f (and thus is a strengthening of standard robustness). Robust completeness on the other hand is a weakening of standard completeness.

Another crucial aspect of robust PCP is its equivalence to Label Cover. Namely, existence of robust PCP for L with parameters $\mathsf{r}, \mathsf{q}, \mathsf{m}, \mathsf{a}, \mathsf{s}, c, \delta$ is equivalent to existence of a reduction from L to Gap Label Cover(c, δ) where $|A| = 2^{\mathsf{r}}, |B| = \mathsf{m}, |\Sigma_A| \leq \mathsf{a}^{\mathsf{q}}, |\Sigma_B| = \mathsf{a}$ and each $v \in A$ has degree q. See Lemma 2.5 of [3]. Also note that the definition of linearity is equivalent in robust PCP and Label Cover.

▶ **Theorem 3.2** (Robust PCP, Analog of [3, Theorem 6.4]). *There exist constants $b_1, b_2 > 0$, $c_0 > 1$ such that for any $c > c_0$ and $\varepsilon = 1/\log^c n$, Gap LIN$(1-\varepsilon, 0.9)$ with n variables has a linear robust verifier with robust completeness $1-\varepsilon$, robust soundness error ε, query complexity $1/\varepsilon^{b_1}$, proof length $\mathrm{poly}(n)$, randomness complexity $O(\log n)$, and proof alphabet size at most $1/\varepsilon^{b_2}$.*

Equivalently, there is a (deterministic) polynomial time reduction from Gap LIN$(1-\varepsilon, 0.9)$ to Gap Linear Label Cover$(1-\varepsilon, \varepsilon)$, where the Label Cover instance has $\mathrm{poly}(n)$ veritces, $|\Sigma_A| \leq \exp(1/\varepsilon^{b_1} \log(1/\varepsilon^{b_2}))$, $|\Sigma_B| \leq 1/\varepsilon^{b_2}$, and each $v \in A$ has degree $1/\varepsilon^{b_1}$.

The proof of this theorem is identical to that of [3, Theorem 6.4] and omitted here. The only difference is Gap LIN$(1-\varepsilon, 0.9)$ with $1/\varepsilon = \log^{O(c)} n$ instead of standard quadratic equations when performing the low degree-extension and the sum-check protocol. The theorem follows by observing that all the operations are linear and hence the final predicate is also linear. The completeness of the robust PCP is dictated by the completeness value in Theorem 2.1.

Combining this reduction with the randomized reduction from Theorem 2.1, we obtain the following theorem (which is a more formal version of Theorem 1.8).

▶ **Theorem 3.3** (Hardness of Linear Label Cover). *There exist constants $b_1, b_2 > 0$, $c_0 > 1$ such that for any $c > c_0$ and $\varepsilon = 1/\log^c n$, unless $\mathbf{NP} \subseteq \mathbf{BPP}$, there is no polynomial time algorithm for Gap Linear Label Cover$(1-\varepsilon, \varepsilon)$ where the Label Cover instance has $\mathrm{poly}(n)$ veritces, $|\Sigma_A| \leq \exp(1/\varepsilon^{b_1} \log(1/\varepsilon^{b_2}))$, $|\Sigma_B| \leq 1/\varepsilon^{b_2}$, and each $v \in A$ has degree $1/\varepsilon^{b_1}$.*

3.2 Decodable PCPs

We now discuss the decodable PCP (dPCP), which differs from a PCP in that it has a decoder as opposed to a verifier. A *decoder* is similar to a verifier in that it checks whether a string is in the given language or not by probabilistically checking a small number of positions in the proof, but it is additionally supposed to return the ith position of the original string for given i.

For $\Sigma = \mathbb{F}_2^a$ for some $a \in \mathbb{N}$, let LIN_Σ denote the problem of solving linear equations where an instance consists of k variables that can have a value from Σ, and a system of linear equations C on $k \cdot a$ variables over $\mathbb{F}_2$ canonically represented by the k variables over Σ. It is equivalent to LIN over $\mathbb{F}_2$ on $k \cdot a$ variables, except that we consider each block of a variables as one variable that can take a value from Σ. We define a decoder for LIN_Σ below.

▶ **Definition 3.4** (Decoder for LIN_Σ). *Let $\Sigma = \mathbb{F}_2^a$ and $\sigma = \mathbb{F}_2^b$ for some a and b. A decoder for LIN_Σ over a proof alphabet σ with parameters $\mathsf{m}, \mathsf{q}, \mathsf{r} : \mathbb{N} \to \mathbb{N}$ is a probabilistic polynomial-time algorithm $\mathcal{D}$. It is given a system of linear equations C on n variables over Σ, and an index $j \in [n]$ as input, and oracle access to a proof π of length $\mathsf{m}(n)$ over proof alphabet σ. It tosses $\mathsf{r} = \mathsf{r}(n)$ random coins and generates (1) a sequence of $\mathsf{q} = \mathsf{q}(n)$ locations $I = (i_1, \ldots, i_{\mathsf{q}})$ and (2) a (local decoding) function $f : \sigma^{\mathsf{q}} \to \Sigma \cup \{\perp\}$. $\mathcal{D}$ is called linear if for every f, $P := f^{-1}(\Sigma)$ is an affine space of $\sigma^{\mathsf{q}} = (\mathbb{F}_2^{\mathsf{q}b})$ and $f : P \to \Sigma$ is an affine function over the base field $\mathbb{F}_2$.*

Now we define a dPCP for LIN_Σ. The dPCP in [3] is defined for Circuit SAT, whereas ours is for LIN_Σ. Note that unlike in [3], the dPCP we will construct does not imply any computational hardness, because it only proves whether the given system of linear equations is perfectly satisfiable or not, which is a computationally easy problem. The key point is it proves the system is satisfiable using a proof which is in some sense "locally decodable". The dPCP will then be composed with the previous linear robust PCP, which is a system of linear equations with *imperfect completeness*, to reduce the query complexity.

▶ **Definition 3.5** (Decodable PCPs for LIN_Σ)**.** *For functions $\delta : \mathbb{N} \to [0,1]$ and $\mathsf{L} : \mathbb{N} \to \mathbb{N}$, we say that a PCP decoder $\mathcal{D}$ is a* decodable probabilistically checkable proof (dPCP) system *for LIN_Σ with perfect completeness, soundness δ and list size L if the following completeness and soundness properties hold for every system of linear equations C on n variables over Σ.*

Completeness. *For any $y \in \Sigma^n$ that satisfies every equation in C, there exists a proof $\pi \in \sigma^{\mathsf{m}}$, also called a decodable PCP, such that*

$$\Pr_{j,I,f}[f(\pi_I) = y_j] = 1,$$

where $j \in [n]$ is chosen uniformly at random and I, f are distributed according to C, j, and the verifier's random coins.

Soundness. *For any $\pi \in \sigma^{\mathsf{m}}$, there is a list of $0 \leqslant \ell \leqslant \mathsf{L}$ strings $y^1, \ldots, y^\ell$, where each y^i satisfies all equations in C, such that*

$$\Pr_{j,I,f}[f(\pi_I) \notin \{\perp, y^1_j, \ldots, y^\ell_j\}] \leqslant \delta.$$

Robust soundness. *We say that $\mathcal{D}$ is a* robust dPCP system *for LIN_Σ with robust soundness error δ, if the soundness criterion above can be strengthened to the following robust soundness criterion,*

$$\boldsymbol{E}_{j,I,f}[\mathsf{agr}(\pi_I, \mathrm{BAD}(f))] \leqslant \delta,$$

where

$$\mathrm{BAD}(f) := \{w \in \sigma^{\mathsf{q}} : f(w) \notin \{\perp, y^1_j, \ldots, y^\ell_j\}\}.$$

The dPCP result we use is the following.

▶ **Theorem 3.6** (dPCP, Analog of [3, Theorem 6.5])**.** *There exist constants $\alpha, \gamma > 0$ such that for every $\delta \geqslant n^{-\alpha}$ and input alphabet size Σ of size at most n^γ, LIN_Σ has a linear robust decodable PCP system with perfect completeness, robust soundness error $\delta > 0$ and list size $\mathsf{L} \leqslant 2/\delta$, query complexity $n^{1/8}$, proof alphabet σ of size n^γ, proof length $\mathrm{poly}(n)$, and randomness complexity $O(\log n)$.*

The proof of this theorem is identical to that of [3, Theorem 6.5], except that the initial starting point is LIN_Σ instead of Circuit SAT_Σ. Since the starting point is linear and all transformations are linear, the final object is also linear. The perfect completeness is also maintained. As mentioned before, the dPCP constructed here does not imply any computational hardness unlike in [3].

3.3 Composition

After having building blocks, Dinur and Harsha [3] show how to compose those blocks iteratively to reduce the query complexity and the alphabet size. Each composition involves several other operations including alphabet and degree reductions. While the soundness analyses for them are already proved in [3], we show that all of their operations preserve linearity and robust completeness.

Efficient Composition ([3, Theorem 4.2])

In the composition, given a regular robust linear PCP verifier V and a robust linear PCP decoder $\mathcal{D}$, the composed verifier V' expects a decodable PCP for each constraint of V. Recall that the linearity of V is equivalent to the fact that each constraint of V is a system of linear equations over $\mathbb{F}_2$, which is exactly what $\mathcal{D}$ expects. An informal description of the composed verifier is as follows:

1. Randomly choose a location i of the proof for V. Let $C_1, \ldots, C_D$ be the constraints of V containing the location.
2. Using a $(\varepsilon, \varepsilon^2)$-sampler $([D], [D], E)$ and a random $s \in [D]$, choose a subset $S \subseteq \{1, \ldots, D\}$ and run the inner PCP decoder $\mathcal{D}$ for each C_j with $j \in S$ to decode the ith symbol in the original proof.
3. Accept if all the values returned by the PCP decoders are the same.

For the second step above, we use $(\varepsilon, \varepsilon^2)$-samplers given in [5]. Theorem 4.2 of [3] shows the soundness of the composed verifier V', yielding Table 1 below (Table 4.2 in [3]).

Table 1 Parameters for Composition.

	V	$\mathcal{D}$	V'
proof alphabet	Σ	σ	σ
randomness complexity	R	r	$\log M + \mathsf{r} + \log D$
query complexity	Q	q	$4/\varepsilon^4 \cdot q$
proof degree	D	d	d
proof length	M	m	$2^{\mathsf{R}} \cdot \mathsf{m}$
robust soundness error	Δ	δ	$\Delta\mathsf{L} + 4\mathsf{L}\varepsilon + \delta$
list size	-	L	-

We check this composition preserves robust completeness and linearity.

- Linearity: Linearity (over $\mathbb{F}_2$) is preserved if both V and $\mathcal{D}$ are linear, since the only additional check we perform is to check whether the returned values are equal.
- Robust completeness: Suppose that there exists a proof Π for V that achieves the robust completeness of at least $1 - \xi$. Recall that the composed verifier expects, for each constraint of the outer PCP, a satisfying assignment encoded by the inner dPCP. The proof for the composed verifier is the concatenation of all these encodings. Consider the proof to the composed verifier constructed by the honest encoding of the assignment that achieves the robust completeness for the outer PCP verifier. We will show that this proof achieves robust completeness $1 - \xi$.
  Let i be a proof location in the outer PCP and $C_1, \ldots, C_D$ be the constraints involving i. Furthermore, let ξ_i be the fraction of these constraints violated by the proof. Since Π is at least $(1 - \xi)$-robustly complete, we have $\mathbf{E}_i[\xi] \leq \xi$. For each sample s chosen in the sampler, let $\xi_{i,s}$ be the fraction of constraints in S (chosen by sampler) that are violated. By regularity of sampler, we have $\mathbf{E}_s[\xi_{i,s}] \leq \xi_i$.
  A local view of the composed verifier (corresponding to i, s and the inner dPCP randomness) comprises of the concatenation of the local views of the dPCP encodings corresponding to the constraints in S. Since the the inner dPCP has perfect completeness we have the following. Whenever the constraint is satisfied, the corresponding inner dPCP's encodings satisfies all constraints while we have no guarantee when the constraint is not satisfied. Since for each (i, s), the fraction of violated constraints is $\xi_{i,s}$, we have

that at least $(1-\xi_{i,s})$-fraction of the local inner views corresponding to (i,s) are satisfying and furthermore they all decode to the same $\Pi(i)$. Hence, the local view of the composed verifier corresponding to (i,s) is at least $(1-\xi_{i,s})$-close to a satisfying view. Hence, the robust completeness of this honest proof is at least $\mathbf{E}_{i,s}[1-\xi_{i,s}] \geq 1-\xi$.

3.4 Label Cover Operations

After the composition, the alphabet reduction step is applied to ensure that the alphabet size is polynomial in the query complexity and the inverse of the soundness. Also, since the basic robust PCP given in Theorem 3.2 is not necessarily regular, we also need to show how to make the initial robust PCP regular. This subsection introduces various such operations and explains why they preserve robust completeness and linearity.

Degree Reduction ([3, Theorem 5.1])

Given an instance of Label Cover $G=(A,B,E)$, the degree reduction makes the instance right-regular by appropriately duplicating right vertices and each edge exactly the same number of times. Theorem 5.1 of [3] ensures that by increasing robust soundness by 4μ additively, we can ensure that the right degree is $4/\mu^4$ for all right vertices. We check that this operation preserves linearity and robust completeness.

- Linearity: Linearity is obviously preserved, because there is no change in the constraint.
- Robust completeness: Since each edge is duplicated the same number of times, robust completeness does not decrease.

Alphabet Reduction ([3, Theorem 5.5])

Given an instance of Label Cover $G=(A,B,E)$ where Σ_A and Σ_B are the alphabet set of the left (bigger) side and the right (smaller) side respectively, the alphabet reduction replaces Σ_B by a smaller set σ by finding a suitable linear code $C:\Sigma_B \to \sigma^k$ and replacing each vertex $b \in B$ by k vertices $b_1,\ldots,b_k$. Then assigning $x \in \Sigma_B$ to b corresponds to assigning $(C(x))_i$ to $b_1,\ldots,b_i$. Theorem 5.5 of [3] ensures that if C has a relative distance $1-\eta^3$, this operation increases robust soundness by at most 3η additively. We check that this operation preserves linearity and robust completeness.

- Linearity: Linearity over $\mathbb{F}_2$ is preserved if the code $C:\Sigma_B \to \sigma^k$ is linear with $\sigma = \mathbb{F}_{2^a}$ as the base field for some $a \in \mathbb{N}$. The code used in Remark 5.4 of [3] is already linear.
- Robust completeness: If an edge (a,b) of the original Label Cover instance is preserved and the new instance follows the honest encoding, all k edges of the new instance corresponding to (a,b) will be satisfied. Therefore, robust completeness cannot decrease.

Flip Sides ([3, Section 5.3])

Given an instance of Label Cover $G=(A,B,E)$ where each right vertex $b \in B$ has degree d, the flip side is achieved by flipping A and B, and assigning each $v \in B$ a label from Σ_A^d, which is supposed to denote the assignments to its neighbors in the original instance. If $v \in B$ has $u_1,\ldots,u_d \in A$ as neighbors, (v,u_i) in the new instance is satisfied (i) if the label $(a_1,\ldots,a_d) \in \Sigma_A^d$ for v has $b \in \Sigma_B$ such that the label pair (a_i,b) satisfies the edge (u_i,v) in the old instance, and (ii) if a_i is equal to the label assigned to u_i. This obviously does not change the robust soundness. We check that it also preserves linearity and robust completeness.

Table 2 Sequence of steps to regularize the Label Cover instance. * denotes irregular instances where the number denotes the average degree.

Label Cover (Robust PCPs)	I	Degree Red. ($\to d$)	Flip	Degree Red. ($\to d$)	Alphabet Red. ($\to \sigma$)
# left vertices (randomness)	n	n	mD_B	mD_B	mD_B
# right vertices (proof length)	m	mD_B	n	nD_Ad	nD_Adk
left degree (query complexity)	D_A^*	dD_A^*	d	d^2	d^2k
right degree (proof degree)	D_B^*	d	D_Ad^*	d	d
left alphabet (# accepting conf.)	Σ_A	Σ_A	Σ_A^d	Σ_A^d	Σ_A^d
right alphabet (proof alphabet)	Σ_B	Σ_B	Σ_A	Σ_A	σ
soundness error (rob. soundness error)	δ	$\delta + 4\mu$	$\delta + 4\mu$	$\delta + 8\mu$	$\delta + 8\mu + 3\eta$
rob. completeness (rob. completeness)	$1 - \xi$	$1 - \xi$	$1 - \xi$	$1 - \xi$	$1 - \xi$

- Linearity: Linearity is preserved, because for each $v \in B$, the set of $(a_1, ..., a_d)$ satisfying (i) above is an affine subspace of $(\Sigma_A)^d$, and the new constraint is merely a projection.
- Robust completeness: Cannot decrease since if $v \in B$ was assigned $b \in \Sigma_B$ in the original instance, it can be assigned $(a_1, \ldots, a_d) \in \Sigma_A$ such that (i) $\pi_{(u_i,v)}(a_i) = b$, and (ii) a_i was assigned to u_i if (u_i, v) was satisfied in the original instance.

We use a combination of the above 3 operations to get a regular Label Cover instance, as shown below.

Given an $\varepsilon > 0$, by using $(O(\varepsilon), O(\varepsilon^2))$-samplers in the composition and doing the above operations with $\eta = O(\varepsilon)$, $d = O(1/\varepsilon^4)$, distance $1-O(\varepsilon^3)$, $|\sigma| = O(1/\varepsilon^6), k = O(1/\varepsilon^6)\cdot|\Sigma'| \leq O(1/\varepsilon^6) \cdot q|\Sigma|$, we can deduce the following lemma.

▶ **Lemma 3.7** ([3, Lemma 5.7]). *For all $\varepsilon : \mathbb{N} \to [0, 1]$, suppose L has a robust linear PCP verifier V with randomness complexity r, query complexity q, proof length m, average proof degree D_B, robust completeness c, robust soundness error δ over a proof alphabet Σ. Then L has a regular reduced linear robust PCP verifier, which we shall denote by* $\mathsf{regular}_\varepsilon(V)$ *with*

- *randomness complexity* $\log m + \log D_B$,
- *query complexity* $O(q \log |\Sigma|/\varepsilon^{14})$,

- *proof length* $O(q^2 2^r \log|\Sigma|/\varepsilon^{10})$,
- *proof degree* $O(1/\varepsilon^4)$,
- *proof alphabet* σ *of size at most* $O(1/\varepsilon^6)$,
- *robust completeness* c,
- *and robust soundness* $\delta + \varepsilon$.

3.5 Putting things together

Finally we prove Theorem 1.8 on the hardness of Linear Label Cover. Let $c > 0$ be an arbitrary constant. Let $\mathcal{D}$ be the PCP decoder from Theorem 3.6 and $\mathcal{V}$ be the robust PCP from Theorem 3.2 with robust completeness $1-\delta$ with $\delta = \log^c n$, robust soundness error $\varepsilon = 1/\log^{c_0} n$ for some $c_0 > 1$, query complexity $1/\varepsilon^{O(1)}$, randomness complexity $O(\log n)$ and proof length $\mathrm{poly}(n)$.

▶ **Lemma 3.8** ([3, Lemma 6.6]). *Let $\mathcal{D}$, $\mathcal{V}$, ε, δ be as defined above and set $\varepsilon_i = (\varepsilon)^{1/3^i}$. There exist constants c_0, c_1, $c_3 > 0$ such that for every $i \geqslant 0$ as long as $\varepsilon_i < c_0$, the following holds.* Gap $\mathrm{LIN}(1-\delta, 0.9)$ *has a regular linear robust PCP verifier V_i with query complexity $1/\varepsilon_i^{c_1}$, robust completeness $1-\delta$, robust soundness error $2\varepsilon_i$, proof alphabet Σ_i of size ${c_3/\varepsilon_i}^6$, randomness complexity $O(\log n)$ and proof length* $\mathrm{poly}(n)$.

Proof. The proof is similar to [3], and is a sequence of compositions. We start with the regularized robust verifier given by applying the sequence of steps given in Section 3.4 to the robust PCP verifier given in Theorem 3.2. In each subsequent step, we compose the robust verifier obtained in the previous step with a dPCP, and apply the alphabet reduction (Theorem 5.5 of [3]) to reduce the size of the alphabet to c_3/ε_{i+1}^6. All the parameters remain the same as in [3], and we only need to focus on the two additional properties we need, linearity and robust completeness.

Recall that a PCP with robust completeness $1-\delta$, when composed with a dPCP with perfect completeness, yields a composed PCP with robust completeness $1-\delta$. In each step the inner PCP decoder has perfect completeness, therefore the robust completeness of the composed PCP is preserved. Recall that the alphabet reduction step also doesn't affect the perfect completeness.

Linearity is also preserved because all basic components are linear and all steps (e.g., composition, alphabet reduction, and regularization) preserve linearity as previously discussed. ◀

The above lemma shows that we can iteratively reduce the query complexity until some absolute constant while maintaining the soundness and the alphabet size polynomial in the query complexity.(And the total size of the instance always remains polynomial in n.) Only a constant number of iterations is needed until $(\text{proof alphabet size})^{(\text{query complexity})}$, an upper bound on the size of alphabet in the equivalent Label Cover instance, becomes polynomial in n. This proves our main Theorem 1.8 for Linear Label Cover.

Proof of Theorem 1.8. Set i from Lemma 3.8 so that

$$(\text{proof alphabet size})^{(\text{query complexity})} = (c_3/\varepsilon_i^6)^{1/\varepsilon_i^{c_1}} = \exp\left(\frac{1}{\varepsilon_i^{c_1}} \cdot \log\left(\frac{c_3}{\varepsilon_i^6}\right)\right) \leq \mathrm{poly}(n).$$

This ensures that $\varepsilon_i = 1/\log^{c_4} n$ for some $c_4 > 0$. Using the equivalence between Label Cover and robust PCP, we have a hardness of Label Cover where the number of vertices and the size of label are bounded by $\mathrm{poly}(n)$, and the completeness is at least $1 - 1/\log^c n$, the soundness is $1/\log^{c_4} n$. Applying the parallel repetition of [4] $O(c/c_4)$ times to reduce the soundness to $1/\log^c n$ finishes the proof. ◀

4 Reduction from Linear Label Cover to 3LIN

In this section, we prove our main Theorem 1.6 for 3-LIN. Recall that Theorem 3.3 shows a randomized polynomial reduction from 3-SAT to Gap Linear Label Cover$(1 - \log^c n, \log^c n)$ for any constant $c > 0$, where the number of vertices as well as the number of labels are bounded by a polynomial. Therefore, the following theorem finishes the proof of Theorem 1.6. The main idea is to use Hadamard codes instead of long codes using the fact that the Label Cover instance is linear. A similar argument was used in [8].

▶ **Lemma 4.1.** *There is a polynomial time reduction from* Gap Linear Label Cover$(1 - \delta, s)$ *to* Gap 3-LIN$(1 - \delta, 1/2 + \sqrt{s}/2)$*, where the size of the* 3-LIN *instance is polynomial in the number of vertices and the size of label in the* Label Cover *instance.*

Proof. Let $G = (A, B, E)$, Σ_A, Σ_B, $\{\pi_e\}_{e \in E}$ be an instance of Gap Linear Label Cover $(1 - \delta, s)$. Moreover, since the label cover is linear, let the labels to left hand side vertices come from $\mathbb{F}_2^\ell$ and the right hand side vertices from $\mathbb{F}_2^r$, and the mapping on each edge is an affine mapping. Our reduction is described by the following test.

Test

- Consider an edge (u, v). The labels $x \in \mathbb{F}_2^\ell, y \in \mathbb{F}_2^r$ corresponding to the vertices have to satisfy $x = Ay + b$.
- From the proof, we randomly sample the Hadamard code of x at location α, and that of y at locations β and $\beta + \gamma$, where $\gamma = A^T \cdot \alpha$.
- Check if $\langle \alpha, x \rangle + \langle \beta, y \rangle + \langle \beta + \gamma, y \rangle = \langle \alpha, b \rangle$

Completeness

In the completeness case, if the labels x, y satisfy the edge in the Linear Label Cover, then we can see that the test will pass.

$$\begin{aligned}
&\langle \alpha, x \rangle + \langle \beta, y \rangle + \langle \beta + \gamma, y \rangle \\
=&\langle \alpha, Ay \rangle + \langle \alpha, b \rangle + \langle \beta, y \rangle + \langle \beta + \gamma, y \rangle \\
=&\langle \alpha, Ay \rangle + \langle \alpha, b \rangle + \langle A^T \alpha, y \rangle \\
=&\langle \alpha, b \rangle
\end{aligned}$$

Therefore, if $1 - \delta$ edges are satisfiable in the linear Label Cover, at least $1 - \delta$ fraction of 3LIN constraints are satisfied.

Soundness

Consider the case where at most s fraction of edges can be satisfied for any labeling in the Linear Label Cover. Let the Hadamard code encoding function for the left vertices be L and right vertices be R. Consider their Fourier transforms,

$$L(\alpha) = \sum_x \hat{L}(x)\chi_x(\alpha)$$

$$R(\beta) = \sum_y \hat{R}(y)\chi_y(\beta)$$

Let's fix an edge, and analyze the probability that the test will accept. We switch to a -1,+1 notation for convenience.

$$\begin{aligned}\Pr[\text{Test accepts}] &= \Pr_{\alpha,\beta}[\langle\alpha,x\rangle + \langle\beta,y\rangle + \langle\beta + A^T\alpha, y\rangle + \langle\alpha,b\rangle = 0] \\ &= \Pr_{\alpha,\beta}[(-1)^{\langle\alpha,x\rangle+\langle\beta,y\rangle+\langle\beta+A^T\alpha,y\rangle+\langle\alpha,b\rangle} = 1] \\ &= \frac{1 + \mathbf{E}_{\alpha,\beta}\left[L(\alpha)R(\beta)R(\beta + A^T\alpha)(-1)^{\langle\alpha,b\rangle}\right]}{2}\end{aligned}$$

Consider the expectation on the right hand side of the above equation.

$$\begin{aligned}&\mathbf{E}_{\alpha,\beta}\left[L(\alpha)R(\beta)R(\beta + A^T\alpha)(-1)^{\langle\alpha,b\rangle}\right] \qquad (3)\\ &\leqslant \sum_{x,y} \hat{L}(x)\hat{R}(y)^2 \mathbf{E}_{\alpha,\beta}\left[\chi_x(\alpha)\chi_y(\beta)\chi_z(\beta + A^T\alpha)(-1)^{\langle\alpha,b\rangle}\right] \\ &\leqslant \sum_{x,y,x=Ay+b} \hat{L}(x)\hat{R}(y)^2 \\ &\leqslant \sqrt{\sum_{x,y,,x=Ay+b} \hat{R}(y)^2}\sqrt{\sum_{x,y,x=Ay+b} \hat{L}(x)^2\hat{R}(y)^2}\end{aligned}$$

In the above equation, the first term is bounded by 1, and therefore,

$$(3) \leqslant \sqrt{\sum_{x,y,,x=Ay+b} \hat{L}(x)^2\hat{R}(y)^2}$$

Consider a random assignment where a left vertex gets a label x with probability $\hat{L}(x)^2$ and a right vertex gets a label y with probability $\hat{R}(y)^2$. The probability that such a random assignment would satisfy the edge, and therefore the expected fraction of edges satisfied, is exactly

$$\sum_{x,y,x=Ay+b} \hat{L}(x)^2\hat{R}(y)^2$$

If at most s fraction of edges can be satisfied by any assignment, then

$$s \geqslant \sum_{x,y,x=Ay+b} \hat{L}(x)^2\hat{R}(y)^2 \geqslant (2 \cdot \Pr[\text{Test accepts}] - 1)^2$$

or

$$\Pr[\text{Test accepts}] \leqslant \frac{1}{2} + \frac{\sqrt{s}}{2}$$

Therefore, the expected fraction of 3LIN constraints satisfied is at most $\frac{1}{2} + \frac{\sqrt{s}}{2}$. ◀

References

1. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *J. ACM*, 45(3):501–555, May 1998. (Preliminary version in *33rd FOCS*, 1992). doi:10.1145/278298.278306.
2. Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *J. ACM*, 45(1):70–122, January 1998. (Preliminary version in *33rd FOCS*, 1992). doi:10.1145/273865.273901.

3 Irit Dinur and Prahladh Harsha. Composition of low-error 2-query PCPs using decodable PCPs. *SIAM J. Comput.*, 42(6):2452–2486, 2013. (Preliminary version in *51st FOCS*, 2009). doi:10.1137/100788161.

4 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *Proc. 46th ACM Symp. on Theory of Computing (STOC)*, pages 624–633, 2014. doi:10.1145/2591796.2591884.

5 Oded Goldreich. A Sample of Samplers: A Computational Perspective on Sampling. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, volume 6650 of *LNCS*, pages 302–332. Springer, 2011. doi:10.1007/978-3-642-22670-0_24.

6 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, July 2001. (Preliminary version in *29th STOC*, 1997). doi:10.1145/502090.502098.

7 Johan Håstad and Srinivasan Venkatesh. On the advantage over a random assignment. *Random Structures Algorithms*, 25(2):117–149, 2004. (Preliminary version in *34th STOC*, 2002). doi:10.1002/rsa.20031.

8 Subhash Khot. Improved Inaproximability Results for MaxClique, Chromatic Number and Approximate Graph Coloring. In *Proc. 42nd IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 600–609, 2001. doi:10.1109/SFCS.2001.959936.

9 Subhash Khot. Inapproximability Results for Computational Problems on Lattices. In Phong Q. Nguyen and Brigitte Vallée, editors, *The LLL Algorithm - Survey and Applications*, Information Security and Cryptography, pages 453–473. Springer, 2010. doi:10.1007/978-3-642-02295-1_14.

10 Subhash Khot and Assaf Naor. Linear Equations Modulo 2 and the L_1 Diameter of Convex Bodies. *SIAM J. Comput.*, 38(4):1448–1463, 2008. (Preliminary version in *48th FOCS*, 2007). doi:10.1137/070691140.

11 Dana Moshkovitz. The Projection Games Conjecture and the NP-Hardness of *lnn*-Approximating Set-Cover. *Theory Comput.*, 11:221–235, 2015. (Preliminary version in *15th APPROX*, 2012). doi:10.4086/toc.2015.v011a007.

12 Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5), 2010. (Preliminary version in *49th FOCS*, 2008). doi:10.1145/1754399.1754402.

13 Ran Raz. A Parallel Repetition Theorem. *SIAM J. Comput.*, 27(3):763–803, June 1998. (Preliminary version in *27th STOC*, 1995). doi:10.1137/S0097539795280895.

Dynamic Pricing of Servers on Trees

Ilan Reuven Cohen
TU Eindhoven, The Netherlands
CWI, Amsterdam, The Netherlands
ilanrcohen@gmail.com

Alon Eden
Tel Aviv University, Israel
alonarden@gmail.com

Amos Fiat
Tel Aviv University, Israel
fiat@tau.ac.il

Łukasz Jeż
University of Wrocław, Poland
lje@cs.uni.wroc.pl

Abstract

In this paper we consider the k-server problem where events are generated by selfish agents, known as *the selfish k-server problem*. In this setting, there is a set of k servers located in some metric space. Selfish agents arrive in an online fashion, each has a request located on some point in the metric space, and seeks to serve his request with the server of minimum distance to the request. If agents choose to serve their request with the nearest server, this mimics the greedy algorithm which has an unbounded competitive ratio. We propose an algorithm that associates a surcharge with each server independently of the agent to arrive (and therefore, yields a truthful online mechanism). An agent chooses to serve his request with the server that minimizes the distance to the request *plus* the associated surcharge to the server.

This paper extends [9], which gave an optimal k-competitive dynamic pricing scheme for the selfish k-server problem on the line. We give a k-competitive dynamic pricing algorithm for the selfish k-server problem on tree metric spaces, which matches the optimal online (non truthful) algorithm. We show that an α-competitive dynamic pricing scheme exists on the tree if and only if there exists α-competitive online algorithm on the tree that is *lazy* and *monotone*. Given this characterization, the main technical difficulty is coming up with such an online algorithm.

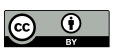
2012 ACM Subject Classification Theory of computation → Online algorithms; Theory of computation → Algorithmic mechanism design

Keywords and phrases Online algorithms, Online mechanisms, k-server problem, Online pricing

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.10

Category APPROX

Funding *Ilan Reuven Cohen*: Partially supported by the ERC consolidator grant 617951.
Alon Eden: Partially supported by the European Research Council under the European Unions Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement number 337122, by the Israel Science Foundation (grant numbers 317/17 and 1841/14).
Amos Fiat: Partially supported by the Israel Science Foundation (grant number 1841/14).
Łukasz Jeż: Partially supported by Polish National Science Centre grant 2016/22/E/ST6/00499.



© Ilan Reuven Cohen, Alon Eden, Amos Fiat, and Łukasz Jeż;
licensed under Creative Commons License CC-BY
Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh; Article No. 10; pp. 10:1–10:22

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Online algorithms were designed to deal with cases where the input arrives piecemeal over time and consists of a sequence of events. Problems such as paging, online matching, online scheduling, etc., are all examples of such problems.

This paper, belongs to a thread of recent research where events are selfish and the goal is to set surcharges on the various decisions that can be made by the agent with some desirable goal in mind such as minimizing social cost, makespan, completion time, flow time, sum of completion times, etc. (See Section 1.1 for some examples.) The prices may change over time, but must be known to the selfish agent upon arrival so that the agent can make an informed decision. Truthfulness is immediate in such settings, the agent gets asked no questions and therefore cannot lie about anything. The agent simply takes the utility maximizing (disutility minimizing) option available.

Specifically, in the dynamic pricing scheme for the k-server problem that we consider, the mechanism sets a surcharge on each server *prior* to an arrival of the next request. The agent that issues the request greedily chooses the server which minimizes the distance between the server and request *plus* the surcharge for the server. Note that the mechanism may update the surcharge of the servers based on *past* requests.

This paper extends the dynamic pricing results obtained for the k-server problem in [9] and deals with servers on a tree rather than restricted to a line. Although the basic idea is the same: use dynamic pricing to "nudge" selfish agents to act as though they were under the control of a centralized online algorithm, the tree metric is much more challenging to deal with than the line.

We show that any α-competitive online algorithm on the tree that is simultaneously *(i) lazy*: moves at most one server and *(ii) monotone*: the set of points served by server (if non-empty) is contiguous and includes the server location, can be converted into a dynamic posted pricing scheme for the selfish k-server problem on the tree with a competitive ratio of α. These properties were defined and in fact proved for the line [9], but they extend naturally to trees; cf. Section 2.2 for formal definitions. Thus, the main challenge in this paper is to give a k-competitive k-server algorithm for the tree that is lazy and monotone.

In the work of Cohen et al. [9], the main idea for obtaining an algorithm with those properties on a line is to run a simulation of the Double Cover (DC) algorithm and serve each request (at point) r with a server that is adjacent to r (i.e., there are no intermediate servers on its path to r) and that can be matched to a simulated Double Cover server which serves r in a min cost matching. This maintains the competitive ratio and ensures laziness and monotonicity. Generalizing this idea to trees is not immediate. In particular, choosing an arbitrary server adjacent to the request which can also be matched to a simulated server in a min cost matching results in non-monotonicity, which cannot be priced. This means that one needs a deeper understanding of the tree topology in deciding which of the servers is to serve the request (We explain this in detail in Section 2.2).

1.1 Related Work

1.1.1 Dynamic Pricing Schemes and Online Mechanisms

Lavi and Nisan [18] initiated the study of competitive analysis of incentive compatible online auctions. In particular, they give an incentive compatible on-line auction for many identical items with a tight competitive ratio. They consider both revenue and social welfare targets.

Awerbuch, Azar, and Myerson [1] give a general scheme that produces posted prices for general combinatorial auctions, with a competitive ratio equal to the logarithm of the ratio between highest and lowest prices, times the underlying competitive ratio for the combinatorial auction.

Although not explicitly stated as a pricing scheme, [14] effectively gives a dynamic pricing scheme for 2 servers in any metric space. Dynamic pricing was used in the context of packets with values and deadlines [12] with the goal of maximizing social welfare. Dynamic subsidies were introduced in [6] in the context selfish agents and facility locations. In [9] selfish agent versions were introduced for metrical task systems [4], for the k-server problem [19] on the line, and for metrical matching [15] on the line, and appropriate dynamic pricing schemes were described for reducing social cost. Dynamic pricing for scheduling selfish agents on related machines to minimize makespan were studied in [11]. In [13] dynamic prices were used to give a good approximation to the maximal flow time. In [10] dynamic prices were used to approximate the sum of weighted completion times. Many problems and extensions remain open.

1.1.2 The k-server problem

The k-server problem was introduced by Manasse et al. [19] as a far reaching generalization of various online problems. The best-studied of those is the paging (caching) problem, which corresponds to k-server problem on a uniform metric space. Sleator and Tarjan [20] gave several k-competitive algorithms for paging and proved that this is the best possible ratio for any deterministic algorithm.

The famous *k-server conjecture* of Manasse et al. [19] hypothesizes that the k-server problem is no harder in other metric spaces, i.e., that k is the optimal ratio for deterministic algorithms in general metrics. A lower bound of k holds in any metric space of at least $k+1$ points [19], and a nearly matching upper bound of $2k-1$ was given for the Work Function Algorithm (WFA) by Koutsoupias and Papadimitriou [17], which remains the best known algorithm for general metrics. The conjecture has been settled (exactly) for several special metrics. In particular, Chrobak et al. [7] gave an elegant k-competitive algorithm for the line metric, called Double Coverage (DC), which was later extended and shown to be k-competitive for all tree metrics [8]. Additionally, Bartal and Koutsoupias have shown that WFA is k-competitive for the line, the star, and all metric spaces with $k+2$ points [3].

Moreover, Bansal et al. [2] have recently shown that the exact competitive ratio of the DC algorithm, which we simulate by dynamic pricing scheme, when it uses k servers but the offline optimum uses only $h \leq k$ servers is $\frac{k(h+1)}{k+1}$. (For such setting, the general lower bound is $\frac{k}{k-h+1}$ [19], which is matched only for the special case of paging [20].)

Most results on the k-server problem can be found in the survey by Koutsoupias [16]. Due to our focus, we ignore the randomized variant, on which there is significant recent progress [5].

1.2 Roadmap to this Paper

The next section, Section 2 gives the model and sufficient condition to give of competitive pricing algorithms on trees. We show that any algorithm that is *lazy* and *monotone* can be used to derive a dynamic pricing scheme, and that a dynamic pricing scheme implies that such an algorithm must exist. Section 3 gives an algorithm that is clearly lazy and monotone, but it remains to show that all points on the tree are associated with some server, *i.e.*, that the algorithm is well defined. This is shown in Section 4. In Section D (in the Appendix) we show that the algorithm of Section 3 can be implemented in polynomial time. The Appendix also contains full proofs of various claims.

2 The Model and Preliminaries

2.1 The Selfish k-server problem

In this problem, there is a set of k-servers located in some metric space defined by an undirected weighted tree $T = (V, E, w)$. A sequence of selfish requests $\sigma = \langle \sigma_1, \sigma_2, \ldots, \rangle$ arrives online, where each request is issued at some point in the metric space. Before an arrival of each request, a dynamic pricing scheme sets a surcharge (price) on each server, and the arriving request chooses to be served by the server s that minimizes the sum of the distance of s from the request and the surcharge on s; the server s is then moved to the request. The dynamic pricing scheme's objective is to minimize the total distance moved by all servers.

Formally, given a request sequence $\sigma = \langle \sigma_1, \sigma_2, \ldots, \sigma_T \rangle$, each of the requests must be served by one of the k servers, let $\ell = \langle \ell_1, \ell_2, \ldots, \ell_T \rangle$ denote the *solution sequence*, where $\ell_i \in \{1, \ldots, k\}$ is the index of the server which serves the i-th request. Define the *event prefix* $\sigma^{\prec t}$ to be the sequence of events up to but not including event t: $\sigma^{\prec t} = \langle \sigma_1, \sigma_2, \ldots, \sigma_{t-1} \rangle$. The servers location after request t is: $s_i(\sigma^{\prec t+1}) = s_i(\sigma^{\prec t})$ for $i \neq \ell_t$ and $s_{\ell_t}(\sigma^{\prec t+1}) = \sigma_t$. Let $s_i(\sigma^{\prec 1})$ denote the initial server location.

The cost of serving σ by the solution sequence ℓ is

$$\mathbf{COST}(\sigma, \ell) = \sum_{t=1}^{T} \mathsf{dist}(\sigma_t, s_{\ell_t}(\sigma^{\prec t})).$$

In the selfish setting, the server that serves the request σ_t in step t is chosen so as to minimize the distance of σ_t to the server's current location *plus* the surcharge function c: $\sigma^{\prec t} \times \{1, \ldots, k\} \mapsto \mathbb{R}^+$ (i.e., c depends only on past events). The chosen server is:

$$\ell_t^c \in \arg\min_i \mathsf{dist}\left(\sigma_t, s_i(\sigma^{\prec t})\right) + c(\sigma^{\prec t}, i).$$

Let $\ell^c = \langle \ell_1^c, \ldots, \ell_t^c \rangle$ be the (solution) sequence of server indices chosen by the selfish requests σ, and let $\ell^* = \langle \ell_1^*, \ldots, \ell_t^* \rangle$ be the servers that minimize the total cost for σ. A pricing scheme c is α-competitive if for any σ:

$$\frac{\mathbf{COST}(\sigma, \ell^c)}{\mathbf{COST}(\sigma, \ell^*)} \leq \alpha.$$

2.2 A Sufficient Condition for Competitive Pricing Algorithms on trees

In this paper, we focus on tree metrics, where given a weighted tree $T = (V, E, w)$, we define a tree metric space to include the vertices of T along with all points along the edges of T (see Fig. 3a in Appendix A). Given two points $a, b \in T$, we denote by $\mathcal{P}[a, b]$ the [unique] path between a and b including both endpoints. We use $\mathsf{dist}(a, b)$ to denote the distance between a and b defined by the metric. We also use $\mathcal{P}(a, b]$ to denote the path from a to b that is open at a and closed at b.

We avoid reasoning about prices by describing how any online algorithm of a certain form can be converted into a dynamic pricing scheme that nudges the [upcoming] selfish agent do exactly as the online algorithm.

We use the following two properties. We say that an online algorithm is

1. *lazy* if it moves at most one server.
2. *monotone* if a server i, located at s_i, serves a point p, then it also serves all the points along the path $\mathcal{P}[s_i, p]$.

The following lemma shows that any algorithm that satisfies the above properties can be translated into a dynamic pricing scheme with the same competitive ratio. We sketch the proof below for a "degenerate" case, and we defer the full proof to Appendix C.

▶ **Lemma 1.** *Given a lazy and monotone online algorithm for the k-server problem on tree metrics, with a competitive ratio of α, there is a dynamic pricing scheme for the k-server problem on tree metrics, with the same competitive ratio.*

Proof sketch. Just before the arrival of some request σ_t (and after serving $\sigma^{\prec t}$), every server s has an associated subtree T_s of points such that for every point $p \in T_s$ if the next request were made at p, then s would serve it; we say that s is *responsible* for T_s (breaking ties lexicographically in case multiple servers are at a request's location). These subtrees partition the whole tree metric, i.e., they are disjoint and their union is the entire tree.

First, we set the price for servers for which $T_s = \emptyset$ at ∞. Next, we observe that when setting the surcharges it is sufficient to consider just the endpoints of the subtrees. We say that two non-empty subtrees, T_s and $T_{s'}$, are *touching* at an endpoint p if there is no server s'' such that in the paths from s to p and from s' to p in T contain a point $q(\neq p) \in T_{s''}$. Note that there may be many mutually touching subtrees.

Consider a maximal collection of non-empty subtrees $T_{s_1}, T_{s_2}, \ldots, T_{s_k}$, which pairwise touch at an endpoint p. (Clearly, p belongs to one of those subtrees.) The key observation is that a selfish agent requesting service at p must be indifferent between choosing any of the servers $s_1, \ldots, s_k$. This induces a set of linear equations giving the difference in the surcharges, $c(s_i) - c(s_j)$,

$$\begin{aligned} \operatorname{dist}(s_i, p) + c(s_i) &= \operatorname{dist}(s_j, p) + c(s_j) \quad \text{for all } 1 \le i < j \le k \\ \Rightarrow c(s_i) - c(s_j) &= \operatorname{dist}(s_j, p) - \operatorname{dist}(s_i, p) \quad \text{for all } 1 \le i < j \le k. \end{aligned} \tag{1}$$

The relationship of subtrees "touching" can itself be described as a tree, so the equations above (1) can all be simultaneously satisfied. Any solution gives the prices we need. ◀

The above argument is incomplete, as when subtrees touch at tree vertices, or at at a server's location, the selfish request may deviate from the prescribed behavior of the algorithm. This issue can be treated easily by "nudging" the subtrees to avoid these phenomena. More on this in Appendix C.

How to find a lazy and monotone algorithm

Any non-lazy algorithm can be trivially transformed into a lazy algorithm simply by delaying the motion of a server that is not serving a request. However, this may result in a server serving a non-empty set of points that does not include its location, contradicting monotonicity. Rather than simply following the simulation, we do as in [9][1], one may move any server matched to the simulated server in a min cost matching – this is guaranteed to preserve the competitive ratio. We show below that monotonicity can be preserved by choosing an appropriate matching. Given an online algorithm A and a set of requests $\overline{\sigma}$, let $\mathsf{cost}(A, \overline{\sigma})$ be the cost of A for serving $\overline{\sigma}$.

▶ **Lemma 2** ([9], Lemma 4.3)**.** *Let* ON *be an online algorithm, let $\mathsf{on}_i^{\prec t}$ be the location of server i after* ON *serves requests $\sigma^{\prec t}$, and let* LAZY *be an algorithm that serves request σ_t by the server ℓ which is matched to σ_t in an arbitrary min-cost matching between $\{\mathsf{on}_i^{\prec t+1}\}_{i\in[k]}$ and $\mathbf{s}^{\prec t}$, where the latter is a vector of locations of* LAZY*'s servers after serving $\sigma^{\prec t}$. Then $\mathsf{cost}(\mathsf{LAZY}, \sigma^{\prec t}) \le \mathsf{cost}(\mathsf{ON}, \sigma^{\prec t})$ for every t.*

[1] Originally shown for the line, but the proof works for any metric space, which we show in Appendix B for completeness.

The above lemma suggests a natural approach to find an algorithm with the desired properties. The approach is to simulate an algorithm that does not satisfy these properties (in our case, the Double Cover algorithm discussed in Section 2.4), and whenever the simulated algorithm serves the request with one of its *simulated servers*, choose a *real server* that is matched to the simulated server in a min-cost matching. While this solution produces a lazy algorithm with the same competitive ratio, it is not a-priori clear *if such a server can be chosen in a way that results in a monotone algorithm*. We show that for the Double Cover algorithm, this can indeed be done.

2.3 Characterization of min-cost matching on trees

We now give a full characterization of min-cost matchings on trees. As mentioned, the matching between two sets of points P and Q ($|P| = |Q|$) in a tree metric T is more involved than in a line, as given a point $p \in P$, there can be multiple points in Q local to p that can be matched to p in a min-cost matching between P and Q. Figure 1 contains a simple example.

In order to characterize the min-cost matching we use the following definition to "cut" a tree T at point x to two trees: $T_x(p), \overline{T}_x(p)$, where $p \in T_x(p)$. Formally,

▶ **Definition 3.** *Given a tree T and two distinct points $p, x \in T$, let $T_x(p)$ be the subtree that contains p and does not contain x when splitting T into two subtrees at point x. Let $\overline{T}_x(p)$ be $T \setminus T_x(p)$.*

We define the lowest common ancestor of two points p and q in the tree when rooted at point r.

▶ **Definition 4.** *The **lowest common ancestor** of two points p, q with respect to a point r, as $\mathsf{LCA}_r(p,q) = \mathsf{argmax}_{x \in T}\{\mathsf{dist}(x,r) : x \in \mathcal{P}(p,r) \cap \mathcal{P}(q,r)\}$.*

The following Lemma gives necessary and sufficient conditions for a point $p \in P$ to be matched to $q \in Q$ in some min cost matching.

▶ **Lemma 5.** *Let P and Q be two sets of points in T such that $|P| = |Q|$, and let $p \in P$ and $q \in Q$. Then there exists a min-cost matching $\mathcal{M} : P \to Q$ that matches p to q if and only if the following holds – when considering every point $x \neq q$ on the path from p to q, $\left|\overline{T}_x(q) \cap P\right| > \left|\overline{T}_x(q) \cap Q\right|$.*

The following structural lemma is used in our proofs (we defer both proofs to Appendix E).

▶ **Lemma 6.** *Let P, Q be two sets of points in T ($|P| = |Q|$). For points $q, r \in T$, let $T_r(q)$ be a sub-tree such that $|T_r(q) \cap P| > |T_r(q) \cap Q|$. Then there exists $p \in T_r(q) \cap P$ such that for all $x \in \mathcal{P}(p,r)$, $\left|\overline{T}_x(r) \cap P\right| > \left|\overline{T}_x(r) \cap Q\right|$.*

2.4 The Double Cover algorithm

In order to achieve an optimal deterministic bound, our surcharge algorithm simulates the Double Cover (DC) algorithm on trees [8]. In [8], the following was shown.

▶ **Theorem 7** ([8])**.** *The Double Cover algorithm is k-competitive.*

The algorithm roughly works as follows: When a request is issued at some point r, move all the servers that "see" r (have no other server on the path to r) at the same speed until either (i) a server d is blocked by another server c that moves towards r, in which case d no longer "sees" r and will cease moving towards r (and all servers that see r will continue moving towards r), or (ii) a server d reached r's position, in which case, the servers stop moving, and d serves r.

We use the following notation throughout the paper. The locations of the Double Cover servers, $\mathsf{dc}_i(\sigma^{\prec t}) \in M$, $i = 1, \ldots, k$, determine the "area of responsibility" for every Double Cover server: should some request occur at point $p \in M$, there is at least one server i at $\mathsf{dc}_i(\sigma^{\prec t})$ that will be used by the Double Cover algorithm to serve the request at p. If the time t and requests $\sigma^{\prec t} = \sigma_1, \ldots, \sigma_{t-1}$ are fixed, we can simplify notation as follows:

$$\begin{aligned} s_i &= s_i(\sigma^{\prec t}), \qquad i = 1, \ldots, k, \\ \mathsf{S} &= \langle s_1, \ldots, s_k \rangle \\ \mathsf{dc}_i &= \mathsf{dc}_i(\sigma^{\prec t}), \\ \mathsf{DC} &= \langle \mathsf{dc}_1, \ldots, \mathsf{dc}_k \rangle \\ \mathsf{dc}_i(r) &= \mathsf{dc}_i(\sigma^{\prec t} r) \qquad r \in T, \\ \mathsf{DC}(r) &= \langle \mathsf{dc}_1(r), \ldots, \mathsf{dc}_k(r) \rangle. \end{aligned}$$

In [9], we showed that for the line metric, exactly one of the two adjacent *real* servers to the request can be matched to the simulated server at the request (Lemma 4.2 in [9]). Moreover, if we use DC on the line as ON, serving the request σ_t using the adjacent real server that is matched to σ_t recovers monotonicity (Lemma 4.4 in [9]). For the case where the underlying metric is a tree, this is much more involved, as there can be multiple adjacent real servers that can be matched to σ_t in a min cost matching, and choosing the wrong one might result in a violation of monotonicity, as shown in Figure 1. In Section 3, we define a binary relation $\succ_r$ on pairs of servers that can serve a request at point r such that if $i \succ_r j$, then server i cannot cause a monotonicity issue with respect to server j (more on that in the relevant section). Since $\succ_r$ is a strict order(see Lemma 15), there exists a server that is maximal with respect to $\succ_r$, and using this server would not cause monotonicity issue.

The following property on the movement of the double cover servers on trees that is used to prove the correctness of our algorithm.

▶ **Lemma 8.** *For any DC server* dc_i*, and any point* $r \in T$*: If* dc_i *does not serve the request at* r *(*$\mathsf{dc}_i(r) \neq r$*), then for any* $p \notin T_r(\mathsf{dc}_i)$ *we have* $\mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(p)] \subseteq \mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(r)]$.

Proof. Consider the trail of a DC server moving in response to a request. Observe that every point along the trail was closer to the (former) location of the DC server than to the (former) location of any other DC server. That is:

$$\text{For all } \mathsf{dc}_j, r \in T, \text{for every } q \in \mathcal{P}(\mathsf{dc}_j, \mathsf{dc}_j(r)], \mathsf{dist}(\mathsf{dc}_j, q) < \mathsf{dist}(\mathsf{dc}_z, q) \text{ for all } z \neq j. \quad (2)$$

Let $\mathsf{dc}_j(r, t)$ be the position of server j after a movement of at most t units for a request r, or the maximum movement the server can make if it is blocked before moving t unites. Let $t_j(r)$ be the distance traversed by dc_j for the request r, i.e., $t_j(r) = \mathsf{dist}(\mathsf{dc}_j, \mathsf{dc}_j(r))$. Since $p \notin T_r(\mathsf{dc}_i)$, the following holds:

$$\text{For all } \mathsf{dc}_j \in T_r(\mathsf{dc}_i),\ t' \leq t_j(r) : \mathcal{P}[\mathsf{dc}_j, \mathsf{dc}_j(p, t')] \subseteq \mathcal{P}[\mathsf{dc}_j, \mathsf{dc}_j(r, t')]. \quad (3)$$

We will prove that $t_i(p) \leq t_i(r)$ and by (3) the condition holds. Let b be the DC server that blocks i, i.e. $\mathsf{dc}_b(r, t_i(r)) \in \mathcal{P}(\mathsf{dc}_i(r, t_i(r)), r)$, and let $y = \mathsf{dc}_b(r, t_i(r))$.

Case 1: $\mathsf{dc}_b \in T_r(\mathsf{dc}_i)$ and $t_b(p) \geq t_i(r)$. By (3), $\mathsf{dc}_b(p, t_i(r)) = y \in \mathcal{P}(\mathsf{dc}_i(p, t_i(r)), p)$, so dc_b block dc_i at $t_i(r)$ when the request is at p.

Case 2: $\mathsf{dc}_b \in T_r(\mathsf{dc}_i)$ and $t_b(p) < t_i(r)$. Let dc_ℓ the server which blocked dc_b, by (2) we have $\mathsf{dc}_\ell(p, t_b(p)) \notin \mathcal{P}(dc_b, y)$. Hence, $\mathsf{dc}_\ell(p, t_b(p)) \in \mathcal{P}(y, p) \subseteq \mathcal{P}(\mathsf{dc}_i(p, t_b(p)), p)$ so dc_ℓ block dc_i at $t_b(p) < t_i(r)$ when the request is at p.

Let $x = \mathsf{LCA}_p(r, \mathsf{dc}_b)$ and $t_b^x = \mathsf{dist}(t_b, x)$. Note that if $\mathsf{dc}_b \notin T_r(\mathsf{dc}_i)$ then $t_b^x \leq t_i(r)$.

Case 3: $\mathsf{dc}_b \notin T_r(\mathsf{dc}_i)$ and $t_b(p) \geq t_b^x$. Hence, $\mathsf{dc}_b(p, t_b^x) = x$ and $x \in \mathcal{P}(r,p) \subseteq \mathcal{P}(dc_i(p, t_b^x), p)$ so dc_b blocks dc_i at $t_b^x \leq t_i(r)$ when the request is at p.

Case 4: $\mathsf{dc}_b \notin T_r(\mathsf{dc}_i)$ and $t_b(p) < t_b^x$. Let dc_ℓ the server which blocked dc_b. By (2), $\mathsf{dc}_\ell(p, t_b(p)) \notin \mathcal{P}(dc_b, x)$ hence $\mathsf{dc}_\ell(p, t_b(p)) \in \mathcal{P}(x,p) \subseteq \mathcal{P}(dc_i(p, t_b^x), p)$ so dc_ℓ blocks dc_i at $t_b(p) < t_i(r)$ when the request is at p. ◀

3 An Algorithm for Dynamic Pricing on Trees

We now present a lazy and monotone k-competitive algorithm. This is a "new" (optimal) algorithm for the k-server problem on trees. As mentioned, our goal is to find a region for each server, such that for any request in the region, there exists a min cost matching which matches the server to the dc server at the request (*after* the movement of the dc servers). Note that, for some requests more than one server can be matched to the request. Figure 1 contains a simple example. Moreover, the figure shows that the naïve approach that matches an arbitrary min-cost server to the DC server serving the request produces non-monotonicity. We need to select the real server to move more carefully – this is the purpose of the precedence relation, $\succ_r$.

Recall the the definition of a lowest common ancestor (LCA) (Definition 4). We now define the precedence relation that is used to determined which of the servers in the min-cost matching to the DC server that serves the request can be used to serve the request. Roughly speaking, a server i *precedes* server j with respect to point r ($i \succ_r j$) if, when inspecting the LCA of i and j with respect to point r, there is a DC server ℓ that comes from j's subtree and leaves the LCA towards r. The intuition behind this definition is as follows. Suppose we choose j as the server that serves r (when j is in the min-cost matching to the DC server that serves r). If the request is at a point r' further away from r, DC server ℓ might not leave the LCA, preventing server j from being in a min-cost matching to the DC server that serves the request at r', which might result in non-monotonicity. This situation is exactly the one depicted in Figure 1.

▶ **Definition 9.** *We say that server $i \succ_r j$ (i has higher priority than j with respect to r) if (i) $\mathsf{LCA}_r(s_i, s_j) \neq s_j$, and (ii) there exists some DC server ℓ such that:*

$$\mathsf{LCA}_r(s_i, s_j) \in \mathcal{P}[\mathsf{dc}_\ell, \mathsf{dc}_\ell(r)] \quad \textit{and} \quad \mathsf{dc}_\ell \in T_{\mathsf{LCA}_r(s_i,s_j)}(s_j).$$

▶ **Definition 10.** *We define*

$$\mathsf{MC}(r) = \{\ell : \exists \textit{ min-cost matching } \mathcal{M} : \mathsf{S} \to \mathsf{DC}(r) \textit{ such that } \mathcal{M}(s_\ell) = r\}$$

to be the set of servers that can be matched to the DC server serving the next request located at r.

Accordingly, we define:

▶ **Definition 11.** *A point $r \in T$ is **ℓ-colorable** for some server ℓ:*

1. $\ell \in \mathsf{MC}(r)$.
2. *There is no server j such that $j \in \mathsf{MC}(r)$ and $j \succ_r \ell$.*

The intuition behind the above definition is that Property 1 ensures that the conditions for Lemma 2 hold and thus the algorithm is k-competitive. Finally, Property 2 ensures that the algorithm is monotone and well-defined, as we will show. See Figure 2 in Section A for illustrations of the various definitions made above.

Algorithm 1 The Monotone Regions algorithm (see Fig. 3 in Appendix A) for illustration.

Input: A tree metric T, initial servers locations $\langle s_1(\emptyset), \ldots s_k(\emptyset)\rangle \in M^k$, and an online sequence of requests $\overline{\sigma} \in T^*$.

1. After serving $\sigma^{\prec t}$, before the current request σ_t is revealed:
   a. Initialize the forest $F^0 \leftarrow T$
   b. For $i = 1, \ldots, k$:
      i. $C_i \leftarrow \{p \in F^{i-1} : p \text{ is } i\text{-colorable}\}$
         # C_i is the set of points that are i-colorable in the current forest F^{i-1}.
      ii. $R_i \leftarrow \{ p \in C_i \ : \text{ for all } q \in \mathcal{P}(p, s_i),\ q \in C_i \}$
         # R_i is the monotone region of C_i around the location of server i.
      iii. $F^i \leftarrow F^{i-1} \setminus R_i$
         # F_i is the remaining forest after removing R_i.
2. Let σ_t be the current request, and let $\ell \in [k]$ be the server such that $\sigma_t \in R_\ell$
   - Serve σ_t with server ℓ
   - $\mathsf{dc}_{t+1} \leftarrow \mathrm{DC}(\mathsf{dc}_t, \sigma_t)$

Our algorithm is described in Algorithm 1. We remark that it is not obviously poly-time. In particular, it may not be clear how R_i's can be computed efficiently. However, we describe how to implement the algorithm in poly-time in Appendix D.

We say that our algorithm is *well defined* if for every sequence $\sigma^{\prec t}$, for every point $x \in T$, there exists a server i such that $x \in R_i$.

▶ **Theorem 12.** *There exists a dynamic pricing scheme for the selfish k-server problem on trees with an optimal competitive ratio of k.*

Proof. Assuming Algorithm 1 is lazy, monotone and well defined, it can be simulated by a pricing scheme by Lemma 1 and it is k-competitive by Lemma 2, because a point $r \in T$ is served by server ℓ only if r is in R_ℓ, and therefore r is ℓ-colorable, which implies $\ell \in \mathsf{MC}(r)$. The algorithm laziness follows by definition and the monotonicity of the algorithm follows by step 1(b)ii of Algorithm 1, since the region contains only points p such that all other points on the path from p to the server are also in the region of the server[2]. To conclude the proof, Lemma 13 below implies the algorithm is well-defined. ◀

4 Algorithm 1 is Well Defined

In this section, we show that Algorithm 1 is well defined, i.e. that every point in the tree would be in some server's region, concluding the proof of Theorem 12. To help the reader in following this section, various figures, depicting important lemmas of this section, are presented in Figure 4 of Section A.

▶ **Lemma 13** (Well-Defined Lemma). *For any sequence σ, Algorithm 1 is well-defined.*

[2] We note that C_i itself might not be continuous, and therefore, step 1(b)ii is needed in order to ensure monotonicity.

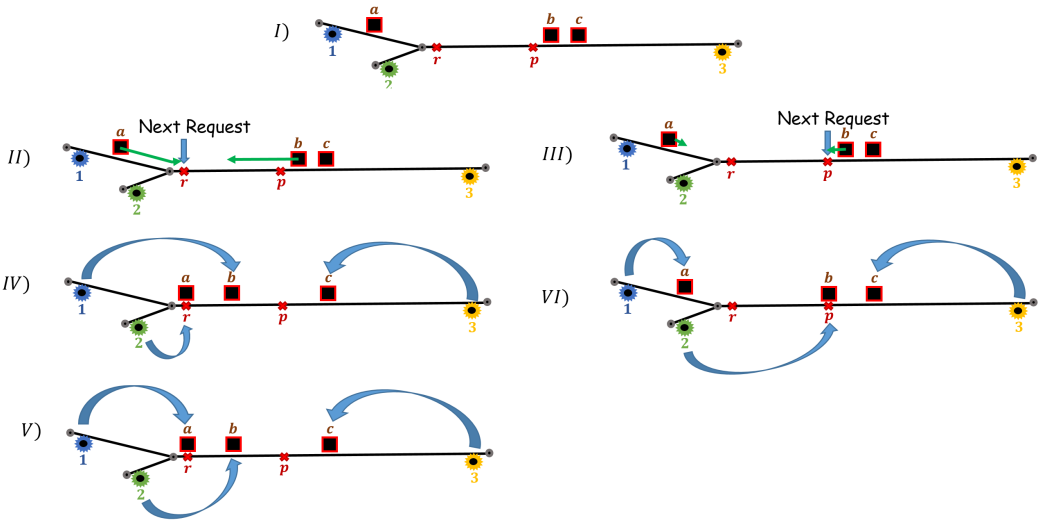


Figure 1 In order to maintain double cover's (DC) competitive ratio, we want to serve each request with a real server that "sees" the request (has no intermediate real servers along the path to the request), and is matched to a DC server that serves the request in a min cost matching between the *real* servers and the *simulated* DC servers. Unfortunately, choosing an arbitrary real server that is matched to the DC server might violate monotonicity. In the figure above DC servers are depicted by squares, namely a, b, c, and real servers by circles, namely $1, 2, 3$. Figure I depicts the initial configuration. We consider two possible locations of the next request, r, p. If the next request is at r, depicted in Figure II, then after the DC servers move, server a which served the request can either be matched to the green(2) server (Figure IV), or to the blue(1) server (Figure V) in the min-cost matching. If one chooses to serve the request with the blue(1) server, then it violates monotonicity. This is since if the next request in the initial configuration is on p (Figure III) instead, then the unique min-cost matching matches the green(2) server to server b. Finally, note that in the initial configuration r *is not* blue(1) colorable. According to Definition 11, properties 1 and 2 hold for the blue(1) server, but property 3 does not since $(2) \in MC(r)$ and $(2) \succ_r (1)$ (DC server a traverses $\mathsf{LCA}_r(1,2)$ and "arrives" from the blue(1) server subtree).

We use the following observation:

▶ **Observation 14** (See Figure 4a). *From the definition, we observe that for every r, p, q in T ($r \neq p$):*

(1) *For $q \in T_r(p)$, we have: $x \in T_r(p) \iff r \notin \mathcal{P}[x, q]$.*

(2) *For $q \notin T_r(p)$, we have: $x \in T_r(p) \Rightarrow r \in \mathcal{P}[x, q]$.*

In order to prove Lemma 13, we first show that the relation $\succ_r$ is a strict partial order.

▶ **Lemma 15.** *$\succ_r$ is a strict partial order relation for every $r \in T$.*

Proof. In order to show that $\succ_r$ is a strict partial order relation, we need to show it is irreflexive and transitive. (Note that these two properties imply asymmetry.) Since it is clear that $\succ_r$ is irreflexive ($\mathsf{LCA}_r(s_j, s_j) = s_j$ for every $r \in T$ and j), we show that it is transitive.

Assume that $i \succ_r j$ and $j \succ_r \ell$, we prove that $i \succ_r \ell$. Let $L_{i,j} = \mathsf{LCA}_r(s_i, s_j)$ and $L_{j,\ell} = \mathsf{LCA}_r(s_j, s_\ell)$ and $L_{i,\ell} = \mathsf{LCA}_r(s_i, s_\ell)$. Let $\mathsf{dc}_{i,j}$ and $\mathsf{dc}_{j,\ell}$ be the respective dc servers which order the servers, i.e., $L_{i,j} \in \mathcal{P}[\mathsf{dc}_{i,j}, \mathsf{dc}_{i,j}(r)]$ and $\mathsf{dc}_{i,j} \in T_{L_{i,j}}(s_j)$, and $L_{j,\ell} \in \mathcal{P}[\mathsf{dc}_{j,\ell}, \mathsf{dc}_{j,\ell}(r)]$ and $\mathsf{dc}_{j,\ell} \in T_{L_{j,\ell}}(s_\ell)$.

Case 1. $L_{i,j} \in \mathcal{P}[L_{j,\ell}, r]$, hence $L_{i,\ell} = L_{i,j}$ and $T_{L_{i,j}}(s_j) = T_{L_{i,j}}(s_\ell)$, and therefore $L_{i,\ell} \in \mathcal{P}[\mathsf{dc}_{i,j}, \mathsf{dc}_{i,j}(r)]$ and $\mathsf{dc}_{i,j} \in T_{L_{i,\ell}}(s_\ell)$. By Definition 9 $i \succ_r \ell$.

Case 2. $L_{j,\ell} \in \mathcal{P}[L_{i,j}, r]$, hence $L_{i,\ell} = L_{j,\ell}$ and therefore $L_{i,\ell} \in \mathcal{P}[\mathsf{dc}_{j,\ell}, \mathsf{dc}_{j,\ell}(r)]$ and $\mathsf{dc}_{j,\ell} \in T_{L_{i,\ell}}(s_\ell)$. By Definition 9 $i \succ_r \ell$. ◀

This allows us to conclude that every point in the tree T is colorable by some server.

▶ **Corollary 16.** *For any $r \in T$, there exist j such that r is j-colorable.*

Proof. Consider a point $r \in T$. Recall that $\mathsf{MC}(r)$ is the set of servers the can be matched to r in a min-cost matching between S and $\mathsf{DC}(r)$. Since $\succ_r$ is a strict order relation (by Lemma 15), there is a server $\ell \in \mathsf{MC}(r)$ that is maximal with respect to $\succ_r$ in $\mathsf{MC}(r)$, i.e., such that for every server $j \in \mathsf{MC}(r)$, $j \not\succ_r \ell$. Hence, there is a server ℓ for which Properties 1 and 2 of ℓ-colorability hold. ◀

A subtree $\tilde{T}$ is *fully-colorable* if for any point $p \in \tilde{T}$ there exists a server ℓ such that p is ℓ-colorable and $s_\ell \in \tilde{T}$. Since Algorithm 1 preserves monotonicity, it follows that a server would color points only in the subtree containing this server. Therefore, in order to prove that Algorithm 1 is well-defined we need to show that not only the original tree T is *fully-colorable* (Corollary 16), but also that every $\tilde{T} \in F^{i-1}$ is fully-colorable as well.

For the sake of proving this property (Corollary 22), we characterize properties of the min-cost matching $\mathsf{MC}(p)$ and the relation $\succ_p$. First, we now show that for any server ℓ the region in which ℓ is in the min-cost matching is monotone.

▶ **Lemma 17** (See Figure 4b)**.** *For any server ℓ and two points r, p in T such that $p \notin T_r(s_\ell)$, the following holds – if $\ell \in \mathsf{MC}(p)$ then $\ell \in \mathsf{MC}(r)$.*

Proof. We will show that for any point $x \in \mathcal{P}[s_\ell, r]$, if $\mathsf{dc}_j(r) \in T_x(s_\ell)$ then $\mathsf{dc}_j(p) \in T_x(s_\ell)$:

First, we observe that $\mathsf{dc}_j(r) \neq r$ (dc_j does not serve request at r), since $r \notin T_x(s_\ell)$ and $\mathsf{dc}_j(r) \in T_x(s_\ell)$. Then, we observe that $\mathsf{dc}_j \in T_x(s_\ell)$, since $\mathcal{P}(\mathsf{dc}_j(r), x) \subseteq \mathcal{P}(\mathsf{dc}_j, x)$. By Lemma 8, we have $\mathcal{P}[\mathsf{dc}_j, \mathsf{dc}_j(p)] \subseteq \mathcal{P}[\mathsf{dc}_j, \mathsf{dc}_j(r)]$, since $x \notin \mathcal{P}(\mathsf{dc}_j, \mathsf{dc}_j(r))$ ($\mathsf{dc}_j(r) \in T_x(s_\ell)$), we have $x \notin \mathcal{P}(\mathsf{dc}_j, \mathsf{dc}_j(p))$ and we have $\mathsf{dc}_j(p) \in T_x(s_i)$.

We get that for every x in $\mathcal{P}[s_\ell, r]$, if $\mathsf{dc}_j(r) \in T_x(s_\ell)$, then $\mathsf{dc}_j(p) \in T_x(s_\ell)$, which implies $|T_x(s_\ell) \cap \mathsf{dc}(p)| \geq |T_x(s_\ell) \cap \mathsf{dc}(r)|$. Since $\ell \in \mathsf{MC}(p)$, for any $x \in \mathcal{P}[s_\ell, r]$ we have $|T_x(s_\ell) \cap S| > |T_x(s_\ell) \cap \mathsf{dc}(p)|$. Which together yields that the condition of Lemma 5 hold also for $\mathsf{dc}(r)$, and therefore $\ell \in \mathsf{MC}(r)$. ◀

Which yields the following lemma which will be used to prove Lemma 21.

▶ **Lemma 18** (See Figure 4c)**.** *For any two servers b, ℓ and a points x in T such that $b \in \mathsf{MC}(x)$ and $s_\ell \notin T_x(s_b)$ we have for any $p \in \mathcal{P}(s_b, x)$ that $\ell \notin \mathsf{MC}(p)$.*

Proof. Assume towards a contradiction that there exists $p \in \mathcal{P}(s_b, x)$ such that $\ell \in \mathsf{MC}(p)$. Consider a point $y \in \mathcal{P}(x, p)$ which isn't a tree vertex, and in which at most a single DC server will arrive if the request is issued at this point (there exists such a point due to the continuity of the metric space). According to Lemma 17, $\ell, b \in \mathsf{MC}(y)$.

Therefore, by Lemma 6 we have:

$$|T_y(s_b) \cap \mathsf{DC}(y)| < |T_y(s_b) \cap \mathsf{S}|\,, \text{ and}$$
$$|T_y(s_\ell) \cap \mathsf{DC}(y)| < |T_y(s_\ell) \cap \mathsf{S}|\,.$$

Since y is not a tree node, $T = T_y(s_\ell) \cup T_y(s_b) \cup \{y\}$. Moreover, there is at most one $\mathsf{DC}(y)$ server at y (by y's selection), so overall there are more real servers than $\mathsf{DC}(y)$ servers, a contradiction. ◀

The following is an important property of the strict partial order $\succ_r$.

▶ **Lemma 19** (See Figure 4d)**.** *For any two servers ℓ, j, a point r such that $s_j \in T_r(s_\ell)$, and any point $p \notin T_r(s_\ell)$: If $j \succ_p \ell$, then $j \succ_r \ell$.*

Proof. First, since $s_j \in T_r(s_\ell)$ then $\mathsf{LCA}_r(s_\ell, s_j) \in T_r(s_\ell)$, therefore we have that $\mathsf{LCA}_r(s_\ell, s_j) = \mathsf{LCA}_p(s_\ell, s_j)$. Second, $j \succ_p \ell$ therefore there exists dc_i such that $\mathsf{dc}_i \in T_{\mathsf{LCA}_p(s_\ell,s_j)}(s_\ell)$, and $\mathsf{LCA}_p(s_\ell, s_j) \in \mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(p)]$. Clearly, if the request is on r and dc_i serves point r then $\mathsf{LCA}_r(s_\ell, s_j) \in \mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(r)]$. If dc_i does not serves point r, by Lemma 8 we have $\mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(p)] \subseteq \mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(r)]$, and again $\mathsf{LCA}_r(s_\ell, s_j) \in \mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(r)]$. In either case $\mathsf{LCA}_r(s_\ell, s_j) \in \mathcal{P}[\mathsf{dc}_i, \mathsf{dc}_i(r)]$ and by Definition 9 we have $j \succ_r \ell$. ◀

We now prove the main technical lemma used in proving that the algorithm is monotone. The lemma roughly shows the following. Let $r \in T$ be some point that is ℓ colorable by some server ℓ, and let j be another server on the "same side" of ℓ with respect to r. Let p be a point on the other side of ℓ and j with respect to r. The lemma states that if p is j-colorable, then it is also ℓ-colorable (see Figure 4e for a visual depiction).

The significance of this lemma is the following – suppose r is a point that the algorithm decided should be served by some server ℓ (which obviously means r is ℓ-colorable). Since we want our algorithm to be monotone, this immediately disconnects all the points further away from r from the servers that are on the same side as ℓ with respect to r. This would be a problem if there was such a point p that can be served only by servers on the same side as ℓ, but not ℓ itself. The lemma basically shows this situation cannot happen.

▶ **Lemma 20** (See Figure 4e)**.** *For any two servers ℓ, j and two points r, p in T such that $s_j, s_\ell \in \overline{T}_r(p)$: If r is ℓ-colorable and p is j-colorable, then p is ℓ-colorable.*

Proof. Assume for contradiction that p is not ℓ-colorable. We consider the following cases

Case 1. $\ell \in \mathsf{MC}(p)$. By the definition of ℓ-colorable, we have that there is a server i such that $i \in \mathsf{MC}(p)$ and $i \succ_p \ell$. If $s_i \in \overline{T}_r(p)$, then by Lemma 17, $i \in \mathsf{MC}(r)$, and by Lemma 19, $i \succ_r \ell$, Hence r is not ℓ-colorable, a contradiction. Otherwise, $s_i \in T_r(p)$. Let $x = \mathsf{LCA}_p(s_\ell, s_i)$. Note that $r \in \mathcal{P}[s_\ell, p], r \in \mathcal{P}[s_j, p]$ and $r \notin \mathcal{P}[s_i, p]$ by Observation 14. We get that $\mathcal{P}[s_i, p] \cap \mathcal{P}[s_\ell, p] = \mathcal{P}[s_i, p] \cap \mathcal{P}[r, p] = \mathcal{P}[s_i, p] \cap \mathcal{P}[s_j, p]$, hence $\mathsf{LCA}_p(s_j, s_i) = \mathsf{LCA}_p(s_\ell, s_i) = x$. In addition, $T_x(s_\ell) = T_x(r) = T_x(s_j)$, and since $i \succ_p \ell$ we get $i \succ_p j$ by Definition 9. Recall that, $i \in \mathsf{MC}(p)$, therefore p not j-colorable, a contradiction.

Case 2. $\ell \notin \mathsf{MC}(p)$. By Lemma 5, there exists a point x on the path from s_ℓ to p such that

$$|T_x(s_\ell) \cap \mathsf{S}| \leq |T_x(s_\ell) \cap \mathsf{DC}(p)| \,. \tag{4}$$

Let x be the closest point to r for which (4) holds. Since $j \in \mathsf{MC}(p)$, by Lemma 5, for every point y on the path from s_j to p, $|T_y(s_j) \cap \mathsf{S}| > |T_y(s_j) \cap \mathsf{DC}(p)|$, and hence, $x \in \mathcal{P}[s_\ell, \mathsf{LCA}(s_\ell, s_j)] \subseteq \mathcal{P}[s_\ell, r]$. Moreover, since r is ℓ-colorable, $\ell \in \mathsf{MC}(r)$, so Lemma 5 implies that

$$|T_x(s_\ell) \cap \mathsf{S}| > |T_x(s_\ell) \cap \mathsf{DC}(r)| \,. \tag{5}$$

Therefore, combining (4) and (5) yields $|T_x(s_\ell) \cap \mathsf{DC}(r)| < |T_x(s_\ell) \cap \mathsf{DC}(p)|$, and there must be a server dc_a such that $\mathsf{dc}_a \in T_x(s_\ell)$ and $\mathsf{dc}_a(r) \notin T_x(s_\ell) \Rightarrow x \in \mathcal{P}[\mathsf{dc}_a, \mathsf{dc}_a(r)]$. In addition, we have

$$\left|\overline{T}_x(r) \cap \mathsf{S}\right| > \left|\overline{T}_x(r) \cap \mathsf{DC}(p)\right| , \tag{6}$$

since x is the closest point to p for which (4) holds. Combining (4) and (6) yields that in $\widehat{T} = \overline{T}_x(r) \setminus T_x(s_\ell)$ we have $\left|\widehat{T} \cap \mathsf{S}\right| > \left|\widehat{T} \cap \mathsf{DC}(p)\right|$. Notice that for every $b \neq a$ such that $\mathsf{dc}_b \in \overline{T}_x(r)$, we have that $\mathsf{dc}_b(r) \in \overline{T}_x(r)$ since only a single DC server can cross point x. Since $\left|\widehat{T} \cap \mathsf{DC}(p)\right| = \left|\widehat{T} \cap \mathsf{DC}\right|$, by Lemma 8, we get $\left|\widehat{T} \cap \mathsf{DC}(p)\right| = \left|\widehat{T} \cap \mathsf{DC}(r)\right|$. Therefore, $\left|\widehat{T} \cap \mathsf{S}\right| > \left|\widehat{T} \cap \mathsf{DC}(r)\right|$, and Lemma 6 implies that there exists $s_i \in \widehat{T}$ such that for all $z \in \mathcal{P}[s_i, x]$, we have

$$|T_z(s_i) \cap \mathsf{S}| > |T_z(s_\ell) \cap \mathsf{DC}(r)| \,. \tag{7}$$

In addition, (7) holds also for $z \in (x, r)$ by (5), hence, $i \in \mathsf{MC}(r)$. Moreover, since $x = \mathsf{LCA}_r(s_i, s_\ell)$, $x \in \mathcal{P}[\mathsf{dc}_a, \mathsf{dc}_a(r)]$ and $\mathsf{dc}_a \in T_x(s_\ell)$, we also have $i \succ_r \ell$, which combined with $i \in \mathsf{MC}(r)$ is a contradiction to r being ℓ-colorable. ◀

The main lemma to show the property *fully-colorable* is the following:

▶ **Lemma 21.** *For a fully-colorable sub-tree $\tilde{T}$, let $r, p \in \tilde{T}$ be two points and ℓ a server in $\tilde{T}$ such that $p \notin T_r(s_\ell)$. If we have that*

- *r is ℓ-colorable, and*
- *for all servers a such that $s_a \in \tilde{T}$ where p is a-colorable, we have $s_a \in T_r(s_\ell)$,*

then for any $x \in \mathcal{P}(r, p]$, x is ℓ-colorable.

Proof. First, by Lemma 20 we have that p is ℓ-colorable as well. Assume for the purpose of contradiction that it is not true, let $x \in \mathcal{P}(r, p)$ be the closet point to p such that x is not ℓ-colorable. Since $\tilde{T}$ is fully-colorable, there exists a server b, such that $s_b \in \tilde{T}$ and x is b-colorable. Note that, if $s_b \in T_r(s_\ell)$, then $s_b, s_\ell \in \overline{T}_r(x)$, and since r is ℓ-colorable, by Lemma 20, x is ℓ colorable, a contradiction. Let $L = \mathsf{LCA}_r(p, s_b)$

Case 1. One of the following two holds: (i) $x \notin \mathcal{P}(s_b, s_\ell)$, (ii) $x = L$. In this case, $s_b, s_\ell \in \overline{T}_x(p)$ and x is b-colorable. Therefore, by Lemma 20, p is b-colorable, a contradiction.

Case 2. $x \in \mathcal{P}(s_b, s_\ell)$, and $x \neq L$, which implies $s_\ell \notin T_x(s_b)$, and $b \in \mathsf{MC}(x)$ (since x is b-colorable). Therefore, by Lemma 18, we have $\ell \notin \mathsf{MC}(y)$ for any $y \in \mathcal{P}(s_b, x)$, however since $x \neq L$, there exist $z \in \mathcal{P}(x, s_b) \cap \mathcal{P}(x, p)$, on one hand z is ℓ-colorable (by our choice of x), on the other hand $\ell \notin MC(z)$ (since $z \in \mathcal{P}(s_b, x)$), a contradiction. ◀

The above lemma implies the following corollary, yielding that Algorithm 1 is well-defined.

▶ **Corollary 22.** *For a fully-colorable subtree $\tilde{T}$, and i a server such that $s_i \in \tilde{T}$, then for all subtrees $\hat{T} \in \tilde{T} \setminus R_i$ we have that $\hat{T}$ is fully-colorable tree.*

Proof. Let p be the point in $\hat{T}$ for which this does not hold, since $\tilde{T}$ is fully-colorable, let j be the server such that $s_j \in \tilde{T}$ and p is j-colorable. Let $r = \mathsf{argmin}_x\{\mathsf{dist}(p, x) : x \in \mathcal{P}(s_i, p) \cap R_i\}$ be the closest point to p in R_i. Observe that $r \notin \mathcal{P}(s_j, s_i)$ since otherwise $\mathcal{P}(s_j, p) \subseteq \mathcal{P}(s_j, r) \cup \mathcal{P}(r, p)$, where $\mathcal{P}(s_j, r) \cap R_i = \emptyset$ and $\mathcal{P}(r, p) \cap R_i = \emptyset$. Therefore, $\mathcal{P}(s_j, p) \cap R_i = \emptyset$, and thus $s_j \in \hat{T}$, a contradiction. Hence, by Observation 14(1), $s_j \in T_r(s_i)$. Finally, By Lemma 21, the entire $\mathcal{P}(r, p)$ is i-colorable, a contradiction for $p \notin R_i$. ◀

Using this corollary, we can now prove the Well-Defined Lemma.

Proof of Well-Defined Lemma [Lemma 13]. In order for Algorithm 1 to be well-defined, each point in T should be in the R_ℓ region of some server ℓ. We will show that each subtree $\tilde{T} \in F^i$ after iteration i in the run of the algorithm execution is fully-colorable. The initial tree, T is fully-colorable by Corollary 16. After each iteration i, every subtree in F^i is fully-colorable by Corollary 22 (Note that, R_i is a subregion of a single subtree of F^{i-1}). Therefore, eventually a sub-tree would contain a single server and it is fully-colored by this server, which yields that $F^k = \emptyset$ as needed. ◀

References

1 Baruch Awerbuch, Yossi Azar, and Adam Meyerson. Reducing Truth-telling Online Mechanisms to Online Optimization. In *Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 503–510, New York, NY, USA, 2003. ACM. doi:10.1145/780542.780616.

2 Nikhil Bansal, Marek Eliás, Lukasz Jez, Grigorios Koumoutsos, and Kirk Pruhs. Tight Bounds for Double Coverage Against Weak Adversaries. *Theory Comput. Syst.*, 62(2):349–365, 2018. doi:10.1007/s00224-016-9703-3.

3 Yair Bartal and Elias Koutsoupias. On the competitive ratio of the work function algorithm for the k-server problem. *Theor. Comput. Sci.*, 324(2-3):337–345, 2004. doi:10.1016/j.tcs.2004.06.001.

4 Allan Borodin, Nathan Linial, and Michael E. Saks. An Optimal On-Line Algorithm for Metrical Task System. *J. ACM*, 39(4):745–763, 1992. doi:10.1145/146585.146588.

5 Sébastien Bubeck, Michael B. Cohen, Yin Tat Lee, James R. Lee, and Aleksander Madry. k-server via multiscale entropic regularization. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 3–16, 2018. doi:10.1145/3188745.3188798.

6 Niv Buchbinder, Liane Lewin-Eytan, Joseph (Seffi) Naor, and Ariel Orda. Non-Cooperative Cost Sharing Games via Subsidies. *Theor. Comp. Sys.*, 47(1):15–37, July 2010. doi:10.1007/s00224-009-9197-3.

7 Marek Chrobak, Howard J. Karloff, T. H. Payne, and Sundar Vishwanathan. New Results on Server Problems. *SIAM J. Discrete Math.*, 4(2):172–181, 1991. doi:10.1137/0404017.

8 Marek Chrobak and Lawrence L. Larmore. An Optimal On-Line Algorithm for k-Servers on Trees. *SIAM J. Comput.*, 20(1):144–148, 1991. doi:10.1137/0220008.

9 Ilan Reuven Cohen, Alon Eden, Amos Fiat, and Lukasz Jez. Pricing Online Decisions: Beyond Auctions. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 73–91. SIAM, 2015. doi:10.1137/1.9781611973730.7.

10 Alon Eden, Michal Feldman, Amos Fiat, and Tzahi Taub. Truthful Prompt Scheduling for Minimizing Sum of Completion Times. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, pages 27:1–27:14, 2018. doi:10.4230/LIPIcs.ESA.2018.27.

11 Michal Feldman, Amos Fiat, and Alan Roytman. Makespan Minimization via Posted Prices. In *Proceedings of the 2017 ACM Conference on Economics and Computation, EC '17, Cambridge, MA, USA, June 26-30, 2017*, pages 405–422, 2017. doi:10.1145/3033274.3085129.

12 Amos Fiat, Yishay Mansour, and Uri Nadav. Efficient contention resolution protocols for selfish agents. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein, editors, *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 179–188. SIAM, 2007. URL: http://dl.acm.org/citation.cfm?id=1283383.1283403.

13 Sungjin Im, Benjamin Moseley, Kirk Pruhs, and Clifford Stein. Minimizing Maximum Flow Time on Related Machines via Dynamic Posted Pricing. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPIcs*, pages 51:1–51:10. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ESA.2017.51.

14 Sandy Irani and Ronitt Rubinfeld. A Competitive 2-Server Algorithm. *Inf. Process. Lett.*, 39(2):85–91, 1991. doi:10.1016/0020-0190(91)90160-J.

15 Bala Kalyanasundaram and Kirk Pruhs. Online Weighted Matching. *J. Algorithms*, 14(3):478–488, 1993. doi:10.1006/jagm.1993.1026.

16 Elias Koutsoupias. The k-server problem. *Computer Science Review*, 3(2):105–118, 2009. doi:10.1016/j.cosrev.2009.04.002.

17 Elias Koutsoupias and Christos H. Papadimitriou. On the k-Server Conjecture. *J. ACM*, 42(5):971–983, 1995. doi:10.1145/210118.210128.
18 Ron Lavi and Noam Nisan. Competitive Analysis of Incentive Compatible On-line Auctions. In *Proceedings of the 2Nd ACM Conference on Electronic Commerce*, EC '00, pages 233–241, New York, NY, USA, 2000. ACM. doi:10.1145/352871.352897.
19 Mark S. Manasse, Lyle A. McGeoch, and Daniel Dominic Sleator. Competitive Algorithms for Server Problems. *J. Algorithms*, 11(2):208–230, 1990. doi:10.1016/0196-6774(90)90003-W.
20 Daniel Dominic Sleator and Robert Endre Tarjan. Amortized Efficiency of List Update and Paging Rules. *Commun. ACM*, 28(2):202–208, 1985. doi:10.1145/2786.2793.

A Figures

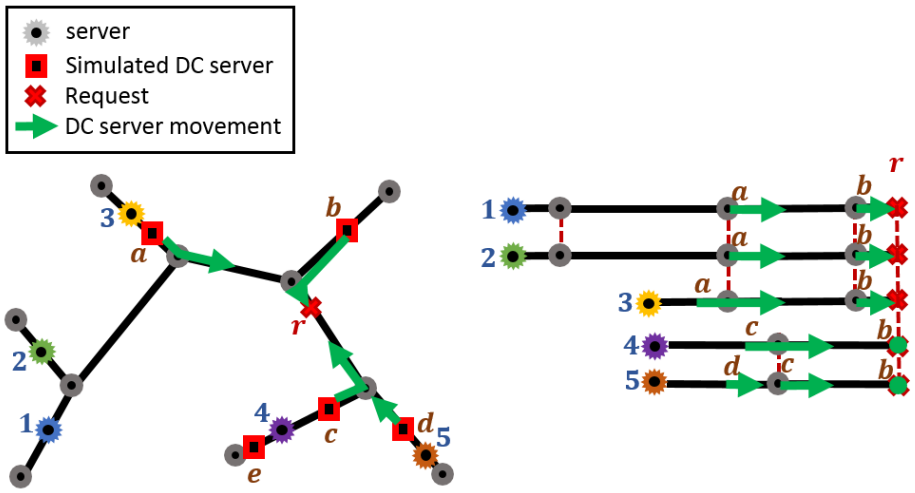
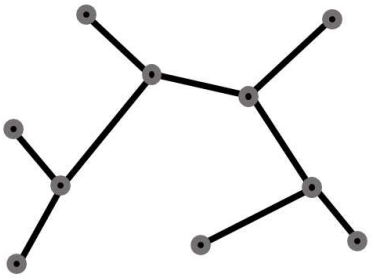
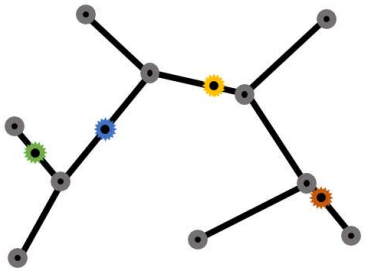


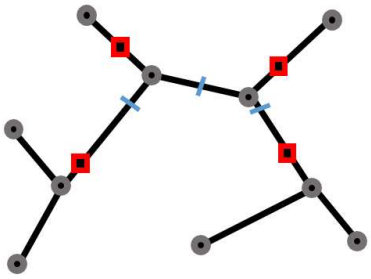
Figure 2 Servers and DC servers are denoted by numbers and letters respectively. Points on the tree are said to be colorable by some set of servers. Colorability of a point r is determined by simulating the double cover (DC) algorithm for a request at r. When DC processes a request, multiple DC servers move towards the request, and one or more arrive to serve it. Imagine a server were to look along the tree towards r when the DC servers were in motion in response to a request at r. Such a server may see a trail left by (at most one) DC server in motion towards r. Different servers may see trails of different DC servers. Two servers see the same trails beyond (above) their lowest common ancestor (when the tree is rooted at r) but for a DC server that traverses their lowest common ancestor, they may observe different trails. We say that server i has higher priority than server j with respect to r, if the trail of the DC server that traverses the lowest common ancestor of i and j is contained in the trail seen by server j (of the same DC server). On the left the movement of the DC servers relative to the real server positions is depicted. On the right, all paths from real servers to r are depicted, with dashed lines indicating vertices seen by more than one real server. In this example, $1 \succ_r 3$ since that trail that server 1 sees of DC server a is contained in the trail that server 3 sees of DC server a. Similarly, $2 \succ_r 3$ (because of a), $5 \succ_r 4$ (because of c), and $4, 5 \succ_r 1, 2, 3$ (because of b). Notice that $\succ_r$ is not defined for all pairs of servers; For example, both $1 \not\succ_r 2$ and $2 \not\succ_r 1$. Subsequent to the motion of the DC servers, there are several min cost matching between real servers and DC servers. In one such matching server 1 is matched to server b, in another such min matching server 2 is matched to server b, in a third such min matching server 3 is matched to server b. Therefore, $\mathsf{MC}(r) = \{1, 2, 3\}$. Since $1 \not\succ_r 2$, $3 \not\succ_r 2$, $2 \not\succ_r 1$ and $3 \not\succ_r 1$. We get that r is $1, 2$-colorable. r is not 3-colorable since $1 \succ_r 3$.



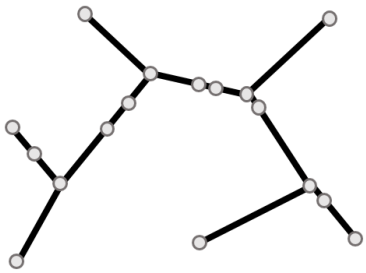
(a) A tree metric.



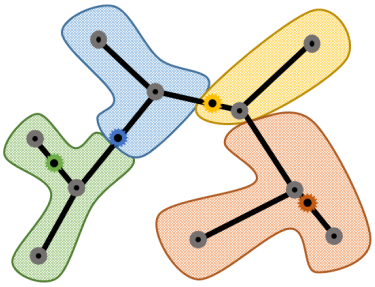
(b) Servers' locations $\mathbf{s}^{\prec t}$.



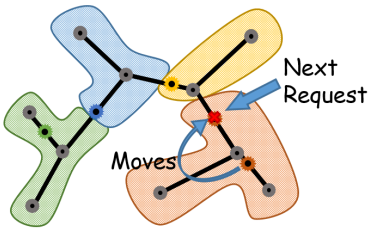
(c) DC servers' locations $\mathsf{dc}^{\prec t}$ with boundary points $B^{\prec t}$.



(d) The critical tree graph $T_c^{\prec t}$.

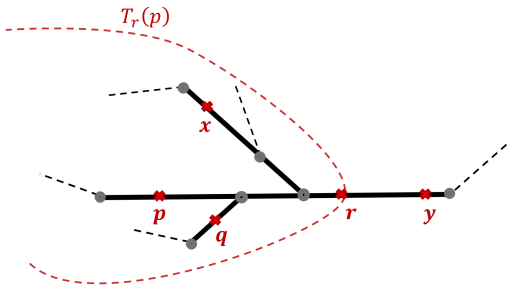


(e) The coloring of the tree as produced by ColorRegion. Notice that the tree is colored irrespective of the next request.

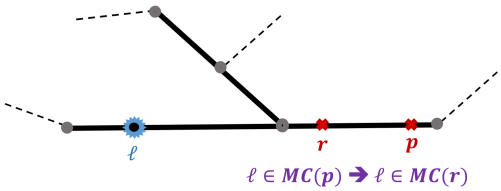


(f) When (next) request σ_t occurs, it is serviced by the server in whose region it is located.

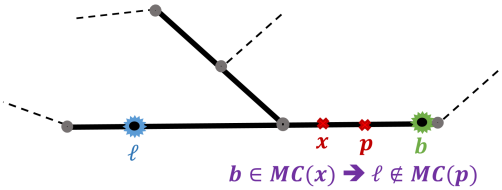
Figure 3 Key ingredients for Algorithm 1.



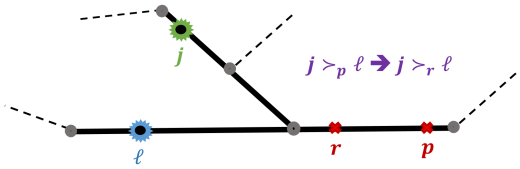
(a) Observation 14.



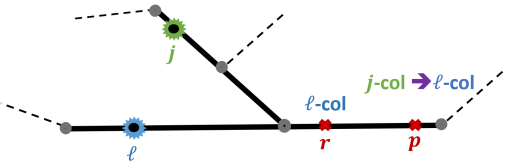
(b) Lemma 17.



(c) Lemma 18.



(d) Lemma 19.



(e) Lemma 20.

Figure 4 A visual depiction of the lemmas used in order to prove the Well-Defined Lemma.

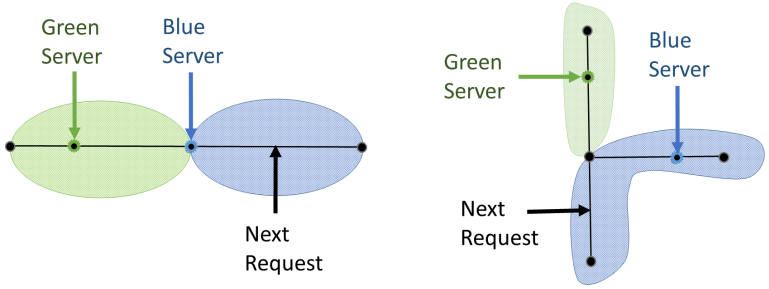


Figure 5 Issues with the naïve pricing algorithm. In the example on the left, the range served by the blue server has the blue server on its left end. The open interval up to the blue server is served by the green server. By setting the surcharges as in the naïve algorithm, a selfish request (the next request) in the blue zone is indifferent between moving the green and blue servers, so we have no guarantee that selfish agents emulate the online algorithm. The figure on the right shows a similar problem where the green and blue regions touch, and, again, by setting the prices naïvely, selfish agents may choose to move either the green or the blue agent in response to a request. In both cases, a solution to this problem is to break the tie by "pushing" the boundary between the green and blue regions slightly "away" from the blue region. See Figure 6 for details.

B Proof of Lemma 2

Proof of Lemma 2. Given two sets of points P, Q such that $|P| = |Q|$, let $w(P, Q)$ be the weight of the min-cost matching between P and Q.

Let $\mathsf{cost}_t(\mathsf{LAZY})$ and $\mathsf{cost}_t(\mathsf{ON})$ be the respective cost of algorithms LAZY and ON when serving request σ_t. We show that for every t,

$$\mathsf{cost}_t(\mathsf{LAZY}) + \Delta\Phi \leq \mathsf{cost}_t(\mathsf{ON}), \tag{8}$$

for a non-negative potential function $\Phi = w(\mathsf{S}, \mathsf{on})$, where S and on are the current locations of the servers of LAZY and ON respectively. To prove (8), it suffices to consider the moves of ON and LAZY independently, in this order.

Fix some min-cost matching $\mathcal{M} : \mathsf{S} \to \mathsf{on}$. We keep $\mathcal{M}$ fixed as ON moves its servers. Clearly, when ON moves a server ℓ by distance d, the cost of $\mathcal{M}$ does not increase by more than d. Hence, the same holds for the min-cost matching. Thus Φ increases by at most d, and (8) holds.

Once ON is done with its moves, we analyze the move of LAZY. Note that at this point $\sigma_t \in \mathsf{on}$, i.e., ON has one of its servers at σ_t. Let $\mathcal{M}'$ be the updated min-cost matching after ON moves, and let ℓ' be some server of LAZY that is matched to σ_t. Upon the move of ℓ' to σ_t, the cost of $\mathcal{M}'$ is decreased by $\mathsf{dist}(s_{\ell'}, \sigma_t)$. Since the cost of the min-cost matching after ℓ' moves is no bigger than that of $\mathcal{M}'$, Φ decreases by at least $\mathsf{dist}(s_{\ell'}, \sigma_t)$ as well, which is exactly $\mathsf{cost}_t(\mathsf{LAZY})$. Therefore, $\mathsf{cost}_t(\mathsf{LAZY}) + \Delta\Phi \leq 0$, and (8) holds. ◀

C Full Argument for Lemma 1

The proof sketch of Lemma 1 shows that one can set surcharges where for the incoming agent there exists a server that minimizes the distance + surcharge *and* this is the same server that the algorithm would choose. Whenever this server can be matched (in a min cost matching) to the DC server that served the request, Lemma 2 implies that the competitive ratio achieved is optimal. This is enough for a truthful online algorithm with optimal competitive ratio if we can break ties for the agent. However, our goal is to let the agents break ties for themselves.

We first notice the are two scenarios where an agent can have more than one disutility minimizing server – (i) either the transition between the responsibility area of server j and adjacent server i is the location of server i (left side of Figure 5). In this case, setting prices using Equation (1) will result in both server i *and* server j being the disutility minimizing servers for the responsibility area of agent i. (ii) the responsibility area of agent i contains a tree vertex x from which starts the responsibility area of agent j (right side of Figure 5, i is blue and j is green). In this case, if a request is made in the responsibility area of agent i but on the other side of x than server i itself (i.e., in $\overline{T}_x(s_i)$), then both server i *and* server j are the disutility minimizing servers for this request.

To resolve this issue, we "nudge" the responsibility area of agent i slightly to the direction of the responsibility area of agent j by an exponentially decreasing tiny ϵ (see Figure 6). We inspect the proof of Lemma 2 to see why this does not change the competitive ratio. Since we do not necessarily use the server that minimizes the min cost matching at the nudged areas, Equation (8) does not hold if the request is in the nudged area. We notice though that this equation is violated by at most $k\epsilon$. To see this, we first move ON to the request. Using the same argument as in Lemma 2, we see that Equation (8) still holds after doing this.

We now move LAZY. Assume LAZY moves some server ℓ'. If the request would have been in the border between two responsibility areas before the nudge, then the cost of the min cost matching would have decreased by at least $\mathsf{dist}(s_{\ell'}, \sigma_t)$ and this would have paid for the

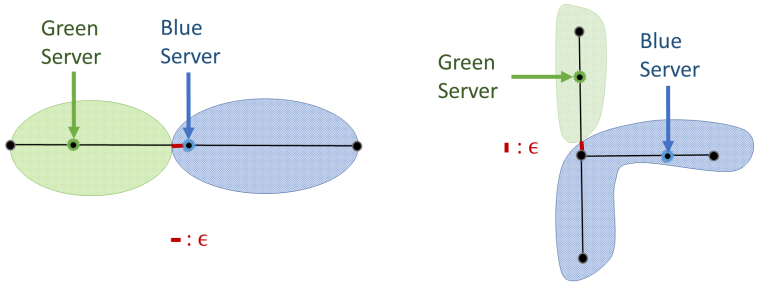


Figure 6 Modifying the regions for which the DC servers are responsible by pushing their boundaries away from real servers and tree vertices. This prevents indifference between different real servers except for isolated points. The boundaries are pushed by small amounts such that even their sum over all regions and all steps is arbitrarily small, thus having no effect on the competitive ratio. See Appendices B and C for the full argument, which uses a potential function.

cost of moving ℓ'. We notice that if the location of a request in DC moves by ϵ, the locations of all servers change by at most ϵ. Therefore, using the same matching in the nudged area as we would have used in the border before the nudge increases the cost of the min cost matching by at most $k\epsilon$. Hence, moving ℓ' decreases the cost of the min cost matching by at least $\mathsf{dist}(s_{\ell'}, \sigma_t) - k\epsilon$, violating Equation (8) by at most $k\epsilon$.

As we can let ϵ exponentially decay (say by a factor of two at each step t), summing Equation (8) for all t's yields that the cost of LAZY is at most $2k\epsilon$ larger than the cost of ON. As ϵ is arbitrarily small, so is the difference between LAZY and ON, which thus have the same competitive ratio.

D Implementation in Polynomial Time

Algorithm 1 as defined in Section 3 is continuous in the sense that every point is considered when deciding which set of points should be in the region R_i of some server i. In this secion, we show that one can discretize the metric space in a way that only polynomially many points (in the number of servers and vertices of the tree) are considered when determining the regions of each server.

Consider a point $p \in T$, such that there exist $1 \leq i < j \leq k$ such that

$$\mathsf{dc}_i(\sigma^{\prec t} \parallel p) = \mathsf{dc}_j(\sigma^{\prec t} \parallel p)$$

(where $\parallel$ denotes concatenation), then p is called a *boundary point*. That is, a boundary point is a point for which, if a request occurs in p, two DC servers will serve the request. Define the set of all boundary points for Double Cover just before event t arrives (see Fig. 3c in Appendix A):

$$B^{\prec t} = \left\{p \mid \exists 1 \leq i < j \leq k \text{ such that } \mathsf{dc}_i(\sigma^{\prec t} \parallel p) = \mathsf{dc}_j(\sigma^{\prec t} \parallel p)\right\}.$$

▶ **Definition 23.** *Given a tree metric $T = (V, E, \mathsf{dist})$, a set of requests $\sigma^{\prec t}$, and the current locations of the servers $\mathsf{S}^{\prec t}$, we define the critical tree graph $T_c^{\prec t}$ by subdividing the edges of the tree (V, E) at all the server locations and boundary points, and retaining the distance function* dist*, see Fig. 3 in Appendix A. Formally:*

- *Define the vertex set of the critical tree graph $T_c^{\prec t}$ to be the set $V_c^{\prec t}$, the union of the following point sets on the tree metric*
  - *Vertices of the tree T.*
  - *Server locations $\left\{\mathsf{S}_\ell^{\prec t}\right\}_{\ell=1,\ldots,k}$.*
  - *The set of boundary points $B^{\prec t}$.*
- *The edge set of $T_c^{\prec t}$ is denoted by $E_c^{\prec t}$. There is an edge $(p,q) \in E_c^{\prec t}$ (where $p \in V_c^{\prec t}$ and $q \in V_c^{\prec t}$) if p and q lie along the same edge of T, and there is no intermediate point $r \in V_c^{\prec t}$ between them. The weight of the edge $(p,q) \in E_c^{\prec t}$ is the distance between p and q in the tree metric T.*

The intuition behind the critical graph is that the vertices of the graph are exactly the points in the metric space where the sets of valid colors ($\{\ell : p \text{ is } \ell\text{-colorable}\}$) change.

▶ **Lemma 24.** *Let $e = \{v_1, v_2\}$ be some edge of $T_c^{\prec t}$, and let ℓ be some server such that $v_1 \in \mathcal{P}[s_\ell, v_2]$ and v_1 is ℓ-colorable. The edge e is ℓ-colorable iff there exists some point p along the edge, excluding the endpoints, such that $\ell \in \mathsf{MC}(p)$.*

Proof. By definition, if e is ℓ-colorable, then for every p along the edge, p is ℓ-colorable, and therefore, $\ell \in \mathsf{MC}(p)$.

Now assume that there exists some p along the edge e such that $\ell \in \mathsf{MC}(p)$. Since there exists some min-cost matching such that s_ℓ is matched to the DC server that serves p, and since p cannot be a vertex of T, by Lemma 5,

$$|T_p(s_\ell) \cap \mathsf{S}| > |T_p(s_\ell) \cap \mathsf{DC}(p)|\,. \tag{9}$$

Since there are no servers and no tree vertices along edge e, for every point $q \in \mathcal{P}[v_1, v_2] \setminus \{v_1, v_2\}$,

$$|T_q(s_\ell)| = |T_p(s_\ell)|\,. \tag{10}$$

For a given $q \in \mathcal{P}[v_1, v_2] \setminus \{v_1, v_2\}$ let

$$d_1(q) = |T_q(v_1) \cap \mathsf{DC}(q)|\,(= |T_q(s_\ell) \cap \mathsf{DC}(q)|)$$

be the set of DC servers in the subtree containing v_1 when splitting T at point q *after* serving a request at q. Let i be the index of the DC server that serves all the requests along the edge e, excluding its endpoints (there must be a unique such DC server since there are no boundary points along e). Notice that for every $j \neq i$, $\mathcal{P}[\mathsf{dc}_j, \mathsf{dc}_j(q)] \cap \mathcal{P}[v_1, v_2] \setminus \{v_1, v_2\} = \emptyset$. Otherwise, there would have been a point q along e which is closer to server j than server i, which implies the existence of a boundary point along e.

Since there are no tree vertices along e, we get that for every $q, q' \in \mathcal{P}[v_1, v_2] \setminus \{v_1, v_2\}$, $d_1(q) = d_1(q')$. Therefore, for every such point q,

$$|T_q(s_\ell) \cap \mathsf{DC}(q)| = d_1(q) = d_1(p) = |T_p(s_\ell) \cap \mathsf{DC}(p)|\,. \tag{11}$$

Combining (9), (10) and (11) yields that for every $q \in \mathcal{P}[v_1, v_2] \setminus \{v_1, v_2\}$, $|T_q(s_\ell) \cap \mathsf{S}| > |T_q(s_\ell) \cap \mathsf{DC}(q)|$. Therefore, $\left|\overline{T}_q(s_\ell) \cap \mathsf{S}\right| < \left|\overline{T}_q(s_\ell) \cap \mathsf{DC}(q)\right|$, and there exists some point $q' \in \mathcal{P}[q, v_2]$ such that

$$\left|\overline{T}_{q'}(s_\ell) \cap \mathsf{S}\right| \leq \left|\overline{T}_{q'}(s_\ell) \cap \mathsf{DC}(q)\right| \Rightarrow \left|\overline{T}_{q'}(q) \cap \mathsf{S}\right| \leq \left|\overline{T}_{q'}(q) \cap \mathsf{DC}(q)\right|.$$

Since there are no servers in $\mathcal{P}[p, v_2]$ (there are no servers along every edge e of $T_c^{\prec t}$), for every server j such that $s_j \in T_q(v_2)$, q' is on the path from s_j to q, and by Lemma 5, $j \notin \mathsf{MC}(q)$. By definition, this implies that for every point q along edge e, and every j such

that $s_j \in \overline{T}_{v_2}(q)$, q is not j-colorable. Since by Corollary 16 every point is colorable by some server, we get that for every q along e, q is ℓ'-colorable by some server ℓ' such that $s_{\ell'} \in \overline{T}_q(v_2) \Rightarrow s_{\ell'} \in \overline{T}_{v_1}(v_2)$. By Lemma 20, since v_1 is ℓ-colorable, we get that every q along the edge e is ℓ-colorable, which implies that e is ℓ-colorable, as desired. ◀

▶ **Lemma 25.** *Let e be some edge $\{v, v'\} \in E_c^{\prec t}$ such that $\mathsf{color}(v) = j$ and $\mathsf{color}(v') = j'$. There exists $i \in \{j, j'\}$ such that all points in $\mathcal{P}[v, v'] \setminus \{v, v'\}$ are i-colorable which can be determined by inspecting a single point in $\mathcal{P}[v, v'] \setminus \{v, v'\}$.*

Proof. consider some edge $e = \{v, v'\} \in E_c^{\prec t}$ such that $\mathsf{color}(v) = j$ and $\mathsf{color}(v') = j'$. Let p be a point between v and v'. By Corollary 16, it is colorable by some server ℓ. Since there are no servers along x, ℓ must be located either in $\overline{T}_v(p)$ or in $\overline{T}_{v'}(p)$. Assume without loss of generality that $\ell \in \overline{T}_v(p)$. By Lemma 20, p is j-colorable, which implies that $j \in \mathsf{MC}(p)$. By Lemma 24, x is j-colorable. ◀

▶ **Lemma 26.** *Determining R_i at every iteration i in Step 1b of Algorithm 1 can be done in polynomial time.*

Proof. Consider the graph $T_c^{\prec t}$. This graph has at most $2k - 1 + |V|$ vertices – k servers, at most $k - 1$ boundary points, and $|V|$ original vertices. The boundary points can of course be computed in polynomial time. Consider iteration i of Step 1b of Algorithm 1. To determine R_i, one can start at s_i, which is obviously in R_i, and then expend R_i using any tree traversal algorithm (that runs in linear time) on $T_c^{\prec t}$. The traversal does not go further down the tree if the vertex/edge currently considered is not i-colorable.

To check if a point $r \in T$ is i-colorable can be done in poly-time: Computing $\mathsf{MC}(r)$ can be done in poly-time using the characterization in Lemma 5. Therefore, property 1 can immediately be checked. For Property 2, one should consider each server $j \in \mathsf{MC}(r)$, and check that $j \not\prec_r i$, which again can be done in poly-time.

From the above, it is clear that determining whether a vertex in $T_c^{\prec t}$ is i-colorable can be done in poly-time. As for an edge, by Lemma 25, checking whether the edge is i-colorable can be done by inspecting an arbitrary point in the edge, and checking whether this point is i-colorable, which again, can be done in poly-time. Therefore, the tree-traversal can be made in poly-time, and so does determining R_i. ◀

E Missing Proofs of Section 4

Proof of Lemma 5. $\Leftarrow$: Let $p \in P$ and $q \in Q$ be two points such that there exists a point $x \in \mathcal{P}(p, q)$ such that $\left|\overline{T}_x(q) \cap P\right| \leq \left|\overline{T}_x(q) \cap Q\right|$ and let $\mathcal{M} : P \to Q$ be a matching such that $\mathcal{M}(p) = q$. Since p is matched to a server in $T_x(q)$, $\left|\overline{T}_x(q) \cap P - \{p\}\right| < \left|\overline{T}_x(q) \cap Q\right|$, and there must be a server $\hat{p} \in T_x(q) \cap P$ that is matched to a server $\hat{q} \in \overline{T}_x(q) \cap Q$. Let $y = \mathsf{LCA}_x(\hat{p}, q)$. Since $\hat{p}$ and q are both in $T_x(q)$, $y \neq x$. Consider the matching $\mathcal{M}'$ in which p is matched to $\hat{q}$, $\hat{p}$ is matched to q, and for every $\tilde{p} \in P \setminus \{p, \hat{p}\}$, $\mathcal{M}'(\tilde{p}) = \mathcal{M}(\tilde{p})$. We have

$$\begin{aligned}
\mathsf{dist}(p, q) + \mathsf{dist}(\hat{p}, \hat{q}) &= \mathsf{dist}(p, x) + \mathsf{dist}(x, y) + \mathsf{dist}(y, q) + \\
&\quad \mathsf{dist}(\hat{p}, y) + \mathsf{dist}(y, x) + \mathsf{dist}(x, \hat{q}) \\
&> \mathsf{dist}(p, x) + \mathsf{dist}(x, \hat{q}) + \mathsf{dist}(\hat{p}, y) + \mathsf{dist}(y, q) \\
&\geq \mathsf{dist}(p, \hat{q}) + \mathsf{dist}(\hat{p}, q),
\end{aligned}$$

where that first equality is due to the fact that the path from x to y is contained in both the path from p to q and the path from $\hat{q}$ to $\hat{p}$, the first strict inequality is due to dropping non-zero terms, and the last inequality follows from the triangle inequality. Therefore, $\mathcal{M}'$ is a matching of a strictly smaller cost than that of $\mathcal{M}$, and $\mathcal{M}$ cannot be a min-cost matching.

$\Rightarrow$: Assume that the condition holds for p, q, let $\mathcal{M}$ be a matching. Let $x = \mathsf{LCA}_q(p, \mathcal{M}(p))$.

Case 1. $x \neq q$, therefore $\left|\overline{T}_x(q) \cap P\right| > \left|\overline{T}_x(q) \cap Q\right|$. Hence, there exists $\hat{p} \in \overline{T}_x(q)$ s.t. $\mathcal{M}(\hat{p}) \notin \overline{T}_x(q)$. Let $\hat{q} = \mathcal{M}(\hat{p})$, and $q' = \mathcal{M}(p)$. Note that $\mathsf{dist}(p, q') = \mathsf{dist}(p, x) + \mathsf{dist}(x, q')$ and $\mathsf{dist}(\hat{p}, \hat{q}) = \mathsf{dist}(\hat{p}, x) + \mathsf{dist}(x, \hat{q})$. Consider the matching $\mathcal{M}'$ in which p is matched to $\hat{q}$, $\hat{p}$ is matched to q', and for every $\tilde{p} \in P \setminus \{p, \hat{p}\}$, $\mathcal{M}'(\tilde{p}) = \mathcal{M}(\tilde{p})$.

$$\begin{aligned}\mathsf{dist}(p, \hat{q}) + \mathsf{dist}(\hat{p}, q') &\leq \mathsf{dist}(p, x) + \mathsf{dist}(x, \hat{q}) + \mathsf{dist}(\hat{p}, x) + \mathsf{dist}(x, q') \\ &= \mathsf{dist}(p, q') + \mathsf{dist}(\hat{p}, \hat{q}),\end{aligned}$$

where the inequality is by the triangle inequality. Therefore, $\mathcal{M}'$ is also a min-cost matching. Let $x' = \mathsf{LCA}_q(p, \mathcal{M}'(p))$ then $\mathsf{dist}(p, x') > \mathsf{dist}(p, x)$ since $x' \notin \overline{T}_x(q)$, therefore we can repeat this process until $x = q$ (**Case 2**).

Case 2. $x = q$, hence $\mathcal{P}(p, q) \subseteq \mathcal{P}(p, \mathcal{M}(p))$. Let $\hat{q} = \mathcal{M}(p)$ and let $\hat{p}$ be such that $q = \mathcal{M}(\hat{p})$. Consider the matching $\mathcal{M}'$ in which p is matched to q, $\hat{p}$ is matched to $\hat{q}$, and for every $\tilde{p} \in P \setminus \{p, \hat{p}\}$, $\mathcal{M}'(\tilde{p}) = \mathcal{M}(\tilde{p})$.

$$\begin{aligned}\mathsf{dist}(p, q) + \mathsf{dist}(\hat{p}, \hat{q}) &= \mathsf{dist}(p, \hat{q}) - \mathsf{dist}(q, \hat{q}) + \mathsf{dist}(\hat{p}, \hat{q}) \\ &\leq \mathsf{dist}(p, \hat{q}) + \mathsf{dist}(\hat{p}, q)\end{aligned}$$

where the last inequality is by the triangle inequality. Therefore, $\mathcal{M}'$ is also min cost matching and $\mathcal{M}'(p) = q$ as needed. ◀

Proof of Lemma 6. Let v be the closest vertex to r in $T_r(q)$ (recall that $r \notin T_r(q)$, so $v \neq r$). If there exists $p \in \mathcal{P}[v, r) \cap P$, let $p \in \mathcal{P}[v, r) \cap P$ be the closest such point to r. In this case, the condition holds for p since for all $x \in \mathcal{P}(p, r)$, $\overline{T}_x(r) \cap P = T_r(q) \cap P$.

If there is no such p, then

$$\left|(\overline{T}_v(r) - \{v\}) \cap P\right| = |T_r(q) \cap P| > |T_r(q) \cap Q| \geq \left|(\overline{T}_v(r) - \{v\}) \cap Q\right|.$$

By the pigeonhole principle, there exists $v' \in \overline{T}_v(r)$ such that $|T_v(v') \cap P| > |T_v(v') \cap Q|$. Therefore, by repeating above process, we find $\hat{p} \in P \cap T_v(v')$ for which the condition holds for all $x \in \mathcal{P}(\hat{p}, v)$. Since the condition holds for every $x \in \mathcal{P}(v, r)$ (as $\overline{T}_x(r) \cap P = T_r(q) \cap P$), the lemma follows. ◀

Approximating the Norms of Graph Spanners

Eden Chlamtáč
Ben Gurion University of the Negev, Beersheva, Israel

Michael Dinitz
Johns Hopkins University, Baltimore, MD, USA

Thomas Robinson
Ben Gurion University of the Negev, Beersheva, Israel

Abstract

The ℓ_p-norm of the degree vector was recently introduced by [Chlamtáč, Dinitz, Robinson ICALP '19] as a new cost metric for graph spanners, as it interpolates between two traditional notions of cost (the sparsity ℓ_1 and the max degree ℓ_∞) and is well-motivated from applications. We study this from an approximation algorithms point of view, analyzing old algorithms and designing new algorithms for this new context, as well as providing hardness results. Our main results are for the ℓ_2-norm and stretch 3, where we give a tight analysis of the greedy algorithm and a new algorithm specifically tailored to this setting which gives an improved approximation ratio.

2012 ACM Subject Classification Theory of computation → Sparsification and spanners

Keywords and phrases Spanners, Approximations

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.11

Category APPROX

Funding *Eden Chlamtáč*: Supported in part by ISF grant 1002/14.
Michael Dinitz: Supported in part by NSF awards CCF-1464239 and CCF-1535887.
Thomas Robinson: Supported in part by ISF grant 1002/14.

1 Introduction

Graph spanners are subgraphs which approximately preserve distances: given a graph $G = (V, E)$ (possibly with lengths on the edges), a subgraph H of G is a *t-spanner* of G if $d_G(u, v) \leq d_H(u, v) \leq t \cdot d_G(u, v)$ for all $u, v \in V$, where d_G denotes shortest-path distances in G (and d_H in H). The value t is called the *stretch* of the spanner.

There have been two traditional ways of studying spanners. The first way is to study universal tradeoffs that can be achieved in all graphs between the stretch and some notion of the "cost" of a spanner, particularly the sparsity [2] or the weight [8]. The second is to study the optimization problem arising from fixing the stretch and trying to optimize the "cost" for the particular given graph. These two lines of work are highly complementary, and have proceeded in parallel. So there is now an extensive line of work on tradeoffs and approximation algorithms for sparsity (total number of edges) and, to a lesser extent, the maximum degree, which are two of the oldest and most well-studied notions of cost.

However, both of these objective functions have drawbacks. If we optimize the sparsity we might end up with a small number of very large degree nodes, which can be a problem for many applications (particularly in distributed systems where the degree is usually related to some notion of "load" on a node). On the other hand, if we try to minimize the maximum degree then we get the opposite problem. If it is unavoidable for there to be some node of large degree d, the maximum degree objective allows us to make every other vertex also of degree d, with no change in the objective function. Since the whole point of using spanners is to get a more compact representation of the graph, this is a significant issue.


© Eden Chlamtáč, Michael Dinitz, and Thomas Robinson;
licensed under Creative Commons License CC-BY



In order to remedy these drawbacks, [10] recently proposed a new objective function: the ℓ_p norm of the degree vector. Given a spanner H, we can define $\|H\|_p$ to be the ℓ_p-norm of the n-dimensional vector in which the coordinate corresponding to a node v contains the degree of v in H. Then $\|H\|_1$ is just (twice) the total number of edges, and $\|H\|_\infty$ is precisely the maximum degree. Thus the ℓ_p-norm is an interpolation between these two classical objectives. Moreover, for $1 < p < \infty$, this notion of cost has the properties that we want: it encourages low-degree nodes rather than high-degree nodes, but if high-degree nodes are unavoidable it still encourages the rest of the nodes to be as low-degree as possible. These properties, of interpolating between the average and the maximum, are why the ℓ_p-norm has appeared as a popular objective for a variety of problems, ranging from clustering (the famous k-means problem [17, 19]), to scheduling [6, 5, 1], to covering [16].

The focus of [10] was on universal guarantees rather than approximation algorithms, although they made interesting and suggestive observations about approximation algorithms. In particular, they showed that for stretch 3 and the ℓ_2-norm, the greedy algorithm performs better than would be expected from its behavior in ℓ_1 and ℓ_∞ (see Section 1.1 for more discussion). In this paper we focus on approximation algorithms, particularly for the special case pointed out by [10] – stretch 3 and the ℓ_2-norm. We precisely characterize the performance of the greedy algorithm, showing that it does even better than was claimed in [10]. We then design a new algorithm which is specialized to this setting and which, when combined with the greedy algorithm, gives the best known approximation.

1.1 Background on ℓ_p-Norm Spanners

We will be concerned with the following problem.

▶ **Definition 1.** *In the* Minimum ℓ_p-Norm t-Spanner *problem we are given an (unweighted) graph $G = (V, E)$ and are asked to find the t-spanner H of G which minimizes $\|H\|_p$.*

In this paper we will focus on Minimum ℓ_2-Norm 3-Spanner, although many of our techniques can be extended to other stretch values and ℓ_p norms.

Recall the classical greedy algorithm for finding t-spanners in undirected graphs: we add edges to the spanner as long as they do not close a cycle of length at most $t + 1$. In the weighted setting, edges are sorted by non-decreasing order of weight, and added as long as they are not already t-spanned. Here, we focus only on the unweighted setting.

In [10], the authors gave the following tight universal bounds on the ℓ_2 norm of a 3-spanner:

▶ **Theorem 2** ([10])**.** *Given an n-vertex connected unweighted undirected graph G:*

1. *There exists a 3-spanner H of G with $\|H\|_2 \leq \min\{O(n), \|G\|_2\}$, and the greedy algorithm returns such a spanner.*
2. *Any 3-spanner H of G must satisfy $\|H\|_2 \geq \max\{\sqrt{n}, \tilde{\Omega}(\sqrt{\|G\|_2})\}$.*

This immediately implies the following approximation guarantees:

▶ **Corollary 3.** *Given an n-vertex unweighted graph G, the greedy algorithm gives an $O(\sqrt{n})$-approximation for* Minimum ℓ_2-Norm 3-Spanner.

▶ **Corollary 4.** *Given an n-vertex unweighted graph G, the greedy algorithm gives an $\tilde{O}(n/\sqrt{\|G\|_2})$-approximation for* Minimum ℓ_2-Norm 3-Spanner.

Corollary 3 is the strongest approximation guarantee, as a function of n, that follows from the universal bounds in Theorem 2. However, unlike in the ℓ_1 and ℓ_∞ case, the authors of [10] showed that such tight universal upper and lower bounds do not give a tight analysis of the approximation guarantee for ℓ_2. In particular, the authors showed that the greedy algorithm actually gives a slightly better $O(n^{63/128})$-approximation.

1.2 Our Results and Techniques

We begin in Section 2 by giving a new analysis of the greedy algorithm, improving on the $O(n^{63/128})$ bound from [10].

▶ **Theorem 5.** *Given an n-vertex unweighted graph G, the greedy algorithm gives an $\tilde{O}(n^{3/7})$-approximation for* Minimum ℓ_2-Norm 3-Spanner.

We also show that this analysis is tight, i.e., there are graphs in which greedy is an $\tilde{\Omega}(n^{3/7})$-approximation. Thus we resolve the question raised by [10] on the performance of the greedy algorithm for the ℓ_2-norm and stretch 3.

Interestingly, despite the fact that greedy is purely combinatorial, we analyze it via a constant-size linear program: we show that the problem of finding the worst-case approximation ratio of the greedy algorithm reduces to solving a single LP. To do this, we decompose the input graph into a small collection of nearly-biregular subgraphs. For any such subgraph, this LP has variables describing degree and size parameters in the relevant portion of the greedy spanner and an optimal spanner (and thus has constant size). The objective function in the LP captures the ratio of the upper bound on the ℓ_2-norm of a greedy spanner to an optimal spanner for any of these subgraphs. We find an optimal solution to this LP, thus giving a tight bound on the approximation ratio.

We then go beyond previously proposed algorithms to give a new algorithm which is specialized to the case of the ℓ_2-norm and stretch 3. First we rewrite the standard flow-based LP for spanners (from [12]) to have an ℓ_p-norm objective, which leaves it as a convex (rather than linear) program which is polynomial-time solvable via Ellipsoid. We then give two new rounding algorithms, one of which is essentially the algorithm used in [9] for the ℓ_∞-norm objective, but with different parameters and a different objective, and thus a different analysis. Our second new algorithm draws independent random values for every edge and vertex in the graph, and includes an edge e if these values satisfy one of three conditions relating to the solution of the convex relaxation. Similar ideas have been used for stretch 3 and 4 with the ℓ_1-objective [12, 7, 13], but this is the first algorithm (to the best of our knowledge) which combines vertex and edge random sampling.

While it is common to trade off two different algorithms at the parameter setting where they have the same approximation ratio (e.g., as was done in the ℓ_1-objective for spanners [12, 7]), this is not what we do. Instead, the most important question is *correctness*: we carefully parameterize these algorithms so that every edge is spanned in the combined algorithm. Proving that this combined algorithm does yield a 3-spanner, and analyzing its approximation ratio, is surprisingly complex and takes up the bulk of this paper. In the end, we prove the following theorem.

▶ **Theorem 6.** *There is a polynomial-time algorithm for* Minimum ℓ_2-Norm 3-Spanner *with approximation ratio $\tilde{O}(\|G\|_2^{5/16})$.*

Finally, trading our new algorithm off with the greedy guarantee of Corollary 4, we immediately get our strongest approximation guarantee as a function of n:

▶ **Corollary 7.** *Trying both the algorithm of Theorem 6 and the greedy algorithm and returning the better of the two gives an $\tilde{O}(n^{5/13})$-approximation.*

In light of all of these upper bound results, a natural question is whether Minimum ℓ_2-Norm 3-Spanner is also hard to approximate. This is also important because strong hardness results are known for both the ℓ_1 and ℓ_∞ norms. Strong hardness of approximation for the ℓ_1-norm in directed graphs has been known since [18, 15] (where strong means

the same hardness that is known for the famous Label Cover problem, i.e., hard to approximate better than $2^{\log^{1-\epsilon} n}$ for arbitrarily small constant ϵ), and this was recently extended to undirected graphs by [11] by proving hardness for instances of Label Cover with some extra structure. For the ℓ_∞-norm objective, the techniques of [18, 15, 11] were significantly extended in both the directed and undirected settings by [9] in order to prove similar hardness bounds.

It is not hard to see that if we attempt to use the hardness results for ℓ_1 or ℓ_∞ as a "black box" then we will not be able to prove anything useful, simply because the hardness results are subpolynomial (with respect to n) and thus changing the norm loses the entire hardness. In fact, the hardness reduction used in the ℓ_1 case [11] does not seem to work for the ℓ_2-norm, since it relies on adding many low-degree nodes to amplify the hardness. On the other hand, we show that if we use the ℓ_∞ hardness reduction of [9] (with slightly different parameters), which amplifies hardness by adding a small number of high-degree nodes, we can prove a similar hardness bound.

▶ **Theorem 8.** *Unless* NP $\subseteq$ BPTIME$(2^{\text{polylog}(n)})$*, for any constant* $\epsilon > 0$ *there is no polynomial-time algorithm that can approximate* Minimum ℓ_2-Norm 3-Spanner *better than* $2^{\log^{1-\epsilon} n}$.

At a very high level, this is obtained by re-analyzing the reduction of [9] more carefully. In [9], since they cared only about the maximum degree, it was not necessary to analyze the (many) nodes with smaller degrees. Moreover, some of the key arguments in [9] are false in the context of the ℓ_2-norm: there is an argument that we can change the optimal solution to be "canonical" without affecting the ℓ_∞ norm, but in the ℓ_2-norm there is an effect. So we need to instantiate the reduction with different parameters, perform a more detailed analysis, and replace some key steps with more refined arguments. This all significantly complicates the analysis. Since our main focus is on algorithms rather than hardness, we defer the proof to Appendix A.

2 Greedy

Here, we improve over the the analysis in [10] and give a nearly tight (up to polylogarithimic factors) analysis of the greedy algorithm. We give only a high-level overview here, and defer many of the details to the full version of the paper. We show the following:

▶ **Theorem 9.** *Given an* n*-vertex unweighted graph* G*, the greedy algorithm gives an* $\tilde{O}(n^{3/7})$*-approximation for the minimum* ℓ_2*-norm 3-spanner.*

This can be seen to be tight by considering the following graph: Let T be a tree of depth 3, where the root has $n^{4/7}$ children, all level 1 nodes have $n^{2/7}$ children, and all level 2 nodes have $n^{1/7}$ children (so the number of leaves is n). Now let G be the graph created by taking T and adding an edge between the root and every leaf. Clearly, T is a 3-spanner of G with $\|T\|_2 = O(n^{4/7})$. However, the greedy algorithm could start by taking all the edges from the root to the leaves of T, right away creating a subgraph with ℓ_2-norm at least n.

Our analysis will decompose the graph into a small number of well structured subgraphs, and analyze the behavior of the greedy algorithm on each part. The condition on each subgraph is the following.

▶ **Definition 10.** *We say a graph* (L, R, E) *with vertex set* $L \cup R$ *is* nearly bi-regular *if there exist integers* d_L, d_R *such that every vertex* $u \in L$ *has* $|\Gamma(u) \cap R| \in [d_L/6, d_L]$ *and every vertex* $v \in R$ *has* $|\Gamma(v) \cap L| \in [d_R/(6 \log |R|), d_R]$.

We use standard regularization techniques to give the following decomposition (the proof is deferred to the full version).

▶ **Lemma 11.** *Given an undirected graph $G = (V, E)$ with $|V| = n$, there exist $O(\log^3 n)$ subgraphs $H_i = (L_i, R_i, E_i)$ of G such that the edge sets $\{E_i\}$ are a partition of E, and each H_i is nearly bi-regular.*

To analyze the performance of the greedy algorithm on each subgraph in the above partition, we use a specific constant size linear program, similar to the linear program used for the universal lower bound in [10], but with a different objective function: finding the worst-case ratio between the ℓ_2-norm of the greedy algorithm and an optimal spanner. The linear program assumes that an optimal set of paths of length ≤ 3 that span the edges of any biregular graph H_i has a fairly regular structure. In particular, it assumes that the union of such an extremal set of paths is a four layered graph such that the subgraph induced on every two subsequent layers is bipartite and biregular. Such a graph can be succinctly described by the cardinalities of the different layers and the degrees of the bipartite graphs connecting every two consecutive layers. A pruning argument shows that this assumption is without loss of generality, up to a polylogarithmic factor in the ℓ_2 norm.

We solve this linear program and show that the example graph described after Theorem 5 gives a feasible solution with value $n^{3/7}$, for which there is a dual solution giving the complementary bound. This linear program, its optimal solution, and its connection to the performance of the greedy algorithm are all given in the full version. Here we only mention the conclusion:

▶ **Lemma 12.** *Let H be an N-vertex nearly bi-regular graph, and let P be a graph (not necessarily a subgraph) which spans every edge in E by a path of length at most 3. Then we have $\min\{N, \|H\|_2\}/\|P\|_2 = \tilde{O}(N^{3/7})$.*

We can now prove our main theorem for this section.

Proof of Theorem 9. Let $\{H_i\}$ be the partition of G into $O(\log^3 n)$ subgraphs given in Lemma 11, and let N_i be the number of vertices in H_i. If H is a spanner returned by the greedy algorithm, we know by Theorem 2 that for each i, we have $\|H \cap H_i\|_2 = \min\{O(N_i), \|H_i\|_2\}$. Choose an i_0 that maximizes this expression. Then we have $\|H\|_2 \leq \sum_i \|H \cap H_i\|_2 = O(\log^3 n) \cdot \min\{N_{i_0}, \|H_{i_0}\|_2\}$.

On the other hand, letting P be an optimal 3-spanner of G, we know in particular that P spans the edges in H_{i_0}. And so our approximation ratio is bounded by

$$\begin{aligned}\frac{\|H\|_2}{\|P\|_2} &\leq \frac{O(\log^3 n)\min\{N_{i_0}, \|H_{i_0}\|_2\}}{\|P\|_2} \\ &= \tilde{O}(N_{i_0}^{3/7}) && \text{by Lemma 12} \\ &= \tilde{O}(n^{3/7}).\end{aligned}$$

◀

3 LP-Based Rounding

We now turn to algorithms based on rounding LP relaxations. In particular, we analyze the performance of the linear programming relaxation (though with a different objective function) suggested by [12, 9] for Minimum ℓ_1 3-Spanner and Minimum ℓ_∞ 3-Spanner, respectively. Focusing on ℓ_2, we consider the following convex program, noting that it is only the objective function which is nonlinear:

$$\min \quad \left(\sum_{v \in V} \left(\sum_{e \sim v} x_e \right)^2 \right)^{1/2}$$

$$\text{s.t.} \quad \sum_{p: u \rightsquigarrow v, |p| \leq 3} y_p = 1 \qquad \forall (u,v) \in E \tag{1}$$

$$x_e \geq \sum_{\substack{p: u \rightsquigarrow v, |p| \leq 3 \\ p \ni e}} y_p \qquad \forall (u,v), e \in E \tag{2}$$

$$x_e, y_p \geq 0 \qquad \forall e, p \tag{3}$$

While the objective function is not linear, it is convex, and so this LP can be efficiently solved by standard techniques (e.g., the Ellipsoid Method). Let us briefly see why this is a relaxation. In the intended (integral) solution, for every edge $e \in E$, x_e is an indicator for whether e appears in our spanner. Thus the objective function describes the ℓ_2 norm of our spanner. Furthermore, for every edge (u,v) we can pick a unique path p of length at most 3 between u and v in our spanner, and set $y_p = 1$, while setting $y_{p'} = 0$ for every other path p' between u and v. This is clearly a feasible solution.

3.1 Independent Edge Sampling

Given an optimum solution to the linear program, consider the following simple rounding algorithm, which slightly generalizes the rounding suggested in [9], parametrized by a constant $\alpha \in (0,1)$:

Edge-Round(α): Independently add each edge $e \in E$ to the spanner with probability x_e^α.

One part of our rounding algorithm will use this rounding for a specific value of α, though it will not necessarily return a spanner. We would like to bound the ℓ_2 norm of the subgraph returned by this algorithm. For our anlalysis of this and other rounding algorithms, we will need the following standard Chernoff bound (cf. [14], Theorem 1.1):

▶ **Theorem 13.** *Let $X = \sum_{i=1}^n X_i$, where X_i are independently distributed in [0,1]. Then for all $t > 2e\mathbb{E}[X]$, we have* $\text{Prob}[X > t] \leq 2^{-t}$.

We have the following bound on the ℓ_p-norm of the subgraph returned by Algorithm Edge-Round:

▶ **Lemma 14.** *Let H be the output of Edge-Round(α). Then with probability at least $1 - 2^{\log n - \log^2 n}$, we have $\|H\|_2 \leq \log^2(n) \|G\|_2^{1-\alpha} \text{LP}^\alpha$.*

Proof. First note that for every vertex $v \in V$ the expected degree of v in H is

$$\mathbb{E}[d_H(v)] = \sum_{e \sim v} x_e^\alpha \geq 1,$$

where the inequality follows since the x_e's support a flow of 1 from v to any neighbor. Thus, by Theorem 13, and taking a union bound over all vertices, we have that with probability at least $1 - n2^{-\log^2 n}$ all vertices $v \in V$ satisfy

$$\begin{aligned} \mathbb{E}[d_H(v)] &\leq \log^2 n \cdot \sum_{e \sim v} x_e^\alpha \\ &\leq \log^2 n \cdot d_G(v)^{1-\alpha} \left(\sum_{e \sim v} x_e \right)^\alpha. \qquad \text{(by Hölder's inequality)} \end{aligned}$$

Thus if we define vectors f, g as $f_v = d_G(v)^{2(1-\alpha)}$ and $g_v = (d_{\text{LP}}(v))^{2\alpha}$, then we get

$$\begin{aligned}
\|H\|_2^2 &\leq \log^4 n \sum_{v \in V} d_G(v)^{2(1-\alpha)} (d_{\text{LP}})^{2\alpha} \\
&= \log^4 n \; f \cdot g \\
&\leq \log^4 n \; \|f\|_{1/(1-\alpha)} \|g\|_{1/\alpha} && \text{(by Hölder's inequality)} \\
&= \log^4 n \left(\sum_{v \in V} d_G(v)^2 \right)^{1-\alpha} \left(\sum_{v \in V} (d_{\text{LP}}(v))^2 \right)^{\alpha} \\
&= \log^4 n \|G\|_2^{2(1-\alpha)} \text{LP}^{2\alpha}
\end{aligned}$$

◀

The above lemma on its own does not give a clear approximation guarantee. However, when combined with the known lower bounds on OPT, we can get the following bound:[1]

▶ **Lemma 15.** *Let H be the output of Edge-Round(α). Then with probability at least $1 - 2^{\log n - \log^2 n}$, we have $\|H\|_2 = \tilde{O}\left(\|G\|_2^{(1-\alpha)/2}\right) \cdot \text{OPT}$.*

Proof. With the stated probability, by Lemma 14 we have

$$\begin{aligned}
\|H\|_2 &\leq \log^2 n \|G\|_2^{1-\alpha} \text{LP}^{\alpha} \leq \log^2 n \|G\|_2^{1-\alpha} \text{OPT}^{\alpha} = \log^2 n \left(\frac{\|G\|_2}{\text{OPT}} \right)^{1-\alpha} \cdot \text{OPT} \\
&\leq \log^2 n \left(\frac{\|G\|_2}{\tilde{\Omega}(\sqrt{\|G\|_2})} \right)^{1-\alpha} \cdot \text{OPT},
\end{aligned}$$

where the final inequality is from the lower bound in Theorem 2, which proves the lemma. ◀

3.2 A New Rounding Algorithm

We now present a new rounding algorithm for the same linear programming relaxation, which we have designed specifically for the ℓ_2-norm, and which gives our best approximation guarantee (when traded off with the greedy algorithm).

In fact, we will round our LP solution by trying two different algorithms, and returning the union of the edge sets returned by the two algorithms. We will show that every edge will be spanned by at least one of the two algorithms with high probability. Our first algorithm is simply Edge-Round(3/7).

Algorithm 1 Edge-Round(3/7).

Independently add each edge $e \in E$ to the spanner with probability $x_e^{3/7}$.

Lemma 15 directly implies that with high probability this algorithm returns a subgraph with ℓ_2 norm at most $\text{OPT} \cdot \tilde{O}(\|G\|_2^{2/7})$, which is even better than our final guarantee (see Lemma 17). However, it is not guaranteed to return a valid spanner.

Our second algorithm takes a different approach. We balance the need in our objective function for both few edges overall and low degrees for individual vertices by simultaneously limiting which vertices can buy edges and what edges they can buy.

[1] In [9], it was shown that this algorithm gives a 3-spanner for $\alpha = 1/3$, which already gives an $\tilde{O}(\|G\|_2^{1/3})$-approximation via Lemma 15. However, this is weaker than our final $\tilde{O}(\|G\|_2^{5/16})$ guarantee.

Algorithm 2 Edge/Vertex Sampling.

- For every vertex $v \in V$, and for every edge $e \in E$, independently sample uniformly random variables $z_v^- \in_R [0,1]$, $z_v^+ \in_R [0,1]$, and $z_e \in_R [0,1]$.
- For every edge $e = (u,v) \in E$, add e to the spanner if at least one of the following three conditions holds:
  1. $z_e \le x_e^{1/4}$ and $z_v^- \le x_e^{1/4}$.
  2. $z_u^- \le x_e^{1/4}$ and $z_v^+ \le x_e^{1/4}$.
  3. $z_u^+ \le x_e^{1/4}$ and $z_e \le x_e^{1/4}$.

▶ Remark 16. The algorithm is formulated for directed graphs. If the graph is undirected, run the algorithm on the directed graph where every original edge is considered with both possible orientations.

Both showing that these algorithms give a good approximation, and showing that together they give a valid 3-spanner, requires a technically involved argument. We separate these two arguments in the next two subsections.

In Section 3.2.2, we will show that every edge has a probability of $\Omega(1/\text{polylog}n)$ of being spanned by at least one of the two algorithms. Thus, the complete algorithm will be

- For some constant $c > 0$, run both Algorithm 1 and Algorithm 2 $O(\log^c n)$ times, and output the union of all the edges chosen by either algorithm over the various iterations.

Thus, for an approximation guarantee of $\tilde{O}(f)$, it suffices to show that the probability that either algorithm returns a subgraph with ℓ_2 norm greater than $\tilde{O}(\text{OPT} \cdot f)$ is at most $O(1/\log^c n)$ for some sufficiently large constant $c > 0$. This approximation guarantee (for $f = \|G\|_2^{5/16}$) is given in Section 3.2.1.

3.2.1 Approximation guarantee

As mentioned earlier, Lemma 15 implies that Algorithm 1 returns a subgraph with ℓ_2 norm at most $\text{OPT} \cdot \tilde{O}(\|G\|_2^{2/7})$ with probability at least $1 - 2^{-(1-o(1))\log^2 n}$. This is in fact better than our final approximation guarantee, so we will focus now on Algorithm 2.

We give the following upper bound on the ℓ_2 norm of the subgraph given by Algorithm 2.

▶ **Lemma 17.** *For any $b = b(n) > 1$, Algorithm 2 outputs a graph with ℓ_2 norm at most $\text{OPT} \cdot \tilde{O}(b^{1/2}\|G\|_2^{5/16})$ with probability at least $1 - \exp(-\Omega(\log^2 n)) - 1/b$.*

Proof. We will bound the contribution to the ℓ_2 norm of every kind of edge added by the algorithm. In particular, we define the three corresponding edge sets

$$E_1 = \left\{(u,v) \in E \mid z_{(u,v)} \le x_{(u,v)}^{1/4},\ z_v^- \le x_{(u,v)}^{1/4}\right\}$$
$$E_2 = \left\{(u,v) \in E \mid z_u^- \le x_{(u,v)}^{1/4},\ z_v^+ \le x_{(u,v)}^{1/4}\right\}$$
$$E_3 = \left\{(u,v) \in E \mid z_u^+ \le x_{(u,v)}^{1/4},\ z_{(u,v)} \le x_{(u,v)}^{1/4}\right\}$$

Now consider the various degrees defined by these edge sets:

$$d_1(u) = |\{v \in V \mid (u,v) \in E_1\}| \qquad d_2(v) = |\{u \in V \mid (u,v) \in E_1\}|$$
$$d_3(u) = |\{v \in V \mid (u,v) \in E_2\}| \qquad d_4(v) = |\{u \in V \mid (u,v) \in E_2\}|$$
$$d_5(u) = |\{v \in V \mid (u,v) \in E_3\}| \qquad d_6(v) = |\{u \in V \mid (u,v) \in E_3\}|$$

To bound the ℓ_2 norm of the subgraph returned by the algorithm, we bound each of $\sum_{u\in V}(d_i(u))^2$ seperately for each $i \in [6]$. However, we only analyze the contribution for $i = 1$ and $i = 3$. The analysis for $i = 6$ is identical to the analysis for $i = 1$, and the analysis for $i \in \{2, 4, 5\}$ is essentially identical to the analysis for $i = 3$.

Let us start by analyzing $\sum_{u\in V}(d_1(u))^2$. Note that for every $u \in V$, $d_1(u)$ is a sum of independent Bernoulli random variables with success probabilities

$$\text{Prob}[z_{(u,v)} \le x_{(u,v)}^{1/4}] \cdot \text{Prob}[z_v^- \le x_{(u,v)}^{1/4}] = x_{(u,v)}^{1/2}.$$

Thus, individual degrees behave exactly as in Edge-Round(1/2). Therefore, the proof of Lemma 15 (which did not use any property of the correlation between different degrees) shows that with high probability the total contribution to the ℓ_2 norm from these degrees is at most $\text{OPT} \cdot \tilde{O}(\|G\|_2^{1/4})$ (which is even smaller than our claim).

Now let us analyze the contribution from the d_3 degrees. First, for every vertex $u \in V$, let us define

$$\hat{\Gamma}(u) := \left\{ v \in V \mid (u, v) \in E,\ z_v^+ \le x_{(u,v)}^{1/4} \right\}.$$

Note, as before, that $|\hat{\Gamma}(u)|$ is a sum of independent Bernoulli random variables with probabilities $x_{(u,v)}^{1/4}$, and so with high probability, using Hölder's inequality, we have

$$|\hat{\Gamma}(u)| \le \log^2 n \cdot \sum_{v:(u,v)\in E} x_{u,v}^{1/4} \le \log^2 n \cdot d_G(u)^{3/4} d_{\text{LP}}(u)^{1/4}.$$

We will also need to bound (in expectation and with high probability) the following expression:

$$\begin{aligned}\mathbb{E}\left[\sum_{v\in\hat{\Gamma}(u)} x_{(u,v)}^{1/4}\right] &= \sum_{v:(u,v)\in E} x_{(u,v)}^{1/2} \\ &\le d_G(u)^{1/2} d_{\text{LP}}(u)^{1/2} \qquad \text{(by Cauchy-Schwarz).}\end{aligned}$$

This is not a sum of Bernoulli random variables, but it is a sum of independent random variables distributed in $[0, 1]$, where the expectation of the sum is at least 1, so we can use Theorem 13 and get that with probability at least $1 - 2^{-\log^2 n}$ we have

$$\sum_{v\in\hat{\Gamma}(u)} x_{(u,v)}^{1/4} \le \log^2 n \cdot d_G(u)^{1/2} d_{\text{LP}}(u)^{1/2}.$$

Suppose we have sampled the z^+ variables, so the sets $\hat{\Gamma}(u)$ are fixed and the two bounds on $\hat{\Gamma}(u)$ above hold for all $u \in V$. Note that this high probability event is completely independent of the z^- variables. Conditioned on this event, for every $u \in V$, $(d_3(u))^2$ is a random variable distributed in $[0, |\hat{\Gamma}(u)|^2]$ that depends only on z_u^-. The expected contribution from a single vertex $u \in V$ is

$$\begin{aligned}\mathbb{E}[(d_3(u))^2] &= \sum_{v_1,v_2\in\hat{\Gamma}(u)} \text{Prob}[z_u^- \le x_{(u,v_1)}^{1/4},\ z_u^- \le x_{(u,v_2)}^{1/4}] \\ &\le \sum_{v_1\in\hat{\Gamma}(u)} \sum_{v_2\in\hat{\Gamma}(u)} x_{(u,v_2)}^{1/4} \\ &= |\hat{\Gamma}(u)| \cdot \sum_{v\in\hat{\Gamma}(u)} x_{(u,v)}^{1/4} \\ &\le \log^4 n \cdot d_G(u)^{5/4} d_{\text{LP}}(u)^{3/4} \qquad \text{by our bounds on } \hat{\Gamma}(u)\end{aligned}$$

Thus, by Markov's inequality, with probability at least $1 - 1/b$ we have

$$\begin{aligned}
\sum_{u \in V} \mathbb{E}[d_3(u)^2] &\leq b \log^4 n \cdot \sum_{u \in V} d_G(u)^{5/4} d_{\mathrm{LP}}^{3/4} \\
&\leq b \log^4 n \|(d_G(u)^{5/4})_{u \in V}\|_{8/5} \|d_{\mathrm{LP}}(u)^{3/4}\|_{8/3} \qquad \text{by Hölder's inequality} \\
&= b \log^4 n \left(\sum_{u \in V} d_G(u)^2 \right)^{5/8} \left(\sum_{u \in V} d_{\mathrm{LP}}(u)^2 \right)^{3/8} \\
&= b \log^4 n \cdot \mathrm{LP}^{3/4} \|G\|_2^{5/4} \\
&\leq b \log^4 n \cdot \mathrm{OPT}^{3/4} \|G\|_2^{5/4} \\
&\leq b \ \mathrm{polylog}(n) \cdot \mathrm{OPT}^{3/4} \cdot \frac{\mathrm{OPT}^{5/4}}{\|G\|_2^{5/8}} \cdot \|G\|_2^{5/4} \qquad \text{by Theorem 2} \\
&= \tilde{O}(b \|G\|_2^{5/8}) \cdot \mathrm{OPT}^2
\end{aligned}$$

Thus the contribution of the d_3 degrees to the ℓ_2 norm is at most $\tilde{O}(\sqrt{b}\|G\|_2^{5/16}) \cdot \mathrm{OPT}$, as claimed. ◀

3.2.2 Correctness

We will use the following regularization lemma (simplified form of Lemma 2.6 in [9]):

▶ **Lemma 18.** *There exists a constant $C > 0$ such that for any vertices $u, v \in V$ and set P of paths from u to v of length at most 3 such that $\sum_{p \in P} y_p = 1$, there exists a subset $P' \subseteq P$ satisfying the following conditions.*

- *For some $1 \leq k \leq 3$, all paths in P' have length k.*
- *There exists some $y_0 > 0$ such that every path $p \in P'$ has weight $y_p \in [y_0, 2y_0]$. Furthermore, $1 \geq y_0|P'| \geq 1/(\log^C n)$.*
- *If $k = 3$, then all the paths in P' are tuples in $E_1 \times E_2 \times E_3$ for some pairwise disjoint collection of edge sets $E_1, E_2, E_3 \subset E$.*
- *If $k = 3$, then there exist positive integers d_L, d_R such that:*
  - *For every edge $e_1 \in E_1$, the number of paths in P' which include e_1 is in the range $[d_L, d_L \log^C n]$. Note that this gives $|E_1| \leq |P'|/d_L \leq 1/(d_L y_0)$.*
  - *Every edge $e_2 \in E_2$ participates in exactly one path in P'.*
  - *For every edge $e_3 \in E_3$, the number of paths in P' which include e_3 is in the range $[d_R, d_R \log^C n]$. Note that this gives $|E_3| \leq |P'|/d_R \leq 1/(d_R y_0)$.*

Note that given a solution to our LP relaxation, for every edge $(u, v) \in E$ there does exist such a set of paths P, and so there exists a set of paths P' of length k as in the lemma. If $k = 1$, this is just the edge (u, v) which then has LP value $y_{(u,v)} \geq 1/(\log^C n)$, and will be added by Algorithm 1 w.p. $\tilde{\Omega}(1)$. It is also easy to see that Algorithm 1 will span (u, v) if $k = 2$. Indeed, we have:

▶ **Lemma 19.** *Let P' be a set of paths of length k for an edge $(u, v) \in E$ as in Lemma 18. Then if $k = 2$, then w.p. $\tilde{\Omega}(1)$ at least one of these paths will be added by Algorithm 1.*

Proof. Since the paths in P' are of length 2, they are also edge disjoint. By the capacity constraints, every edge e in these paths must have LP value $x_e \geq y_0$. Thus, the probability that at least one path in P' will be added by Algorithm 1 is bounded by

$$\begin{aligned}
\text{Prob[Some path in } P' \text{ is added]} &= 1 - \prod_{p \in P'} \text{Prob[P' is not added]} \\
&= 1 - \prod_{p=(e_1,e_2)\in P'} (1 - \text{Prob}[e_1 \text{ is added}]\text{Prob}[e_2 \text{ is added}]) \\
&\geq 1 - \prod_{p \in P'} \left(1 - \left(y_0^{3/7}\right)^2\right) \\
&= 1 - \left(1 - y_0^{6/7}\right)^{|P'|} \\
&\geq 1 - \exp\left(-y_0^{6/7}|P'|\right) \\
&\geq 1 - \exp\left(1/\log^C n\right) = \tilde{\Omega}(1),
\end{aligned}$$

where the last inequality follows from the fact that $y_0^{6/7}|P'| \geq y_0|P'| \geq 1/\log^C n$. ◀

Thus it remains to deal with edges for which P' is a set of paths of length 3. We will show that every such edge is spanned with probability $\Omega(1)$ by either Algorithm 1 or Algorithm 2, depending on the parameters d_L, d_R from Lemma 18. In both cases, we show that the relevant algorithm adds a large number of paths from P' in expectation, and that outside some easy special cases, the number of paths added is (at least mildly) concentrated around the expectation by Chebyshev's inequality. The following lemma describes the correctness property of Algorithm 1.

▶ **Lemma 20.** *Let $(u, v) \in E$ be an edge and P' be a set of paths of length 3 as in Lemma 18 with corresponding parameters y_0, d_L, d_R. Then if $\max\{d_L, d_R\} \geq y_0^{-2/3}/\log^{C'} n$ for some constant $C' > 0$, Algorithm 1 will add at least one path in P' with probability $\tilde{\Omega}(1)$.*

Proof. First, consider the case where $d_L = \tilde{\Omega}(1/y_0)$. In this case, every edge $e_1 = (u, u') \in E_1$ that participates in any path in P' has LP value $x_{e_1} = \tilde{\Omega}(1)$. Thus, such an edge is added by Algorithm 1 w.p. $\tilde{\Omega}(1)$. Moreover, there are $\tilde{\Omega}(1/y_0)$ paths of length 2 from u' to v with LP value at least y_0 (suffixes of paths in P' starting with e_1), and by Lemma 19, w.p. $\tilde{\Omega}(1)$ Algorithm 1 will add at least one of these, and this event is independent of e_1 being added. Thus, in this case the lemma follows. The lemma follows similarly when $d_R = \tilde{\Omega}(1/y_0)$.

Assume therefore that for some arbitrarily large constant $C'' > 0$ we have

$$d_L, d_R \leq 1/(y_0 \log^{C''} n). \tag{4}$$

Now assume w.l.o.g. that $d_L > d_R$. Thus, by our assumption, we have

$$d_L \geq y_0^{-2/3}/\log^{C'} n. \tag{5}$$

Consider the case where $(d_L \leq) d_L d_R \leq y_0^{-2/3} \log^{C''} n$, which in particular implies $d_R \leq \log^{C''+C'} n$. In this case define a new set of paths $P'' \subseteq P'$ by taking for every edge $e_3 \in E_3$ a single path in P' containing e_3. Note that $|P''| \geq |P'|/(d_R \log^C n)$. Since $|E_1| \leq |P'|/d_L$ and every edge in E_1 participates in at most $d_L \log^C n$ paths in P', this implies that at least $|P'|/(2d_L d_R \log^{2C} n)$ edges in E_1 each participate in at least $d_L/(2d_R \log^C n)$ paths in $|P''|$. Let $E_1' \subseteq E_1$ be this set of edges. So we have

$$|E_1'| \geq \frac{|P'|}{2d_L d_R \log^{2C} n} \geq \frac{1}{2d_L d_R y_0 \log^{3C} n} \geq \frac{1}{2y_0^{1/3} \log^{3C+C''} n}.$$

Now, for every $e_1 \in E_1'$, denote by $P''(e_1)$ the set of paths in P'' containing e_1. By definition of E_1', for every such e_1 we have $|P''(e_1)| \geq d_L/(2d_R \log^C n)$. Note that by capacity constraints we have $x_{e_1} \geq d_L y_0 \geq y_0^{1/3}/\log^{C'} n$. Thus, as in the proof of Lemma 19, the probability that at least one path in $P''(e_1)$ is added is bounded by

$$\begin{aligned}
&\text{Prob[Some path in } P''(e_1) \text{ is added]} \\
&\quad = \text{Prob}[e_1 \text{ is added}] \cdot \left(1 - \prod_{p \in P''(e_1)} (1 - \text{Prob}[p \setminus e_1 \text{ is added}])\right) \\
&\quad \geq (d_L y_0)^{3/7} \left(1 - (1 - y_0^{6/7})^{|P''(e_1)|}\right) \\
&\quad \geq y_0^{1/7} \log^{-3C'/7} n \left(1 - (1 - y_0^{6/7})^{d_L/(2d_R \log^C n)}\right) \\
&\quad \geq y_0^{1/7} \log^{-3C'/7} n \left(1 - (1 - y_0^{6/7})^{y_0^{-2/3}/(2\log^{C+C''-C'} n)}\right) \\
&\quad \geq y_0^{1/7} \log^{-3C'/7} n \left(1 - \exp(-y_0^{4/21}/(2\log^{C+C''-C'} n))\right) \\
&\quad = \tilde{\Omega}(y_0^{1/3})
\end{aligned}$$

By definition of P'', the paths in $P''(e_1)$ are completely edge disjoint from the paths in $P''(e_1')$ for any $e_1, e_1' \in E_1'$. Thus, there is a probability of $\tilde{\Omega}(y_0^{1/3})$ that at least one path in $P''(e_1)$ is added, and these are independent events for the different edges $e_1 \in E_1'$. By our bound on $|E_1'|$, there are $\tilde{\Omega}(y_0^{-1/3})$ such edges, and so the probability that at least one of them will contribute a path in P'' to the spanner is at least $\tilde{\Omega}(1)$. This concludes our analysis of the case where $d_L d_R \leq y_0^{-2/3} \log^{C''} n$. From this point on, we assume that

$$d_L d_R \geq y_0^{-2/3} \log^{C''} n \tag{6}$$

Now define new LP values for $E_1 \cup E_2 \cup E_3$ as follows:

$$x_e' = \begin{cases} d_L y_0 & \text{if } e \in E_1 \\ y_0 & \text{if } e \in E_2 \\ d_R y_0 & \text{if } e \in E_3. \end{cases}$$

Then by capacity constraints, for every edge $e \in E_1 \cup E_2 \cup E_3$, we have $x_e \geq x_e'$. Consider an algorithm that adds edges $e \in E_1 \cup E_2 \cup E_3$ independently with probability $(x_e')^{3/7}$ instead of $x_e^{3/7}$. Clearly, the probability that Algorithm 1 adds at least one path from P' is at least the probability that the algorithm with modified LP values does so. Let us now analyze the probability that the modified algorithm adds at least one path from P', and denote by Y the number of paths from P' added by the modified algorithm. We start by analyzing the expectation of Y. By definition of the modified algorithm, every path $p \in P'$ is added with probability

$$\text{Prob}[p \text{ is added}] = (d_L y_0)^{3/7} y_0^{3/7} (d_R y_0)^{3/7} = (d_L d_R)^{3/7} y_0^{9/7}.$$

Since $|P'| = y_0^{-1}/(\log^c n)$ for some $c \in [0, C]$, for an appropriate choice of C'', the expected number of paths added by the modified algorithm satisfies

$$\begin{aligned}
\mathbb{E}[Y] &= |P'|(d_L d_R)^{3/7} y_0^{9/7} \\
&= \log^{-c} n \cdot (d_L d_R)^{3/7} y_0^{2/7} && (7) \\
&\geq \log^{3C''/7 - c} n && \text{by (6)} \\
&\geq \log^{1+c/2+3C/2} n,
\end{aligned}$$

where the last inequality follows if we choose, say,

$$C'' = 7/3 + 7c/2 + 7C/2.$$

Thus, the expected number of paths in P' added is (relatively) large. However, since the paths in P' are not disjoint, this does not guarantee that at least one path will be added with probability $\tilde{\Omega}(1)$. To show this, we need to show concentration. As in [9], we use Chebyshev's inequality, which guarantees that $\text{Prob}[Y = 0] < \frac{1}{2}$ as long as

$$\text{Var}[Y] = \mathbb{E}[Y^2] - (\mathbb{E}[Y])^2 < \frac{1}{8}(\mathbb{E}[Y])^2. \tag{8}$$

To show that (8) holds, consider the contribution of edge-disjoint versus non-edge-disjoint paths:

$$\begin{aligned}
\mathbb{E}[Y^2] &= \sum_{p_1,p_2 \in P'} \text{Prob}[p_1 \text{ and } p_2 \text{ are added}] \\
&= \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 \neq \emptyset}} \text{Prob}[p_1 \text{ and } p_2 \text{ are added}] + \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 = \emptyset}} \text{Prob}[p_1 \text{ is added}]\text{Prob}[p_2 \text{ is added}] \\
&\leq \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 \neq \emptyset}} \text{Prob}[p_1 \text{ and } p_2 \text{ are added}] + \sum_{p_1,p_2 \in P'} \text{Prob}[p_1 \text{ is added}]\text{Prob}[p_2 \text{ is added}] \\
&= \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 \neq \emptyset}} \text{Prob}[p_1 \text{ and } p_2 \text{ are added}] + \mathbb{E}[Y]^2
\end{aligned}$$

Thus, to show (8), it suffices to bound the contribution from pairs of non-edge-disjoint paths. These fall into three categories: pairs of identical paths, pairs sharing only the first edge (in E_1), and pairs sharing only the third edge (in E_3). The contribution from paths in the first category is at most

$$\sum_{p_1 = p_2 \in P'} \text{Prob}[p_1 \text{ and } p_2 \text{ are added}] = \mathbb{E}[Y] = o(\mathbb{E}[Y]^2). \qquad (\text{since } \mathbb{E}[Y] > \log n)$$

The analysis for the second and third categories is identical, so let us focus only on pairs of paths sharing only the first edge. These pairs contribute

$$\begin{aligned}
\sum_{e_1 \in E_1} \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 = \{e_1\}}} \text{Prob}[p_1 \text{ and } p_2 \text{ are added}] &= \sum_{e_1 \in E_1} \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 = \{e_1\}}} (d_L y_0)^{3/7} y_0^{6/7} (d_R y_0)^{6/7} \\
&= \sum_{e_1 \in E_1} \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 = \{e_1\}}} d_L^{3/7} d_R^{6/7} y_0^{15/7} \\
&\leq |E_1| (d_L \log^C n)^2 \cdot d_L^{3/7} d_R^{6/7} y_0^{15/7} \\
&\leq \frac{1}{d_L y_0} \cdot (\log^{2C} n) d_L^{17/7} d_R^{6/7} y_0^{15/7} \\
&= (\log^{2C} n) d_L^{10/7} d_R^{6/7} y_0^{8/7} \\
&\leq (\log^{2C - 4C''/7} n) d_L^{6/7} d_R^{6/7} y_0^{4/7} && \text{(by (4))} \\
&= \log^{2C + 2c - 4C''/7} n \cdot \mathbb{E}[Y]^2 && \text{(by (7))} \\
&= \log^{-4/3} n \cdot \mathbb{E}[Y]^2
\end{aligned}$$

Thus, all three categories contribute at most $o(\mathbb{E}[Y]^2)$, and this concludes the proof. ◀

Finally, we examine edges that will be spanned by Algorithm 2. Note that the trade-off between the two algorithms has nothing to do with the approximation guarantee. In fact, at the parameter threshold between the two algorithms (where $\max\{d_L, d_R\} \approx y_0^{-2/3}$), both algorithms either give or could be easily modified to give a better approximation than $\|G\|_2^{5/16}$. The reason for trading off the two algorithms is that beyond the threshold, Algorithm 2 will still give a large number of spanning paths for each edge in expectation, but in reality will only span such an edge with very low probability (in which event it will span it with many paths). The following lemma gives the parameter regime for edges spanned by Algorithm 2.

▶ **Lemma 21.** *Let $(u, v) \in E$ be an edge and P' be a set of paths of length 3 as in Lemma 18 with corresponding parameters y_0, d_L, d_R. Then there exists a constant $C' > 0$ such that if $d_L, d_R \leq y_0^{-2/3}/\log^{C'} n$, Algorithm 2 will add at least one path in P' with probability $\tilde{\Omega}(1)$.*

Proof. First consider the case where $d_L d_R \leq \log^{C''} n$ for some constant $C'' > 0$. This is an easy special case, but also the tight case of our entire analysis. In this case, define a new set of paths $P'' \subseteq P'$ by taking for every edge $e_3 \in E_3$ a single path in P' containing e_3, and then out of these, choosing for every edge $e_1 \in E_1$ used in these paths a single path containing e_1. Note that $|P''| \geq |P'|/\log^{2C+C''} n$, and all the paths in P'' are edge disjoint. For any path $p = (u, u', v', v) \in P''$, we can bound the probability that p is added by Algorithm 1 by

$$\begin{aligned}\text{Prob}[p \text{ is added}] &\geq \text{Prob}[z_{(u,u')} \leq x_{(u,u')}^{1/4}] \cdot \text{Prob}[z_{u'}^- \leq x_{(u,u')}^{1/4}, x_{(u',v')}^{1/4}] \\ &\quad \cdot \text{Prob}[z_{v'}^+ \leq x_{(u',v')}^{1/4}, x_{(v',v)}^{1/4}] \cdot \text{Prob}[z_{(v,v')} \leq x_{(v,v')}^{1/4}] \\ &\geq \text{Prob}[z_{(u,u')} \leq y_0^{1/4}] \cdot \text{Prob}[z_{u'}^- \leq y_0^{1/4}] \cdot \text{Prob}[z_{v'}^+ \leq y_0^{1/4}] \cdot \text{Prob}[z_{(v,v')} \leq y_0^{1/4}] \\ &= y_0.\end{aligned}$$

Since the paths in P'' are completely edge-disjoint, and share no interior vertices, the above events are independent for the various paths, and so the probability that at least one path in P'' is added is at least

$$\begin{aligned}1 - (1 - y_0)^{|P''|} \geq 1 - \exp(-y_0|P''|) &\geq 1 - \exp(-y_0|P'|/\log^{2C+C''} n) \\ &\geq 1 - \exp(-1/\log^{3C+C''} n) = \tilde{\Omega}(1).\end{aligned}$$

Now consider the remaining case. That is, assume from now on that

$$d_L d_R \geq \log^{C''} n. \tag{9}$$

As in the proof of Lemma 20, consider the result of running Algorithm 2 with LP values $\{x'_e\}$ as defined in that proof. As before, modifying the LP values in such a way can only decrease the probability that one of the paths in P' will be chosen. Let Y be the following set of paths, added by Algorithm 2 using the modified LP values:

$$\begin{aligned}Y = \Big\{(u, u', v', v) \in P' \mid &\left(z_{(u,u')} \leq (x'_{(u,u')})^{1/4}\right) \wedge \left(z_{u'}^- \leq (x'_{(u,u')})^{1/4}, (x'_{(u',v')})^{1/4}\right) \\ &\wedge \left(z_{v'}^+ \leq (x'_{(u',v')})^{1/4}, (x'_{(v',v)})^{1/4}\right) \wedge \left(z_{(v,v')} \leq (x'_{(v,v')})^{1/4}\right)\Big\}\end{aligned}$$

Since $d_L, d_R \geq 1$, the probability that a single path $p \in P'$ is added to Y is exactly

$$(d_L y_0)^{1/4} \cdot \min\{(d_L y_0)^{1/4}, y_0^{1/4}\} \cdot \min\{y_0^{1/4}, (d_R y_0)^{1/4}\} \cdot (d_R y_0)^{1/4} = (d_L d_R)^{1/4} y_0.$$

Since $|P'| = y_0^{-1}/(\log^c n)$ for some $c \in [0, C]$, for an appropriate choice of C'', the expected number of paths added by the modified algorithm satisfies

$$\begin{aligned}\mathbb{E}[Y] &= |P'|(d_L d_R)^{1/4} y_0 = \log^{-c} n \cdot (d_L d_R)^{1/4} && (10)\\ &\geq \log^{C''-c} n && \text{by (9)}\\ &\geq \log n,\end{aligned}$$

where the last inequality follows if we choose $C'' = 1 + c$.

As before, it is not enough to show that the expected number of paths is large, we also need to show concentration. As in the proof of Lemma 20, it suffices to show that

$$\sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 \neq \emptyset}} \text{Prob}[p_1, p_2 \subseteq Y] = o\left((\mathbb{E}[Y])^2\right).$$

These pairs of non-edge-disjoint paths fall into three categories: pairs of identical paths, pairs sharing only the first edge (in E_1), and pairs sharing only the third edge (in E_3). As before, the contribution from identical paths is $\mathbb{E}[Y] = o((\mathbb{E}[Y])^2)$ (where the final bound follows since $\mathbb{E}[Y] \geq \log n$). Since the analysis for the second and third categories is essentially the same, we focus on pairs of paths sharing only the first edge. These pairs contribute

$$\begin{aligned}\sum_{e_1 \in E_1} \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 = \{e_1\}}} \text{Prob}[p_1, p_2 \subseteq Y] &= \sum_{e_1 \in E_1} \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 = \{e_1\}}} (d_L y_0)^{1/4} y_0^{1/4} y_0^{1/2} (d_R y_0)^{1/2}\\ &= \sum_{e_1 \in E_1} \sum_{\substack{p_1,p_2 \in P' \\ p_1 \cap p_2 = \{e_1\}}} d_L^{1/4} d_R^{1/2} y_0^{3/2}\\ &\leq \frac{1}{d_L y_0} \cdot (d_L \log^C n)^2 \cdot d_L^{1/4} d_R^{1/2} y_0^{3/2}\\ &= (\log^{2C} n) d_L^{5/4} d_R^{1/2} y_0^{1/2}\\ &\leq (\log^{2C-3C'/4} n) d_L^{1/2} d_R^{1/2} \qquad (\text{since } d_L \leq y_0^{-2/3}/\log^{C'} n)\\ &= \log^{2C+2c-3C'/4} n \cdot \mathbb{E}[Y]^2 \qquad (\text{by (10)})\\ &\leq \log^{-3/4} n \cdot \mathbb{E}[Y]^2,\end{aligned}$$

where the final bound follows if we choose $C' \geq 1 + 8C/3 + 8c/3$. Thus, all three categories contribute at most $o(\mathbb{E}[Y]^2)$, and this concludes the proof. ◀

4 Generalizations and Open Questions

While we provided approximation and hardness bounds for Minimum ℓ_2-Norm 3-Spanner, the true approximability still remains open. Perhaps more interesting, though, is the question of the more general Minimum ℓ_p-Norm k-Spanner problem. Some of our techniques easily extend to this more general setting, but some do not. The linear-programming based framework we use to analyze the greedy algorithm should basically work for other values of p and k, but the details become more complicated.

Recall that our strongest approximation algorithm (from Section 3) is a careful tradeoff between greedy, independent edge sampling (**Edge-Round**), and a combined vertex and edge sampling (Algorithm 2). Independent edge sampling (**Edge-Round**) can also be analyzed for other values of p and k, where the right α to use depends on the value of k (indeed, this is the main technique used by [9] for $p = \infty$, and correctness for other values of k

follows directly from [9]). Our more tailored algorithm (Algorithm 2), which combines edge and vertex sampling, seems harder to generalize for larger values of k. Algorithm 2 is a generalization of the ideas used for $k = 3, 4$ in the ℓ_1 case (due to [12, 7, 13]), and it is a fascinating open question to extend these techniques to larger stretch values. For stretch $k = 3$ and other values of p, Algorithm 2 can be reanalyzed with appropriate parameters and seems to give nontrivial guarantees. In general, designing and analyzing approximation algorithms for other values of p and k remains an exciting challenge which may require new algorithmic ideas.

With respect to hardness, our results in Appendix A already include other values of p. For larger stretch values, the basic construction can be extended by including "outer paths" in the same way as has been done for many other spanner hardness results ([15, 9] in particular).

References

1 Noga Alon, Yossi Azar, Gerhard J. Woeginger, and Tal Yadid. Approximation Schemes for Scheduling. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '97, 1997.

2 Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete Comput. Geom.*, 9(1):81–100, 1993. doi:10.1007/BF02189308.

3 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *J. ACM*, 45(3):501–555, May 1998. doi:10.1145/278298.278306.

4 Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *J. ACM*, 45(1):70–122, January 1998. doi:10.1145/273865.273901.

5 Yossi Azar, Leah Epstein, Yossi Richter, and Gerhard J. Woeginger. All-Norm Approximation Algorithms. In Martti Penttonen and Erik Meineche Schmidt, editors, *Algorithm Theory — SWAT 2002*, 2002.

6 Nikhil Bansal and Kirk Pruhs. Server Scheduling in the L_p Norm: A Rising Tide Lifts All Boat. In *Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 242–250, 2003.

7 Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Approximation algorithms for spanner problems and Directed Steiner Forest. *Inf. Comput.*, 222:93–107, 2013. doi:10.1016/j.ic.2012.10.007.

8 Barun Chandra, Gautam Das, Giri Narasimhan, and José Soares. New Sparseness Results on Graph Spanners. In *Proceedings of the Eighth Annual Symposium on Computational Geometry*, SCG '92, pages 192–201, New York, NY, USA, 1992. ACM. doi:10.1145/142675.142717.

9 Eden Chlamtáč and Michael Dinitz. Lowest-Degree k-Spanner: Approximation and Hardness. *Theory of Computing*, 12(15):1–29, 2016. doi:10.4086/toc.2016.v012a015.

10 Eden Chlamtáč, Michael Dinitz, and Thomas Robinson. The Norms of Graph Spanners. In *Proceedings of the 46th International Colloquium Conference on Automata, Languages, and Programming*, ICALP '19, 2019.

11 Michael Dinitz, Guy Kortsarz, and Ran Raz. Label Cover Instances with Large Girth and the Hardness of Approximating Basic k-Spanner. *ACM Trans. Algorithms*, 12(2):25:1–25:16, December 2015. doi:10.1145/2818375.

12 Michael Dinitz and Robert Krauthgamer. Directed Spanners via Flow-based Linear Programs. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 323–332, New York, NY, USA, 2011. ACM. doi:10.1145/1993636.1993680.

13 Michael Dinitz and Zeyu Zhang. Approximating Low-stretch Spanners. In *Proceedings of the Twenty-seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 821–840, Philadelphia, PA, USA, 2016. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2884435.2884494.

14 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009. URL: http://www.cambridge.org/gb/knowledge/isbn/item2327542/.

15 Michael Elkin and David Peleg. The Hardness of Approximating Spanner Problems. *Theor. Comp. Sys.*, 41(4):691–729, December 2007. doi:10.1007/s00224-006-1266-2.

16 Daniel Golovin, Anupam Gupta, Amit Kumar, and Kanat Tangwongsan. All-Norms and All-L_p-Norms Approximation Algorithms. In *FSTTCS*, volume 2 of *LIPIcs*, pages 199–210. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2008.

17 Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. *Computational Geometry*, 28(2):89–112, 2004. Special Issue on the 18th Annual Symposium on Computational Geometry - SoCG2002. doi:10.1016/j.comgeo.2004.03.003.

18 Guy Kortsarz. On the Hardness of Approximating Spanners. *Algorithmica*, 30(3):432–450, 2001. doi:10.1007/s00453-001-0021-y.

19 S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, March 1982. doi:10.1109/TIT.1982.1056489.

20 Ran Raz. A Parallel Repetition Theorem. *SIAM J. Comput.*, 27(3):763–803, 1998. doi:10.1137/S0097539795280895.

A Hardness Results

Since the analysis is simpler in the directed setting, we follow [9] and begin with it. We will also prove hardness for the more general ℓ_p-norm version, and hardness when $p = 2$ will follow as a corollary. First, though, we give some notation and background necessary for the reduction.

A.1 Background: Min-Rep and Spanner Hardness

Our hardness bounds rely on the Min-Rep problem. In Min-Rep we are given a bipartite graph $G = (A, B, E)$ where A is partitioned into groups $A_1, A_2, \ldots, A_r$ and B is partitioned into groups $B_1, B_2, \ldots, B_r$, with the additional property that every set A_i and every set B_j has the same size (which we will call $|\Sigma|$ due to its connection to the alphabet of a 1-round 2-prover proof system). This graph and partition induces a new bipartite graph G' called the *supergraph* in which there is a vertex a_i for each group A_i and similarly a vertex b_j for each group B_j. There is an edge between a_i and b_j in G' if there is an edge in G between some node in A_i and some node in B_j. A node in G' is called a supernode, and similarly an edge in G' is called a superedge.[2]

A REP-cover is a set $C \subseteq A \cup B$ with the property that for all superedges $\{a_i, b_j\}$ there are nodes $a \in A_i \cap C$ and $b \in B_j \cap C$ where $\{a, b\} \in E$. We say that $\{a, b\}$ *covers* the superedge $\{a_i, b_j\}$. The goal is to construct a REP-cover of minimum size.

For any fixed constant $\epsilon > 0$, we say that an instance of Min-Rep is a YES instance if $OPT = 2r$ (i.e. a single node is chosen from each group) and is a NO instance if $OPT \geq 2^{\log^{1-\epsilon} n} r$. We will sometimes refer to the hardness gap (in this case $2^{\log^{1-\epsilon} n}$) as the *soundness* s, due to the connection between Min-Rep and proof systems. The following theorem is due to Kortsarz [18] (the polynomial relations between the parameters are implicit rather than explicit in his proof, but are straightforward to verify since the instances used in [18] are obtained by parallel repetition [20] applied to instances of 3SAT-5 which have a constant gap [4, 3]).

[2] Rather than G being the graph and G' being the supergraph, sometimes G' is referred to as the graph and G is called the *label-extended graph*.

▶ **Theorem 22** ([18]). *Unless* NP $\subseteq$ DTIME($2^{\text{polylog}(n)}$)*, for any constant* $\epsilon > 0$ *there is no polynomial-time algorithm that can distinguish between YES and NO instances of Min-Rep. This is true even when the graph and the supergraph are regular, and both the supergraph degree and* $|\Sigma|$ *are polynomial in the soundness.*

A.2 Directed Hardness

A.2.1 Reduction

We first consider the directed setting (note that here the "degree" in the degree vector is the sum of the in-degree and the out-degree). Suppose we are given a Min-Rep instance $\widetilde{G} = (A, B, \widetilde{E})$ with associated supergraph $G' = (U, V, E')$ from Theorem 22. For any vertex $w \in U \cup V$ we let $\Gamma(w)$ denote its group. So $\Gamma(u) \subseteq A$ for $u \in U$, and $\Gamma(v) \subseteq B$ for $v \in V$. Similarly, for $a \in A \cup B$, we let $\Gamma^{-1}(a)$ be the unique $w \in U \cup V$ such that $a \in \Gamma(w)$. We will assume without loss of generality that G' is regular with degree $d_{G'}$ and $\widetilde{G}$ is regular with degree $d_{\widetilde{G}}$. Our reduction will also use a special bipartite regular graph $H = (X, Y, E_H)$, which will simply be the directed complete bipartite graph with $|X| = |Y|$. Let d_H denote the degree of a node in H, so $d_H = |X| = |Y|$ (later when we move to the undirected setting H will not just be the complete bipartite graph). We will set all of these values to $(d_{G'} + |\Sigma| + 1)^{p/(p-1)}$ (for the undirected setting we will set d_H to this value, but $|X| = |Y|$ will be larger).

Our instance $G = (V_G, E_G)$ of Minimum ℓ_p-Norm 3-Spanner will be a combination of these three graphs. The four sets of vertices are

$$V^L_{out} = U \times X \qquad V^R_{out} = V \times Y$$
$$V^L_{in} = A \times E_H \qquad V^R_{in} = B \times E_H.$$

The actual vertex set V_G of our instance G will be $V^L_{out} \cup V^R_{out} \cup V^L_{in} \cup V^R_{in}$. Defining the edge set is a little more complex, as there are a few different types of edges. We first create *outer edges*, which are incident on outer nodes:

$$E_{out} = \{((u, x), (v, y)) : u \in U \wedge v \in V \wedge x \in X \wedge y \in Y \wedge \{u, v\} \in E' \wedge (x, y) \in E_H\}.$$

Note that if we fix x and y the corresponding outer edges form a copy of the supergraph G'. Thus these edges essentially form $|E_H|$ copies of the supergraph.

We also have *inner edges*, which correspond to $|E_H|$ copies of the Min-Rep instance (note that unlike the supergraph copies, these copies are vertex disjoint):

$$E_{in} = \{((a, e), (b, e)) : a \in A \wedge b \in B \wedge e \in E_H \wedge \{a, b\} \in \widetilde{E}\}.$$

We will now add *connection edges*, i.e., edges that connect some of the outer nodes to some of the inner nodes. Let

$$E^L_{con} = \{((u, x), (a, (x, y))) : u \in U \wedge a \in \Gamma(u) \wedge x \in X \wedge (x, y) \in E_H\}, \text{ and}$$
$$E^R_{con} = \{((b, (x, y)), (v, y)) : v \in V \wedge b \in \Gamma(v) \wedge y \in Y \wedge (x, y) \in E_H\}.$$

In other words, the outer node (u, x) (resp. (v, y)) is connected to the inner nodes in its group in each copy of $\widetilde{G}$ that corresponds to an E_H edge that involves x (resp. y).

Finally, for technical reasons we need to add *group edges* internally in each group in each copy of $\tilde{G}$: let $E^L_{group} = \{((a, e), (a', e)) : e \in E_H \wedge a, a' \in A \wedge \Gamma^{-1}(a) = \Gamma^{-1}(a')\}$, and let $E^R_{group} = \{((b, e), (b', e)) : e \in E_H \wedge b, b' \in B \wedge \Gamma^{-1}(b) = \Gamma^{-1}(b')\}$.

Our final edge set is the union of all of these, namely $E_G = E_{out} \cup E_{in} \cup E^L_{con} \cup E^R_{con} \cup E^L_{group} \cup E^R_{group}$.

A.2.2 Analysis

We first consider the YES case. We can use almost the same spanner as was used to prove the equivalent lemma in [9] (Lemma 3.3). Unfortunately, since in [9] only the maximum degree mattered, they did not need to optimize the degrees of non-extremal vertices, while we do. So we actually use a slightly sparser spanner construction.

▶ **Lemma 23.** *If $\widetilde{G}$ is a YES instance of Min-Rep, then there is a 3-spanner S of G with $\|S\|_p \leq 3d_H(|U||X| + |V||Y|)^{1/p}$*

Proof. Since $\widetilde{G}$ is a YES instance, for each $u \in U$ there is some $f(u) \in \Gamma(u)$ and for each $v \in V$ there is some $f(v) \in \Gamma(v)$ so that $\{f(u), f(v)\} \in \tilde{E}$ for all $\{u, v\} \in E'$. Our spanner S the connection edges suggested by the REP-cover: for every $u \in U$ and $x \in X$ and $(x, y) \in E_H$, it contains the connection edge $((u, x), (f(u), (x, y)))$. Similarly, for every $v \in V$ and $y \in Y$ and $(x, y) \in E_H$, it contains the connection edge $((f(v), (x, y)), (v, y))$. It also contains a star of group edges centered at the chosen node in every group: for every $u \in U$ and $e \in E_H$ and $a \in \Gamma(u)$ it includes the group edges $((f(u), e), (a, e))$ and $((a, e), (f(u), e))$, and for every $v \in V$ and $e \in E_H$ and $b \in \Gamma(v)$ it includes the group edges $((f(v), e), (b, e))$ and $((b, e), (f(v), e))$. Finally, it contains the appropriate inner edges: for every $\{u, v\} \in E'$ with $u \in U$ and $v \in V$ and every $e \in E_H$, we add the inner edge $((f(u), e), (f(v), e))$.

This is precisely the spanner from [9, Lemma 3.3] but with fewer group edges (we include stars in each group, while [9, Lemma 3.3] included all group edges). It is easy to verify that this change does not affect the correctness of the spanner: all edges in G not in S are still spanned. So we rely on [9, Lemma 3.3] for correctness.

So we just need to analyze $\|S\|_p$. To do this, we can just count the degrees in S of each type of nodes. There are $|U||X| + |V||Y|$ outer nodes, each of which has degree at most d_H in S. For the inner nodes, we divide into those that are chosen (those that are $(f(u), e)$ or $(f(v), e)$ for some u or v in $U \cup V$) and those that are not. There are at most $|E_H|(|A| + |B|)$ inner nodes which are not chosen, and in S they all have degree 2 (an incoming and outgoing group edge from the node in the same group that is chosen). There are at most $|E_H|(|U| + |V|)$ inner nodes which are chosen, each of which has degree in S of at most $|\Sigma| + d_{G'} + 1$ (the group edges, inner edges, and connection edges that it is incident with respectively). Putting all this together, we get that

$$\begin{aligned}
\|S\|_p &\leq ((|U||X| + |V||Y|) \cdot d_H^p \\
&\quad + |E_H|(|A| + |B|) \cdot 2^p + |E_H|(|U| + |V|)(|\Sigma| + d_{G'} + 1)^p)^{1/p} \\
&\leq ((|U||X| + |V||Y|) \cdot d_H^p + 2|E_H|(|U| + |V|)(|\Sigma| + d_{G'} + 1)^p)^{1/p} \\
&\leq (3(|U||X| + |V||Y|) \cdot d_H^p)^{1/p} \\
&\leq 3d_H(|U||X| + |V||Y|)^{1/p},
\end{aligned}$$

where we have used our setting of d_H and the fact that $|A| + |B| = (|U| + |V|)|\Sigma|$. ◀

Now we analyze the NO setting.

▶ **Lemma 24.** *If $\widetilde{G}$ is a NO instance of Min-Rep, then every 3-spanner S of G has $\|S\|_p \geq sd_H(|U||X| + |V||Y|)^{1/p}$.*

Proof. Suppose for the sake of contradiction that this is false. Let S be a 3-spanner of G with $\|S\|_p < sd_H(|U||X| + |V||Y|)^{1/p}$. For every outer node (u, x) in V_{out}^L and edge $(x, y) \in E_H$, let $d_{out}^{x,y}(u, x)$ be the number of outer edges in S that are incident on (u, x) and have the other

endpoint of the form (v, y) for some $v \in V$. Similarly, for every outer node (v, y) in V^R_{out} and edge $(x, y) \in E_H$, let $d^{x,y}_{out}(v, y)$ be the number of outer edges in S that are incident with (v, y) and have the other endpoint of the form (u, y). For every outer node (u, x) in V^L_{out} and edge $(x, y) \in E_H$, let $d^{x,y}_{con}(u, x)$ be the number of connection edges in S that are incident with (u, x) and have the other endpoint of the form $(a, (x, y))$ for some $a \in \Gamma(u)$. Similarly, for every outer node (v, y) in V^R_{out} and edge $(x, y) \in E_H$, let $d^{x,y}_{con}(v, y)$ be the number of connection edges in S that are incident with (v, y) and have the other endpoint of the form $(b, (x, y))$ for some $b \in \Gamma(v)$.

Now with this notation in hand, the fact that $\|S\|_p \leq sd_H(|U||X|+|V||Y|)^{1/p}$ implies that

$$\sum_{(x,y)\in E_H} \left(\sum_{u\in U} ((d^{x,y}_{out}(u,x))^p + (d^{x,y}_{con}(u,x))^p) + \sum_{v\in V} ((d^{x,y}_{out}(v,y))^p + (d^{x,y}_{con}(v,y))^p) \right)$$
$$\leq (sd_H)^p(|U||X| + |V||Y|)$$

Now a simple application of Hölder's inequality gives us the following.

$$\sum_{(x,y)\in E_H} \left(\sum_{u\in U} (d^{x,y}_{out}(u,x) + d^{x,y}_{con}(u,x)) + \sum_{v\in V} (d^{x,y}_{out}(v,y) + d^{x,y}_{con}(v,y)) \right)$$
$$\leq (sd_H)(|U||X| + |V||Y|) \qquad \text{(Hölder's inequality)}$$
$$\leq 2sd_H(|U||X|) \qquad (H \text{ and } G' \text{ are both balanced})$$
$$\leq |E_H|2s|U| \qquad (H \text{ is regular with degree } d_H)$$
$$\leq |E_H|s(|U| + |V|) \qquad (G' \text{ is balanced})$$

Thus averaging now implies that there is some $(x, y) \in E_H$ such that

$$\sum_{u\in U} (d^{x,y}_{out}(u,x) + d^{x,y}_{con}(u,x)) + \sum_{v\in V} (d^{x,y}_{out}(v,y) + d^{x,y}_{con}(v,y)) \leq s(|U| + |V|) \tag{11}$$

Fix this (x, y). We create a set $C_1(u) \subseteq \Gamma(u)$ for each $u \in U$ by adding all $a \in \Gamma(u)$ such that there is a connection edge $((u, x), (a, (x, y)))$ which contributes to $d^{x,y}_{con}(u, x)$. Similarly, we create a set $C_1(v) \subseteq \Gamma(v)$ for each $v \in V$ by adding all $b \in \Gamma(v)$ such that there is a connection edge $((b, (x, y)), (v, y))$ which contributes to $d^{x,y}_{con}(v, y)$.

Now we create similar sets for the outer edges. For each $u \in U$ we create a set $C_2(u) \subseteq \Gamma(u)$ and for each $v \in V$ we create a set $C_2(v)$ as follows. For every outer edge $((u, x), (v, y))$ in S (i.e., every outer edge which contributes to $d^{x,y}_{out}(u) + d^{x,y}_{out}(v)$), we pick an arbitrary $a \in \Gamma(u)$ and $b \in \Gamma(v)$ such that $\{a, b\} \in \widetilde{E}$ and add a to $C_2(u)$ and b to $C_2(v)$.

Let $C(u) = C_1(u) \cup C_2(u)$ for all $u \in U$, and let $C(v) = C_1(v) \cup C_2(v)$ for all $v \in V$. Let $C = (\cup_{u\in U} C(u))) \cup (\cup_{v\in V} C(v))$. Clearly by construction we know that

$$|C| \leq \sum_{u\in U} ((d^{x,y}_{out}(u,x)) + (d^{x,y}_{con}(u,x))) + \sum_{v\in V} ((d^{x,y}_{out}(v,y)) + (d^{x,y}_{con}(v,y)))$$
$$\leq s(|U| + |V|). \qquad \text{(by (11))}$$

Now we claim that C is a valid REP-cover. This will prove the lemma, since it will imply that S is not a NO instance, giving a contradiction and thus implying that no such S can exist. To see that C is a REP-cover, consider an arbitrary superedge $\{u, v\} \in E'$. It is not hard to see (and was proved in [9]) that the only way that S can span the outer edge $((u, x), (v, y))$ is to either include that edge in S or include a *canonical path* between the endpoints: a path which uses a connection edge to get to some $(a, (x, y))$, then an inner

edge to get to some $(b,(x,y))$, then a connection edge to get to (v,y). If the outer edge is included in S, then when we constructed $C_2(u)$ and $C_2(v)$ we explicitly added some $a \in \Gamma(u)$ and $b \in \Gamma(v)$ that cover $\{u,v\}$. Otherwise S spans the outer edge using a canonical path, which from the construction of $C_1(u)$ and $C_1(v)$ means that there is some $a,b \in C$ which covers $\{u,v\}$. Thus C is a REP-cover, which proves the lemma. ◀

Now we can prove our hardness bound using these lemmas.

▶ **Theorem 25.** *Unless* NP $\subseteq$ DTIME$(2^{\text{polylog}(n)})$*, for any constant* $\epsilon > 0$ *and* $p \geq 1$ *there is no polynomial-time algorithm that can approximate* Minimum ℓ_p-Norm 3-Spanner *in directed graphs better than* $2^{\left(\frac{p-1}{3p-1}\right)^{1-\epsilon}\log^{1-\epsilon}n}$.

Proof. Lemmas 23 and 24, together with Theorem 22, imply that under the complexity assumption, there is no polynomial-time algorithm with approximation ratio better than

$$\frac{sd_H(|U||X|+|V||Y|)^{1/p}}{3d_H(|U||X|+|V||Y|)^{1/p}} = \frac{s}{3}.$$

The only thing that remains is to argue about the increase in the size: the n in the value of s is really $|A|+|B|$, while our graph G is larger. But it is not too much larger: the number of vertices in G is $|V_G| = |U||X|+|V||Y|+|A||E_H|+|B||E_H| = O(n(|\Sigma|+d_{G'})^{2p/(p-1)}) \leq O(n^{1+\frac{2p}{p-1}}) = O(n^{(3p-1)/(p-1)})$. Thus the overall hardness that we obtain is

$$\frac{s}{3} = \frac{1}{3}2^{\log^{1-\epsilon}n} = \frac{1}{3}2^{\log^{1-\epsilon}(N^{(p-1)/(3p-1)})} = \frac{1}{3}2^{\left(\frac{p-1}{3p-1}\right)^{1-\epsilon}\log^{1-\epsilon}N}.$$

The extra $1/3$ factor can be absorbed by using a smaller ϵ. ◀

Our claimed hardness theorem for $p=2$, the directed version of Theorem 8, is a corollary of this theorem for $p=2$.

A.3 Undirected Hardness

We extend the directed hardness to the undirected setting in the same way that it was extended for LDkS in [9]. First, we start with a slightly different Min-Rep instance with some useful extra properties (from [11] instead of from [18], and with some extra analysis from [9]). Then we combine it with a graph H which is the finite projective plane of degree $d_H = (d_{G'}+|\Sigma|+1)^{p/(p-1)}$, which is a graph of girth 6 with $|X| = |Y| = d_H^2$. Then we further subsample G to ensure that there are no cycles of length less than 5 consisting of outer edges (some were introduced via the way we combined $\widetilde{G}$ with H). All of this is necessary in order to ensure that in any 3-spanner of G, the only ways of spanning an outer edge are through the outer edge itself or through a canonical path (and in particular, there is no way to span it using just other outer edges).

We give a sketch of the analysis and proof here, since it is simply re-analyzing the construction of [9] using the ideas from the previous section. It is straightforward to prove the analog of Lemma 23, since we use the same spanner suggested by the existence of a good REP-cover and analyze all degrees in the same way. This implies that in a YES instance, there will be a k-spanner S with $\|S\|_p \leq O((|U||X|+|V||Y|)^{1/p} \cdot d_H)$.

The NO setting is more difficult to analyze, since it requires arguing directly about the subsampling process. But if we follow the analysis in [9] but with the notation from the previous section, we get that in a NO instance,

$$\sum_{u \in U}(d^{x,y}_{out}(u,x) + d^{x,y}_{con}(u,x)) + \sum_{v \in V}(d^{x,y}_{out}(v,y) + d^{x,y}_{con}(v,y)) \geq s^{1/8}(|U|+|V|)$$

for every $\{x, y\} \in E_H$. This is the equivalent of Equation (11) but as a direct proof rather than by contradiction. Then as in the directed case, we can combine these to get the following theorem (the dependence on p is slightly worse since the graph that we build is larger due to using the finite projective plane rather than the complete bipartite graph).

▶ **Theorem 26.** *Unless* NP $\subseteq$ BPTIME($2^{\text{polylog}(n)}$), *for any constant* $\epsilon > 0$ *and* $p \geq 1$ *there is no polynomial-time algorithm that can approximate* Minimum ℓ_p-Norm 3-Spanner *better than* $2^{\left(\frac{p-1}{4p-1}\right)^{1-\epsilon} \log^{1-\epsilon} n}$.

Theorem 8 is now a corollary of this theorem when $p = 2$.

Conditional Hardness of Earth Mover Distance

Dhruv Rohatgi
MIT, Cambridge, Massachusetts, USA
drohatgi@mit.edu

Abstract

The Earth Mover Distance (EMD) between two sets of points $A, B \subseteq \mathbb{R}^d$ with $|A| = |B|$ is the minimum total Euclidean distance of any perfect matching between A and B. One of its generalizations is asymmetric EMD, which is the minimum total Euclidean distance of any matching of size $|A|$ between sets of points $A, B \subseteq \mathbb{R}^d$ with $|A| \leq |B|$. The problems of computing EMD and asymmetric EMD are well-studied and have many applications in computer science, some of which also ask for the EMD-optimal matching itself. Unfortunately, all known algorithms require at least quadratic time to compute EMD exactly. Approximation algorithms with nearly linear time complexity in n are known (even for finding approximately optimal matchings), but suffer from exponential dependence on the dimension.

In this paper we show that significant improvements in exact and approximate algorithms for EMD would contradict conjectures in fine-grained complexity. In particular, we prove the following results:

- Under the Orthogonal Vectors Conjecture, there is some $c > 0$ such that EMD in $\Omega(c^{\log^* n})$ dimensions cannot be computed in truly subquadratic time.
- Under the Hitting Set Conjecture, for every $\delta > 0$, no truly subquadratic time algorithm can find a $(1 + 1/n^\delta)$-approximate EMD matching in $\omega(\log n)$ dimensions.
- Under the Hitting Set Conjecture, for every $\eta = 1/\omega(\log n)$, no truly subquadratic time algorithm can find a $(1 + \eta)$-approximate asymmetric EMD matching in $\omega(\log n)$ dimensions.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Earth Mover Distance, Hardness of Approximation, Fine-Grained Complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.12

Category APPROX

Acknowledgements I want to thank Piotr Indyk and Arturs Backurs for numerous helpful discussions and guidance. I am also grateful to an anonymous reviewer for pointing towards Theorem 2 and its proof.

1 Introduction

In the *Earth Mover Distance (EMD) problem*, we are given two sets A and B each with n vectors in $\mathbb{R}^d$, and want to find the minimum cost of any perfect matching between A and B, where an edge between $a \in A$ and $b \in B$ has cost $\|a - b\|_2$.

In a harder variant of the problem ("EMD matching"), we want to actually *find* a perfect matching with the optimal cost. This is a special case of the *geometric transportation problem*, in which each vector of A has a positive supply and each vector of B has a positive demand, and the goal is to find an optimal "transportation map", i.e., match each unit of supply with a unit of demand while minimizing the total distance, summed over all units of supply.

A more general variant of the EMD problem (with an analogous extension to arbitrary supplies/demands) allows for the possibility that $|A| < |B|$, and requires the map from A to B to be an injection. We refer to this variant as the *asymmetric* EMD problem.


© Dhruv Rohatgi;
licensed under Creative Commons License CC-BY



Earth Mover Distance is a discrete analogue of the Monge-Kantorovich metric for probability measures, which has connections to various areas of mathematics [26]. Furthermore, computing distance between probability measures is an important problem in machine learning [23, 20, 7, 13] and computer vision [22, 10, 25], to which Earth Mover Distance is often applied. To provide a few specific examples, computing geometric transportation cost has applications in image retrieval [22], where asymmetric EMD allows the distance to deal with occlusions and clutter. In computer graphics, computing the actual transportation map is useful for interpolation between distributions, though the metric may be non-Euclidean [10].

For the exact geometric transportation problem, the best known algorithm simply formulates the problem in terms of minimum cost flow, yielding a runtime of $O(n^{2.5} \cdot \text{polylog}(U))$ where U is the total supply (assuming that d is subpolynomial in n) [18, 19]. Even for EMD, the best known algorithm follows directly from the general graph algorithms for maximum matching in $O(m\sqrt{n})$ time [14].

The situation is better for approximation algorithms. There has been considerable work on both estimating the transportation cost [15, 6] and computing the actual map [24, 3, 5] in time nearly linear in n but exponential in dimension d. Most recently, it was shown [17] that there is an $O(n\epsilon^{-O(d)} \log(U)^{O(d)} \log^2 n)$ time algorithm which outputs a transportation map with cost at most $(1+\epsilon)$ times the optimum. This algorithm is very efficient when the dimension d is constant or nearly constant, and when ϵ is not too small – say, constant or $O(1/\text{polylog}(n))$. However, when $d = \omega(\log n)$, the algorithm is not guaranteed to find even a constant-factor approximation in quadratic time.

Despite considerable progress on improving the *algorithms* for geometric matching problems over the last two decades, little is known about *lower bounds* on their computational complexity. In particular, we do not have any evidence that a running time of the form $O(n \cdot \text{poly}(d, \log n, 1/\epsilon))$ is not achievable. This is the question we address in this paper.

1.1 Our Results

In this paper we provide evidence that geometric transportation problems in high-dimensional spaces cannot be solved in (truly) subquadratic time. This applies to both exact and approximate variants of the problem, and even in the special case of unit supplies. In particular we show a conditional quadratic hardness for the exact EMD problem, as well as the approximate variant of EMD when the (approximately) optimal matching must be reported.

Our hardness results are based on two well-studied conjectures in fine-grained complexity: Orthogonal Vectors Conjecture and Hitting Set Conjecture (see [29] for a comprehensive survey).

1.1.1 Exact EMD and Orthogonal Vectors Conjecture

The *Orthogonal Vectors (OV) problem* takes as input two sets $A, B \subseteq \{0,1\}^{d(n)}$ where $|A| = |B| = n$ and asks whether there are some vectors $a \in A$ and $b \in B$ such that $a \cdot b = 0$. The popular *Orthogonal Vectors Conjecture* hypothesizes that in sufficiently large dimensions, the obvious quadratic time algorithm for OV is nearly optimal:

▶ **Orthogonal Vectors Conjecture.** *Let $d(n) = \omega(\log n)$. For every constant $\epsilon > 0$, no randomized algorithm can solve $d(n)$-dimensional OV in $O(n^{2-\epsilon})$ time.*

A plethora of problems have been shown to have nontrivial lower bounds under the Orthogonal Vectors Conjecture; often these lower bounds are essentially tight (e.g. [1, 2, 8, 11, 28]; see [29] for a comprehensive survey). It is known that if the conjecture fails, then the Strong Exponential Time Hypothesis (SETH) fails as well [27], providing evidence for hardness of OV, and by extension of these problems to which OV can be reduced.

Our first result shows that EMD in "nearly constant" dimension is hard to compute exactly in truly subquadratic time, under the Orthogonal Vectors Conjecture:

▶ **Theorem 1.** *There is a constant $c > 0$ under which the following holds. If there exists $\epsilon > 0$ and $d(n) = \Omega(c^{\log^* n})$ such that EMD on $O(\log n)$-bit vectors in $d(n)$ dimensions can be computed in $O(n^{2-\epsilon})$ time, then the Orthogonal Vectors Conjecture is false.*

Using techniques similar to those for the above theorem, we also address a question raised in [9] about the complexity of the maximum/minimum weighted assignment problem when the weight matrix has low rank. The minimum weighted assignment problem is defined as follows: given an $n \times n$ weight matrix which determines a complete bipartite graph, find the cost of the minimum weight perfect matching. Motivated by the observation that the problem can be solved in $O(n \log n)$ time if the weight matrix is rank-1, it is asked whether there is an $O(nr^2 \log n)$ time algorithm for rank-r matrices [9]. We can answer this question in the negative, under the Orthogonal Vectors Conjecture. In fact, we can show something stronger (see Appendix A for the proof):

▶ **Theorem 2.** *There is a constant $c > 0$ under which the following holds. If there exists $\epsilon > 0$ and $r(n) = \Omega(c^{\log^* n})$ such that the minimum assignment problem with rank-r weight matrices can be solved in $O(n^{2-\epsilon})$ time, then the Orthogonal Vectors Conjecture is false.*

1.1.2 Approximate EMD and the Hitting Set Conjecture

The second conjecture on which we base some of our results is hardness of the *Hitting Set (HS) problem*. This problem, similar to OV, takes two sets of vectors $A, B \subseteq \{0,1\}^d$ as input, and asks whether there exists some $a \in A$ such that $a \cdot b \neq 0$ for every $b \in B$.

▶ **Hitting Set Conjecture.** *Let $d(n) = \omega(\log n)$. For every constant $\epsilon > 0$, no randomized algorithm can solve $d(n)$-dimensional HS in $O(n^{2-\epsilon})$ time.*

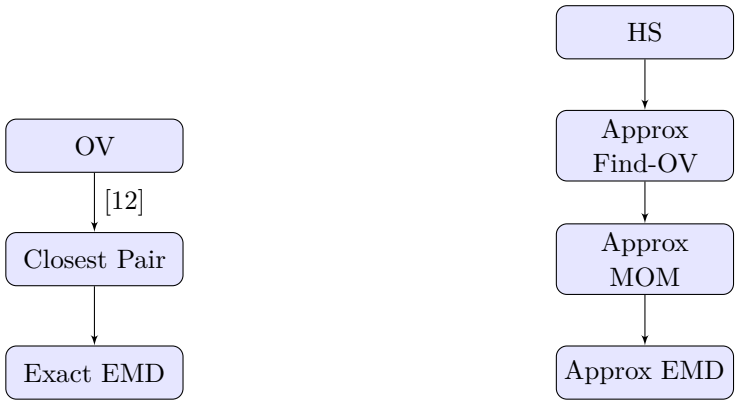
It is known that HS reduces to OV, but the reverse reduction is unknown, so the Hitting Set Conjecture is "stronger" than the Orthogonal Vectors Conjecture [2]. The Hitting Set Conjecture has been used to prove conditional hardness of the Radius problem in sparse graphs [2]. The utility of the Hitting Set problem in conditional hardness results comes from the difference between its "$\exists\forall$" logical structure and the "$\exists\exists$" logical structure of the Orthogonal Vectors problem, which makes it more natural for some types of problems.

Under the Hitting Set Conjecture, we prove hardness of approximation for the EMD matching problem (in which we want to find the optimal or nearly-optimal matching). Simultaneously we obtain stronger hardness of approximation for asymmetric EMD matching.

▶ **Theorem 3.** *For any $\delta > 0$ and $d(n) = \omega(\log n)$, if $(1 + 1/n^\delta)$-approximate EMD matching can be solved in $d(n)$ dimensions in truly subquadratic time, then the Hitting Set conjecture is false.*

▶ **Theorem 4.** *For any $d(n) = \omega(\log n)$ and $\eta = 1/\omega(\log n)$, if $(1 + \eta)$-approximate asymmetric EMD matching can be solved in $d(n)$ dimensions in truly subquadratic time, then the Hitting Set Conjecture is false.*

Finally, motivated by the question of how hard Hitting Set really is, compared to Orthogonal Vectors, we generalize the result that Hitting Set reduces to Orthogonal Vectors by finding a set of approximation problems that lie between Orthogonal Vectors and Hitting Set in difficulty. For a positive integer function $k(n) \leq n/2$, we define the $(k, 2k)$-*Find-OV*



(a) Structure of Theorem 1. **(b)** Structure of Theorem 3.

Figure 1 Summary of reductions.

problem: given two sets $A, B \subseteq \{0,1\}^{d(n)}$ with $|A| = |B| = n$ and the guarantee that there exist at least $2k$ orthogonal pairs between A and B, find k pairs $\{(a_i, b_i)\}_{i=1}^{k}$ such that $a_i \cdot b_i = 0$ for every i.

We prove the following theorem in Appendix C.

▶ **Theorem 5.** *Let $k(n) \leq n/2$. If $(k, 2k)$-Find-OV can be solved in truly subquadratic time, then the Hitting Set conjecture is false.*

See Figure 1 for an overview of the structure of our main results (Theorems 1 and 3 respectively; the proof of Theorem 4 has the same structure as the latter). We provide the remaining definitions of the relevant problems in the next section.

2 Preliminaries

Before diving into the reductions, we formally define the remainder of the problems which we're studying. Each problem we study takes sets of vectors as input, so one parameter of a problem is the dimension d, which is a function of the input size n. That is, every function $d : \mathbb{N} \to \mathbb{N}$ defines a $d(n)$-dimensional EMD problem, and a $d(n)$-dimensional OV problem, and so forth. We gloss over this choice of d in the subsequent definitions.

2.1 Earth Mover Distance

The *Earth Mover Distance (EMD)* problem is defined as follows: given two sets $A, B \subseteq \mathbb{R}^{d(n)}$ with $|A| = |B|$, find

$$\min_{\pi : A \to B} \sum_{a \in A} \|a - \pi(a)\|_2$$

where π is a bijection. We'll restrict our attention to the special cases where $A, B \subseteq \mathbb{Z}^{d(n)}$ with polynomially bounded entries (for hardness of exact EMD) and $A, B \subseteq \{0,1\}^{d(n)}$ (for hardness of approximate EMD).

We can define the *asymmetric EMD* problem as above, except we relax the constraint $|A| = |B| = n$ to $|A| \leq |B| = n$, and require π to be a injection rather than a bijection.

The *EMD matching* problem is the variant of the EMD problem in which the desired output is the optimal matching π. Similarly we can define the *asymmetric EMD matching* problem. An algorithm "solves" EMD matching (or its asymmetric variant) up to a certain additive or multiplicative factor if the cost of the bijection it outputs differs from the optimal cost by at most that additive or multiplicative factor.

2.2 Variants of Orthogonal Vectors

The reduction from Hitting Set to approximate EMD matching will go through the variants of OV defined next.

The *Maximum Orthogonal Matching (MOM)* problem is defined as follows: given two sets $A, B \subseteq \{0,1\}^{d(n)}$, with $|A| \leq |B| = n$, find an injection $\pi : A \to B$ which maximizes

$$|\{a \in A \mid a \cdot \pi(a) = 0\}|.$$

And the *Find-OV* problem is defined as follows: given two sets $A, B \subseteq \{0,1\}^{d(n)}$ with $|A| = |B| = n$, find the set $S \subseteq A$ of vectors $a \in A$ such that there exists some $b \in B$ with $a \cdot b = 0$. An algorithm solves Find-OV up to an additive error of t if it returns a set $S' \subseteq S$ for which $|S'| \geq |S| - t$.

2.3 Relevant prior work

We will apply the following theorem from [12] to our low-dimensional hardness result of exact EMD:

▶ **Theorem 6** ([12]). *Assuming OVC, there is a constant $c > 0$ such that Bichromatic ℓ_2-Closest Pair in $c^{\log^* n}$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$ bit entries.*

3 Exact EMD in low dimensions

To prove hardness of the exact EMD problem under the Orthogonal Vectors Conjecture, we reduce to the bichromatic closest pairs problem, and then apply Theorem 6 due to [12]. The intuition for the reduction is as follows: given two sets A and B of n vectors, we'd like to augment set A with $n-1$ copies of a vector that is equidistant from all of B, and much closer to B than A is. Similarly, we'd like to augment set B with $n-1$ copies of a vector that is equidistant from all of A, and much closer to A than B is. If this were possible, then the minimum cost matching between the augmented sets would only match one pair of the original sets: the desired closest pair.

Unfortunately, it is in general impossible to find a vector equidistant from n vectors in $d \ll n$ dimensions. But this can be circumvented by embedding the vectors in a slightly higher-dimensional space, and adjusting coordinates in the "free" dimensions to ensure that an equidistant vector exists. So long as the free dimensions used to adjust set A are disjoint from the free dimensions used to adjust set B, the inner products between A and B are unaffected, and the distances change in an accountable way.

Since we are working in the ℓ_2 norm, we will need the following simple lemma which shows that any integer can be efficiently decomposed as a sum of a constant number of perfect squares.

▶ **Lemma 7.** *For any $\rho > 0$ and any positive integer m, there is an $O(m^\rho)$ time algorithm to decompose m as a sum of $O(\log 1/\rho)$ perfect squares.*

Proof. Here is the algorithm: repeatedly find the largest square which does not push the total above m, until the remainder does not exceed $O(m^{\rho/2})$. Then compute the minimal square decomposition for the remainder by dynamic programming.

The first, greedy phase takes $O(\text{polylog}(m))$ time and finds $O(\log 1/\rho)$ perfect squares which sum to some m' with $m - m^{\rho/2} \le m' \le m$. The second, dynamic programming phase takes $O(m^\rho)$ time (even naively). By Lagrange's four-square theorem, a decomposition of $m - m'$ into at most four perfect squares is found. ◀

Now we describe the main reduction of this section. We'll use a shorthand notation to define vectors more concisely: for example, $a^x b^y c^z$ refers to an $(x + y + z)$-dimensional vector with value a in the first x dimensions, b in the next y dimensions, and c in the next z dimensions.

▶ **Theorem 8.** *Let $d = d(n) \le n$ be a dimension, and let $k > 0$ be a constant. There is a constant $c = c(k)$ for which the following holds. Suppose that there is an algorithm which computes the ℓ_2 earth mover distance between sets $A', B' \subseteq [1, n^{16k}]^{2d+2c+2}$ of size n in $O(n^{2-\epsilon})$ time. Then bichromatic closest pair between sets $A, B \subseteq [1, n^k]^d$ of size n can be computed in $O(n^{2-\epsilon})$ time as well.*

Proof. Set $\rho = 1/(16k)$, and let $c = O(\log 1/\rho)$ be the constant in Lemma 7 for the number of perfect squares in a decomposition. Let A and B be two sets of vectors from $\{1, \dots, n^k\}^d$. Let $N = n^{16k}$. Our goal is to compute

$$\min_{a \in A, b \in B} \|a - b\|_2 .$$

We can assume without loss of generality that $\|a\|_2^2$ and $\|b\|_2^2$ are odd for all $a \in A$ and $b \in B$: for instance, we can replace each vector $z = (z_1, \dots, z_d)$ by $(2z_1, \dots, 2z_d, 1)$.

We construct sets A' and B' of $(2d + 2c + 2)$-dimensional vectors as follows. Let $u = 0^d(10^c)0^{c+1}0^d$ (parentheses for clarity). Let $v = N^d 0^{c+1}(10^c)0^d$. Add $n - 1$ copies of u to B' and add $n - 1$ copies of v to A'. For each $a \in A$, add the following vector to A', where we'll define vector $\text{adj}_a \in \mathbb{Z}^{c+1}$ later:

$$a' = f(a) = 0^d(\text{adj}_a)0^{c+1}a.$$

Similarly, for each $b \in B$, add the following vector to B', where we'll define $\text{adj}_b \in \mathbb{Z}^{c+1}$ later:

$$b' = g(b) = N^d 0^{c+1}(\text{adj}_b)b.$$

Now pick any $a \in A$. We'll construct adj_a so that the following equalities are both satisfied:

$$\|a' - u\|_2^2 = n^{4k}d^2 = \|\text{adj}_a\|_2^2 .$$

Define the first element $\text{adj}_a(0) = (\|a\|_2^2 + 1)/2$. Since $\|a\|_2^2 \le n^{2k}d$, we can then use Lemma 7 to find c integers $\text{adj}_a(1), \dots, \text{adj}_a(c)$ so that $\|\text{adj}_a\|_2^2 = n^{4k}d^2$. Furthermore,

$$\begin{aligned}\|a' - u\|_2^2 &= \|\text{adj}_a - 10^c\|_2^2 + \|a\|_2^2 \\ &= \|\text{adj}_a\|_2^2 - 2 \cdot \text{adj}_a(0) + 1 + \|a\|_2^2 \\ &= n^{4k}d^2.\end{aligned}$$

For each $b \in B$, we can similarly construct adj_b so that $\|b' - v\|_2^2 = \|\text{adj}_b\|_2^2 = n^{4k}d^2$.

We claim that

$$\text{EMD}(A', B') = 2(n-1)n^{2k}d + \min_{a\in A, b\in B} \sqrt{N^2 d + 2n^{4k}d^2 + \|a-b\|_2^2}.$$

To prove this claim, notice that $\|u-v\|_2 \geq N\sqrt{d}$ and $\|a'-b'\|_2 \geq N\sqrt{d}$ for every $a' \in A' \setminus \{v\}$ and $b' \in B' \setminus \{u\}$, whereas $\|a'-u\|_2 \ll N\sqrt{d}/n$ and $\|b'-v\| \ll N\sqrt{d}/n$. This means that the optimal matching between A' and B' will minimize the number of (u,v) and (a',b') edges. Hence, exactly one element of $A' \setminus \{v\}$ will be matched to an element in $B' \setminus \{u\}$. So if M denotes this optimal matching, and $x' = f(x) \in A'$ is matched with $y' = g(y) \in B'$, then the cost of M is

$$\begin{aligned}
\text{cost}(M) &= \left(\sum_{a'\in A'\setminus\{v,x'\}} \|a'-u\|_2 + \sum_{b'\in B'\setminus\{u,y'\}} \|b'-v\|_2 \right) + \|x'-y'\|_2 \\
&= 2(n-1)n^{2k}d + \sqrt{N^2 d + \|\text{adj}_x\|_2^2 + \|\text{adj}_y\|_2^2 + \|x-y\|_2^2} \\
&= 2(n-1)n^{2k}d + \sqrt{N^2 d + 2n^{4k}d^2 + \|x-y\|_2^2}.
\end{aligned}$$

The claim follows. So the algorithm is simply: run the EMD algorithm on (A', B') and use the computed matching cost to find the closest pair distance, according to the above formula.

The time complexity of constructing A', B' is $O(n^{5/4}d^{1/8})$, dominated by computing a square decomposition for each vector. Since A' and B' are sets of $O(n)$ vectors in $\mathbb{Z}^{2d+2c+2}$ with entries bounded by $\max(N, n^{2k}d) \leq n^{16k}$, the EMD between A' and B' can be computed in $O(n^{2-\epsilon})$ time. Thus, the overall algorithm takes $O(n^{2-\epsilon})$ time. ◀

Theorem 1 follows from the above reduction and Theorem 6.

4 Approximate EMD under the Hitting Set Conjecture

In this section we prove hardness of approximation for the EMD matching problem when the approximately optimal matching must be reported. Note that the techniques from the previous section do not immediately generalize to this scenario, since the reduction in Theorem 8 is not approximation-preserving. A multiplicative error of $1+\epsilon$ in the EMD algorithm would induce an additive error of $\tilde{O}(\epsilon n^{16k})$ in the closest pair algorithm, due to the large integers constructed in the reduction. A bucketing scheme, to ensure that the diameter of the input point set is within a constant factor of the closest pair, could eliminate the dependence on the values of the input coordinates, yielding a multiplicative error of only $1 + \tilde{O}(\epsilon n)$.

However, $(1+\epsilon)$-approximate closest pair is only quadratically hard for $\epsilon = o(1)$ [21]; for any constant $\epsilon > 0$, there is a subquadratic $(1+\epsilon)$-approximation algorithm [16, 4]. Thus, the above arguments would only yield $(1+\tilde{O}(1/n))$-approximate hardness. Furthermore, the factor of n loss intuitively feels intrinsic to the approach of reducing from closest pair, since the EMD is the sum of n distances. Thus, a different approach seems necessary if we are to achieve hardness for $\epsilon = \omega(1/n)$.

Our method broadly consists of two steps. First, we show that EMD can encode orthogonality, by reducing approximate Maximum Orthogonal Matching (the problem of reporting a maximum matching in the implicit graph with an edge for each orthogonal pair) to approximate EMD matching. Second, we show that approximate Maximum Orthogonal matching can solve an instance (A, B) of Hitting Set by finding an orthogonal pair (a, b) for every $a \in A$ if possible, even if the set of orthogonal pairs does not constitute a matching.

We start by proving that asymmetric EMD matching reduces to EMD matching for the appropriate choices of error bounds. The reduction pads the smaller set of vectors A with a vector that is equidistant from the opposite set B, so that its contribution to the earth mover distance can be accounted for. Of course, it is first necessary to transform the vectors so that an equidistant vector exists.

▶ **Lemma 9.** *Suppose that $(1+\epsilon)$-approximate EMD matching in D dimensions can be solved in $T(n, D)$ time. Then $(1+\epsilon)$-approximate asymmetric EMD matching in d dimensions can be solved with an additional additive factor of $n\epsilon\sqrt{d}$ in $T(n, 2d)$ time.*

Proof. Let $A, B \subseteq \{0,1\}^d$ with $|A| \leq |B|$. Define sets $A', B' \subseteq \{0,1\}^{2d}$ by mapping $a \in A$ to the vector

$$(a_1, \ldots, a_d, 1-a_1, \ldots, 1-a_d)$$

and similarly mapping $b \in B$ to

$$(b_1, \ldots, b_d, 1-b_1, \ldots, 1-b_d).$$

Then add $|B| - |A|$ copies of the zero vector to A'.

Now $|A'| = |B'|$, so we can run the approximate EMD algorithm on A' and B' to find some bijection $\pi : A' \to B'$ such that

$$\sum_{a' \in A'} \|a' - \pi(a')\|_2 \leq (1+\epsilon) EMD(A', B').$$

Each vector $b' \in B'$ has $\|b'\|_2^2 = d$, so the distance from the zero vector to each match is exactly $\sqrt{d}$. And for any $a \in A$ and $b \in B$ which map to $a' \in A'$ and $b' \in B'$,

$$\|a' - b'\|_2^2 = 2\|a - b\|_2^2.$$

Hence, the cost of π is

$$\sum_{a' \in A'} \|a' - \pi(a')\|_2 = (|B| - |A|)\sqrt{d} + \sqrt{2} \cdot \sum_{a \in A} \|a - \pi(a)\|_2$$

and the optimal cost is

$$EMD(A', B') = (|B| - |A|)\sqrt{d} + \sqrt{2} \cdot EMD(A, B).$$

It follows that

$$\sum_{a \in A} \|a - \pi(a)\|_2 \leq \frac{\epsilon}{\sqrt{2}}(|B| - |A|)\sqrt{d} + (1+\epsilon) EMD(A, B),$$

which is the stated error bound. ◀

Next, we reduce approximate Maximum Orthogonal Matching to approximate asymmetric EMD matching. The general idea, given input sets (A, B), is to deform A and B so that orthogonal pairs (a, b) are mapped to pairs (a'', b'') with distance d_0, and all other pairs are mapped to pairs with distance at least $d_1 > d_0$. Then add $|A|$ auxiliary vectors to B, each with distance exactly d_1 from all vectors in A. Thus, in an optimal matching, each vector of A is either matched with an orthogonal vector at distance d_0, or some vector with distance exactly d_1. This introduces a nonlinearity, ensuring that in the additive matching cost, an

orthogonal pair's contribution is not "cancelled out" by the contribution of a pair with dot product 2, for instance. A similar trick was used by [8] in the context of edit distance, another "additive" metric.

The following simple lemma will be useful:

▶ **Lemma 10.** *There are maps $\phi_1, \phi_2 : \{0,1\}^d \to \{0,1\}^{3d}$ such that for any $a, b \in \{0,1\}^d$,*

$$\phi_1(a) \cdot \phi_2(b) = d - (a \cdot b).$$

Furthermore, the maps can be evaluated in $O(d)$ time.

Proof. Each dimension expands into three dimensions as follows:

$$a_i \mapsto (\phi_1(a)_{3i}, \phi_1(a)_{3i+1}, \phi_1(a)_{3i+2}) = (a_i, 1-a_i, 1-a_i)$$

$$b_i \mapsto (\phi_2(b)_{3i}, \phi_2(b)_{3i+1}, \phi_2(b)_{3i+2}) = (1-b_i, b_i, 1-b_i).$$

Then for each i,

$$\sum_{j=3i}^{3i+2} \phi_1(a)_j \phi_2(b)_j = a_i(1-b_i) + (1-a_i)b_i + (1-a_i)(1-b_i) = 1 - a_i b_i.$$

Summing over $i = 1, \ldots, d$ we get $\phi_1(a) \cdot \phi_2(b) = d - (a \cdot b)$ as desired. ◀

▶ **Lemma 11.** *Suppose that $(1+\epsilon)$-approximate asymmetric EMD in D dimensions can be solved with an additional additive factor of $n\epsilon\sqrt{D}$ in $T(n, D)$ time. Then the Maximum Orthogonal Matching problem in d dimensions can be solved up to an additive factor of $O(n\epsilon d)$ in $T(2n, 12d+1)$ time.*

Proof. Let $A, B \subseteq \{0,1\}^d$ with $|A| \le |B| = n$. Define $A', B' \subseteq \{0,1\}^{3d}$ by $A' = \phi_1(A)$ and $B' = \phi_2(B)$, where ϕ_1, ϕ_2 are as defined in Lemma 10.

Let $d' = 3d$ for convenience. Now we construct sets $A'', B'' \subseteq \{0,1\}^{4d'+1}$ as follows, starting from sets A' and B'. We add $2d'$ dimensions to ensure that $\|a''\|_2^2 = \|b''\|_2^2 = d'$ for every $a'' \in A''$ and $b'' \in B''$ without changing the inner products. Add another $d'+1$ dimensions, extending each $a'' \in A''$ so that $a''_{3d'+1} = 1$ and $a''_i = 0$ otherwise; and extend each $b'' \in B''$ so that $b''_{3d'+2} = 1$ and $b''_i = 0$ otherwise. Finally augment B'' with $|A|$ copies of the vector $v \in \{0,1\}^{4d'+1}$ with $3d'$ zeros followed by $d'+1$ ones.

Notice that for every $a \in A$ and $b \in B$ corresponding to some $a'' \in A''$ and $b'' \in B''$,

$$\|a'' - b''\|_2^2 = \|a''\|_2^2 + \|b''\|_2^2 - 2a'' \cdot b'' = 2(d'+1) - 2a'' \cdot b'' = 2a \cdot b + 4d + 2,$$

and

$$\|a'' - v\|_2^2 = 2(d'+1) - 2a'' \cdot v = 4d + 4.$$

Now we run the approximate asymmetric EMD matching algorithm on A'' and B'', yielding an injection $\pi : A'' \to B''$ such that

$$\sum_{a'' \in A''} \|a'' - \pi(a'')\|_2 \le |B''|\epsilon\sqrt{4d'+1} + (1+\epsilon)EMD(A'', B'').$$

For each $a'' \in A''$, if $\|a'' - \pi(a'')\|_2^2 > 4d+4$, then we can set $\pi(a'') = v$, preserving injectivity and decreasing the cost of the matching. Therefore every edge has cost either $\sqrt{4d+2}$ or $\sqrt{4d+4}$. In particular, if there are m orthogonal pairs in the matching, the total cost is

$$\sum_{a'' \in A''} \|a'' - \pi(a'')\|_2 = m\sqrt{4d+2} + (|A| - m)\sqrt{4d+4}.$$

By the same argument as above, the minimum cost matching is obtained by maximizing the number of orthogonal pairs. If the maximum possible number of orthogonal pairs in a matching is m_{OPT}, then

$$EMD(A'', B'') = m_{\text{OPT}}\sqrt{4d+2} + (|A| - m_{\text{OPT}})\sqrt{4d+4}.$$

Substituting these expressions into the approximation guarantee and solving, we get that $m \geq m_{\text{OPT}} - O(\epsilon nd)$ as desired. ◀

In the above lemma we assumed that we are given an algorithm for asymmetric EMD matching which has both a multiplicative error of $1+\epsilon$ and an additive error of $n\epsilon\sqrt{d}$, since this is the error introduced by the reduction to (symmetric) EMD. However, we are also interested in the hardness of $(1+\epsilon)$-approximate asymmetric EMD matching in its own right. Removing the additive error from the hypothesized algorithm in Lemma 11 directly translates to an improved Maximum Orthogonal Matching algorithm, with an additive error of $O(\epsilon|A|d)$ instead of $O(\epsilon nd)$, where $n = |A| + |B|$:

▶ **Lemma 12.** *Suppose that there is an algorithm which solves $(1+\epsilon)$-approximate asymmetric EMD matching in $T(|A|+|B|, d)$ time, where the input is $A, B \subseteq \{0,1\}^d$. Then the Maximum Orthogonal Matching problem can be solved up to an additive error of $O(\epsilon|A|d)$ in $T(2n, 12d+1)$ time.*

Now we could reduce OV to approximate Maximum Orthogonal Matching. The proof of the following theorem is given in Appendix B for completeness.

▶ **Theorem 13.** *Let $d = \omega(\log n)$. Under the Orthogonal Vectors Conjecture, for any $\epsilon > 0$ and $\delta \in (0,1)$, $(1+1/n^\delta)$-approximate EMD matching in $\{0,1\}^d$ cannot be solved in $O(n^{2\delta-\epsilon})$ time.*

However, Theorem 13 does not prove quadratic hardness for any approximation factor larger than $(1+1/n)$, and in fact breaks down completely for $(1+1/\sqrt{n})$-approximate EMD matching.

Instead, we reduce Hitting Set to approximate Maximum Orthogonal Matching, through approximate Find-OV. These two problems are structurally similar; the technical difficulty is that Find-OV may require finding many orthogonal pairs even when the largest orthogonal matching may be small, in which case applying the Maximum Orthogonal Matching algorithm would result in little progress. We resolve this with the following insight: if many vectors in set A are orthogonal to at least one vector in set B but there is not a large orthogonal matching, then some vector in set B is orthogonal to *many* vectors in A. But these vectors can be found efficiently by sampling.

In the proof of the following theorem we formalize the above idea.

▶ **Theorem 14.** *Let $d = d(n)$ be a dimension. Suppose that the Maximum Orthogonal Matching problem can be solved up to an additive error of $E(|A|, |B|)$ in $O(n^{2-\epsilon}poly(d))$ time, where the input is $A, B \subseteq \{0,1\}^d$. Then for any (sufficiently small) $\alpha > 0$ there is some $\gamma > 0$ such that Find-OV can be solved with high probability up to an additive error of $E(|A|, 2|B|^{1+\alpha})$ in $O(n^{2-\gamma}poly(d))$ time.*

Proof. Let $A, B \subseteq \{0,1\}^d$ with $|A| = |B| = n$. Let $\alpha > 0$ be a constant we choose later. We may safely assume that $\alpha < 1$. Let the degree of a vector $a \in A$, denoted $d(a)$, be the number of $b \in B$ which are orthogonal to a. The algorithm for Find-OV consists of three steps:

1. For every $a \in A$, sample $n^{1-\alpha/4}$ vectors from B to get an estimate $\hat{d}(a)$ of $d(a)$. Mark and remove the vectors for which $\hat{d}(a) \geq n^{\alpha/2}$.
2. Next, for every $b \in B$, sample $n^{1-\alpha/2}$ vectors from A to get an estimate $\hat{d}(b)$ of $d(b)$. Let $B_{\text{large}} \subseteq B$ be the set of vectors for which $\hat{d}(b) \geq n^{\alpha}$. For each $b \in B_{\text{large}}$, iterate over A and mark and remove each $a \in A$ for which $a \cdot b = 0$. Now remove B_{large} from B.
3. Run the Maximum Orthogonal Matching algorithm on the remaining set A, and the multiset consisting of $2n^{\alpha}$ copies of each remaining $b \in B$. This produces a set of pairs (a_i, b_i) where $a_i \cdot b_i = 0$. Output the union of $\{a_i\}_i$ and the set of all vectors marked and removed from A in the previous steps.

In the first step, a Chernoff bound shows that with high probability, every vector for which $d(a) \geq 2n^{\alpha/2}$ is marked and removed. Now summing over the remaining vectors,

$$\sum_{a \in A} d(a) = \sum_{b \in B} d(b) \leq 2n^{1+\alpha/2}.$$

In the second step, with high probability B_{large} contains no $b \in B$ for which $d(b) \leq \frac{1}{2}n^{\alpha}$, by a Chernoff bound on each such $b \in B$. Therefore $|B_{\text{large}}| \leq 4n^{1-\alpha/2}$. Furthermore, with high probability B_{large} contains every $b \in B$ for which $d(b) \geq 2n^{\alpha}$.

So after the first two steps, every remaining vector $b \in B$ has degree at most $2n^{\alpha}$. Suppose there are t vectors $a \in A$ with positive degree, and t' of these are found in the first two steps. Then by the degree bound, the remaining $t - t'$ vectors inject into $2n^{\alpha}$ copies of B. Therefore there is an orthogonal matching of size at least $t - t'$. By the approximation guarantee of the Maximum Orthogonal Matching algorithm, we find an orthogonal matching of size at least $t - t' - 2n^{(1+\alpha)\delta}$ in step 3. Overall, we find at least $t - 2n^{(1+\alpha)\delta}$ vectors with positive degree, which gives the desired approximation guarantee.

The time complexity is $O((n^{2-\alpha/4} + n^{(2-\epsilon)(1+\alpha)})\text{poly}(d))$. This is subquadratic in n for sufficiently small α. ◀

As the final step of the reduction, we show that approximate Find-OV can solve Hitting Set. Note that exact Find-OV obviously solves Hitting Set. It's also clear that Find-OV with an additive error of $n^{1-\epsilon}$ solves Hitting Set: simply run Find-OV, and then exhaustively check the remaining unpaired vectors of A – unless there are more than $n^{1-\epsilon}$ unpaired vectors, in which case there must be a hitting vector.

To reduce Hitting Set to Find-OV with additive error of $\Theta(n)$, the essential idea is simply to repeatedly run Find-OV on the remaining unpaired vectors. If the Find-OV algorithm has an additive error of $n/2$, then given an input A, B with no hitting vector, the algorithm will find orthogonal pairs for at least $n/2$ vectors of A. Naively, we'd like to recurse on the remaining half of A. Unfortunately, the set B cannot similarly be halved, so the error bound in the next step would not be halved. Thus, the algorithm might make no further progress.

The workaround is to duplicate every unpaired vector of A before recursing. If $n/2$ orthogonal pairs are found but every vector of A has been duplicated once, then matches are found for at least $n/4$ distinct vectors. This suffices to terminate the recursion in $O(\log n)$ steps.

▶ **Theorem 15.** *Suppose that Find-OV in d dimensions can be solved up to additive error of $n/2$ in $T(n, d)$ time. Then Hitting Set in d dimensions can be solved in $O((T(n, d) + nd)\log n)$ time.*

Proof. Let $A, B \subseteq \{0, 1\}^d$ with $|A| = |B| = n$. Our hitting set algorithm consists of $t = \lceil \log n \rceil + 1$ phases. Initialize $R_1 = A$.

In phase $i \geq 1$, run Find-OV on $(2^{i-1}R_i, B)$, where $2^i R_i$ is the multiset with 2^i copies of each vector in R_i. Let $P \subseteq A$ be the output multiset and let P' be the corresponding set (removing duplicates). Set $R_{i+1} = R_i \setminus P'$. If $|R_{i+1}| > n/2^i$, report failure (i.e. there is a hitting vector). Otherwise, proceed to the next phase. If phase t is complete, report success (i.e. no hitting vector).

Suppose that the algorithm reports success. Then after phase t, we have $R_{t+1} \leq n/2^t < 1$. Then for every $a \in A$ there was some phase i in which a was removed from R_i, and therefore was orthogonal to some $b \in B$. So there is no hitting vector.

Suppose that the algorithm reports failure in phase i. Then $|R_i| \leq n/2^{i-1}$ and $|R_{i+1}| > n/2^i$, so $|P'| < n/2^i$. Therefore $|P| \leq 2^{i-1}|P'| < n/2$. By the Find-OV approximation guarantee, not every element of R_i is orthogonal to an element of B. So there is a hitting vector.

The time complexity is dominated by $O(\log n)$ applications of Find-OV on inputs of size $O(n)$, along with $O(nd)$ extra processing in each phase. Thus, the time complexity is $O((T(n,d) + nd)\log n)$. ◀

The next theorem shows that hardness for approximate EMD matching (conditioned on the Hitting Set Conjecture) follows from chaining together the above reductions.

▶ **Theorem 16.** *If there are any $\epsilon, \delta > 0$ such that $(1+1/n^\delta)$-approximate EMD matching can be solved in $O(n^{2-\epsilon})$ time for some dimension $d = \omega(\log n)$, then the Hitting Set Conjecture is false.*

Proof. Fix $d = \omega(\log n)$, and assume without loss of generality that $d(n)$ is polylogarithmic. Let $\epsilon, \delta > 0$ and suppose that $(1+1/n^\delta)$-approximate EMD matching can be solved in $O(n^{2-\epsilon})$ time. Then $(1 + 1/n^\delta)$-approximate asymmetric EMD can be solved with an additional additive error of $n^{1-\delta}\sqrt{d}$ with the same time complexity, by Lemma 9. Hence, the Maximum Orthogonal Matching problem can be solved with an additive error of $n^{1-\delta}d$ in the same time, by Lemma 11.

Applying Theorem 14 with parameter $\alpha = \delta$, we get a randomized algorithm for Find-OV with an additive error of $O(n^{1-\delta^2}d^{1+\delta})$ and time complexity $O(n^{2-\gamma})$ for some $\gamma > 0$. For sufficiently large n, the error is at most $n/2$. Thus, we can apply Theorem 15 to get a randomized algorithm for Hitting Set with time complexity $\tilde{O}(n^{2-\gamma})$, which contradicts the Hitting Set Conjecture. ◀

Furthermore, we obtain stronger hardness of approximation for asymmetric EMD matching:

▶ **Theorem 17.** *Let $d = \omega(\log n)$ and $\eta = 1/\omega(\log n)$. If there is a truly subquadratic $(1+\eta)$-approximation algorithm for asymmetric EMD matching in d dimensions, then the Hitting Set Conjecture is false.*

Proof. Fix $d' = \omega(\log n)$ and $\eta = 1/\omega(\log n)$ and $\epsilon > 0$. Suppose that there is an $O(n^{2-\epsilon})$ time algorithm which achieves a $(1+\eta)$ approximation for asymmetric EMD matching in d' dimensions. Set $d = \min(d', \sqrt{(\log n)/\eta})$. Since $\mathbb{R}^d$ embeds isometrically in $\mathbb{R}^{d'}$, the algorithm also achieves a $(1+\eta)$ approximation for asymmetric EMD in d dimensions.

By Lemma 12, the Maximum Orthogonal Matching problem can be solved up to an additive error of $O(\eta|A|d)$ in $O(d)$ dimensions and $O(n^{2-\epsilon})$ time. By Theorem 14 there is some $\gamma > 0$ such that Find-OV can be solved up to an additive error of $O(\eta nd)$ in $O(d)$ dimensions and $O(n^{2-\gamma})$ time. By choice of d we have $\eta nd = o(n)$, so for sufficiently large n the algorithm achieves additive error of at most $n/2$. Therefore by Theorem 15, Hitting Set can be solved in $O(d)$ dimensions and $\tilde{O}(n^{2-\epsilon})$ time. Since $d = \omega(\log n)$, this contradicts the Hitting Set Conjecture. ◀

References

1. Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight Hardness Results for LCS and Other Sequence Similarity Measures. In *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, FOCS '15, pages 59–78, Washington, DC, USA, 2015. IEEE Computer Society. doi:10.1109/FOCS.2015.14.
2. Amir Abboud, Virginia Vassilevska Williams, and Joshua Wang. Approximation and Fixed Parameter Subquadratic Algorithms for Radius and Diameter in Sparse Graphs. In *Proceedings of the Twenty-seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 377–391, Philadelphia, PA, USA, 2016. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2884435.2884463.
3. Pankaj K. Agarwal, Kyle Fox, Debmalya Panigrahi, Kasturi R. Varadarajan, and Allen Xiao. Faster Algorithms for the Geometric Transportation Problem. In Boris Aronov and Matthew J. Katz, editors, *33rd International Symposium on Computational Geometry (SoCG 2017)*, volume 77 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:16, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.SoCG.2017.7.
4. Josh Alman, Timothy M. Chan, and Ryan Williams. Polynomial Representations of Threshold Functions and Algorithmic Applications. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 467–476, October 2016. doi:10.1109/FOCS.2016.57.
5. Jason Altschuler, Jonathan Weed, and Philippe Rigollet. Near-linear Time Approximation Algorithms for Optimal Transport via Sinkhorn Iteration. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 1961–1971, USA, 2017. Curran Associates Inc. URL: http://dl.acm.org/citation.cfm?id=3294771.3294958.
6. Alexandr Andoni, Aleksandar Nikolov, Krzysztof Onak, and Grigory Yaroslavtsev. Parallel Algorithms for Geometric Graph Problems. In *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*, STOC '14, pages 574–583, New York, NY, USA, 2014. ACM. doi:10.1145/2591796.2591805.
7. Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint*, 2017.
8. Arturs Backurs and Piotr Indyk. Edit Distance Cannot Be Computed in Strongly Subquadratic Time (Unless SETH is False). In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 51–58, New York, NY, USA, 2015. ACM. doi:10.1145/2746539.2746612.
9. Amitabh Basu. Open Problem: Maximum weighted assignment problem. In *Workshop: Combinatorial Optimization*, Oberwolfach Report 50/2018, page 44, 2018. doi:10.4171/OWR/2018/50.
10. Nicolas Bonneel, Michiel van de Panne, Sylvain Paris, and Wolfgang Heidrich. Displacement Interpolation Using Lagrangian Mass Transport. *ACM Transactions on Graphics*, 30(6):158:1–158:12, December 2011. doi:10.1145/2070781.2024192.
11. Karl Bringmann and Marvin Kunnemann. Quadratic Conditional Lower Bounds for String Problems and Dynamic Time Warping. In *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, FOCS '15, pages 79–97, Washington, DC, USA, 2015. IEEE Computer Society. doi:10.1109/FOCS.2015.15.
12. Lijie Chen. On the Hardness of Approximate and Exact (Bichromatic) Maximum Inner Product. In *Proceedings of the 33rd Computational Complexity Conference*, CCC '18, pages 14:1–14:45, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2018.14.
13. Rémi Flamary, Marco Cuturi, Nicolas Courty, and Alain Rakotomamonjy. Wasserstein Discriminant Analysis. *Machine Learning*, 107(12):1923–1945, December 2018. doi:10.1007/s10994-018-5717-1.
14. John Hopcroft and Richard Karp. An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973. doi:10.1137/0202019.

15 Piotr Indyk. A Near Linear Time Constant Factor Approximation for Euclidean Bichromatic Matching (Cost). In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 39–42, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1283383.1283388.

16 Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM. doi:10.1145/276698.276876.

17 Andrey Boris Khesin, Aleksandar Nikolov, and Dmitry Paramonov. Preconditioning for the Geometric Transportation Problem. In Gill Barequet and Yusu Wang, editors, *35th International Symposium on Computational Geometry (SoCG 2019)*, volume 129 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.SoCG.2019.15.

18 Yin Tat Lee and Aaron Sidford. Path Finding II: An $\tilde{O}(m\sqrt{n})$ Algorithm for the Minimum Cost Flow Problem. *arXiv preprint*, 2013. arXiv:1312.6713.

19 Yin Tat Lee and Aaron Sidford. Path Finding Methods for Linear Programming: Solving Linear Programs in Õ(Vrank) Iterations and Faster Algorithms for Maximum Flow. In *Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, FOCS '14, pages 424–433, Washington, DC, USA, 2014. IEEE Computer Society. doi:10.1109/FOCS.2014.52.

20 Jonas Mueller and Tommi Jaakkola. Principal Differences Analysis: Interpretable Characterization of Differences Between Distributions. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 1702–1710, Cambridge, MA, USA, 2015. MIT Press. URL: http://dl.acm.org/citation.cfm?id=2969239.2969429.

21 Aviad Rubinstein. Hardness of Approximate Nearest Neighbor Search. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 1260–1268, New York, NY, USA, 2018. ACM. doi:10.1145/3188745.3188916.

22 Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision*, 40(2):99–121, November 2000. doi:10.1023/A:1026543900054.

23 Roman Sandler and Michael Lindenbaum. Nonnegative Matrix Factorization with Earth Mover's Distance Metric for Image Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1590–1602, August 2011. doi:10.1109/TPAMI.2011.18.

24 R. Sharathkumar and Pankaj K. Agarwal. A Near-linear Time ϵ-approximation Algorithm for Geometric Bipartite Matching. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pages 385–394, New York, NY, USA, 2012. ACM. doi:10.1145/2213977.2214014.

25 Justin Solomon, Fernando de Goes, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. Convolutional Wasserstein Distances: Efficient Optimal Transportation on Geometric Domains. *ACM Transactions on Graphics*, 34(4):66:1–66:11, July 2015. doi:10.1145/2766963.

26 Cédric Villani. *Topics in optimal transportation*. Number 58 in Graduate Studies in Mathematics. American Mathematical Society, 2003.

27 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2):357–365, 2005. Automata, Languages and Programming: Algorithms and Complexity (ICALP-A 2004). doi:10.1016/j.tcs.2005.09.023.

28 Ryan Williams. On the Difference Between Closest, Furthest, and Orthogonal Pairs: Nearly-linear vs Barely-subquadratic Complexity. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1207–1215, Philadelphia, PA, USA, 2018. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=3174304.3175348.

29 Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the ICM*, 2018.

A Hardness of Low-Rank Minimum Weighted Assignment

The methods we used to prove hardness of exact EMD in low dimensions can be adapted to prove hardness of minimum weighted assignment with low-rank weight matrices, under the Orthogonal Vectors Conjecture. In particular, we show in the following theorem that bichromatic closest pair in d dimensions can be reduced to minimum weighted assignment with a rank-$O(d)$ weight matrix. The reduction algorithm uses the same input transformation as Theorem 8, and then solves minimum weighted assignment on the matrix M with entries $M_{ij} = \left\|A'_i - B'_j\right\|_2^2$, where A' and B' are the transformed input sets. The key is that M has rank $O(d)$, and its minimum weight assignment encodes the squared closest pair distance of the input – just as the EMD of the transformed input in Theorem 8 encoded the closest pair distance of the input.

▶ **Theorem 18.** *Fix a dimension $d = d(n) \leq n$, and let $\epsilon > 0$. Suppose that there is an algorithm which solves minimum weighted assignment in $O(n^{2-\epsilon})$ time, if the weight matrix has rank at most $O(d)$. Then bichromatic closest pair in d dimensions can be solved in $O(n^{2-\epsilon})$ time.*

Proof. Let A and B be two sets of n vectors in d dimensions, with entries in $\{1, \ldots, n^k\}$ for some constant $k > 0$. Apply the transformation described in Theorem 8 to construct sets $A', B' \in \{0, \ldots, n^{16k}\}^{2d+2c+2}$ where c is as defined in the proof of the theorem. Define

$$\text{SQEMD}(A', B') = \min_{\sigma: A' \to B'} \sum_{a' \in A'} \|a' - \sigma(a')\|_2^2$$

where σ ranges over all bijections from A' to B'. Since $\|u - v\|_2^2 \geq N^2 d$ and $\|a' - b'\|_2^2 \geq N^2 d$ for every $a' \in A' \setminus \{v\}$ and $b' \in B' \setminus \{u\}$, whereas $\|a' - u\|_2^2 \ll N^2 d/n$ and $\|b' - v\|_2^2 \ll N^2 d/n$, the optimal matching σ minimizes the number of (u, v) and (a', b') edges. In particular, exactly one element of $A' \setminus \{v\}$ is matched to an element of $B' \setminus \{u\}$. Thus, paralleling the proof of Theorem 8, we get

$$\text{SQEMD}(A', B') = 2(n-1)n^{4k}d^2 + \left(N^2 d + 2n^{4k}d^2 + \min_{a \in A, b \in B} \|a - b\|_2^2\right).$$

Hence, to compute the bichromatic closest pair distance between A and B, it suffices to compute $\text{SQEMD}(A', B')$. Representing A' and B' as $n \times (2d + 2c + 2)$ matrices, let M be the $n \times n$ matrix defined by $M_{ij} = \left\|A'_i - B'_j\right\|_2^2$. Then observing that

$$M_{ij} = \sum_{k=1}^{2d+2c+2} (A'_{ik} - B'_{jk})^2 = \sum_{k=1}^{2d+2c+2} (A'_{ik})^2 + \sum_{k=1}^{2d+2c+2} (B'_{jk})^2 - 2 \sum_{k=1}^{2d+2c+2} A'_{ik} B'_{jk},$$

we can write M as the sum of $2d + 2c + 4$ rank-1 matrices, so $\text{rank}(M) \leq 2d + 2c + 4$. So by assumption, the minimum weight perfect matching in the complete bipartite graph determined by M can be found in $O(n^{2-\epsilon}\text{poly}(d))$ time. But the cost of the optimal matching is precisely $\text{SQEMD}(A', B')$. ◀

Applying Theorem 6 completes the proof of Theorem 2.

B Proof of Theorem 13

The theorem follows immediately from the reduction from Maximum Orthogonal Matching to EMD matching shown in section 4, and this next proposition.

▶ **Proposition 19.** *Suppose the Maximum Orthogonal Matching problem can be solved up to an additive factor of n^δ in $O(n^\gamma)$ time where $\delta < 1/2$. Then OV can be solved in $O(n^{\gamma/(1-\delta)})$ time.*

Proof. Let $A, B \subseteq \{0,1\}^d$ with $|A| = |B| = n$. We construct multisets A' and B' which consist of $2n^{\delta/(1-\delta)}$ copies of each $a \in A$, and $2n^{\delta/(1-\delta)}$ copies of each $b \in B$, respectively. We then run our approximate Maximum Orthogonal Matching algorithm on A' and B'. If any orthogonal pair is found, we return it; otherwise we return that there is no orthogonal pair.

Since $|A'| = |B'| = 2n^{1/(1-\delta)}$, the time complexity of this algorithm is $O(n^{\gamma/(1-\delta)})$. It is clear that if A and B have no orthogonal pair, then A' and B' have no orthogonal pair, so the algorithm correctly returns "no pair".

Suppose that there are $a \in A$ and $b \in B$ with $a \cdot b = 0$ but the algorithm returns "no pair". Then the matching found by the algorithm had no orthogonal pairs. However, there is a matching consisting of $2n^{\delta/(1-\delta)}$ pairs. Since $|B'|^\delta < 2n^{\delta/(1-\delta)}$, this contradicts the approximation guarantee of the Maximum Orthogonal Matching algorithm. ◀

C Hardness of $(k, 2k)$-Find-OV

The $(k, 2k)$-Find-OV problem provides some sense of the relative "powers" of the Orthogonal Vectors Conjecture and the Hitting Set Conjecture, as well as another example of how the Hitting Set Conjecture can be used to explain hardness of approximation problems. Reducing from OV, we get the following hardness result, and it is not clear how to make any improvement. Note that this proof extends to the $(1, 2k)$-Find-OV problem, for which this lower bound is tight, due to a random sampling algorithm.

▶ **Proposition 20.** *Fix $\delta \in (0,1)$. Assuming OVC, any algorithm for $(n^\delta, 2n^\delta)$-Find-OV requires $\Omega(n^{2-\delta-o(1)})$ time.*

Proof. Suppose that there exists an $O(n^{2-\delta-\epsilon})$ time algorithm FIND for $(n^\delta, 2n^\delta)$-Find-OV. Here is an algorithm for OV: given sets $A, B \subseteq \{0,1\}^d$ with $|A| = |B| = n$, duplicate each $a \in A$ and each $b \in B$ exactly $2n^{\delta/(2-\delta)}$ times. If the original number of orthogonal pairs was r, then the new number is $4rn^{2\delta/(2-\delta)}$. For $r \geq 1$, this exceeds $2(n \cdot 2n^{\delta/(2-\delta)})^\delta$, so applying FIND yields a positive number of orthogonal vectors if and only if $r > 0$. It's easy to check that the time complexity is subquadratic. ◀

On the other hand, under the Hitting Set Conjecture, we can obtain quadratic hardness. When $k = n/2$, hardness follows from Theorem 15, but it holds in greater generality. In particular, we provide a proof of conditional hardness for $k = \sqrt{n}$, and it extends naturally to any $k = n^\gamma$ for $\gamma \in (0,1)$. The proof takes inspiration from the reduction from Hitting Set to OV [2], with a few extra twists.

▶ **Theorem 21.** *If the $(\sqrt{n}, 2\sqrt{n})$-Find-OV problem can be solved in $O(n^{2-\epsilon})$ time for some $\epsilon > 0$, then Hitting Set can be solved in $O(n^{2-\delta})$ time for some $\delta > 0$.*

Proof. Let FIND be the presupposed algorithm for $(\sqrt{n}, 2\sqrt{n})$-Find-OV. Set $\alpha = \epsilon/7$. Let $A, B \subseteq \{0,1\}^d$ with $|A| = |B| = n$. Without loss of generality, assume that no vector is all-zeroes. Here is an algorithm:

1. For each $a \in A$, randomly sample $n^{1-\alpha}$ vectors from B. If any of these is orthogonal to a, mark a and remove it from A, replacing it with an all-ones vector.
2. Set $k = n^{1/3-\alpha}$. Partition A into sets $A_1, \ldots, A_k$ of approximately equal size, and similarly partition B into sets $B_1, \ldots, B_k$. For each pair (A_i, B_j):

   a. Apply FIND to (A_i, B_j).
   b. If the output is not $\sqrt{n/k}$ orthogonal pairs, then continue to the next pair (A_i, B_j).
   c. Otherwise, suppose that the output is $\{(a_m, b_m)\}_{m=1}^{\sqrt{n/k}}$. For each vector $a \in \{a_m\}_{m=1}^{\sqrt{n/k}}$, mark a and remove it from A_i (and from A), replacing it with an all-ones vector.
   d. Go to (a).
3. If the number of unmarked input vectors exceeds $2n^{1-3\alpha/2}$, return "NO" and exit.
4. For each $a \in A$, if a is not the all-ones vector, iterate over all $b \in B$, and mark a if any $b \in B$ is orthogonal.
5. Return "YES" if every vector originally in A is now marked, and "NO" otherwise.

We claim that this algorithm solves Hitting Set in strongly subquadratic time. Correctness is relatively simple: a vector $a \in A$ is only marked by the above algorithm if some $b \in B$ is found for which $a \cdot b = 0$. Thus, if some $a \in A$ is a hitting vector for B, then it is never marked, so the algorithm returns "NO".

Conversely, suppose that every $a \in A$ is orthogonal to some $b \in B$. Then the number of unmarked input vectors in Step 3 is at most the number of remaining orthogonal pairs. But each (A_i, B_j) contains at most $2\sqrt{n/k}$ orthogonal pairs after Step 2 finishes, so the number of remaining orthogonal pairs in Step 3 is at most $k^2(2\sqrt{n/k}) = 2n^{1-3\alpha/2}$. Thus, the algorithm continues to Step 4. Every $a \in A$ which has not been marked by the end of Step 2 is tested against every $b \in B$ in Step 4. Therefore every vector is marked, so the algorithm returns "YES".

Turning to time complexity, Step 1 takes $O(n^{2-\alpha})$ time. The complexity of Step 2 is dominated by the calls to FIND. For each pair (A_i, B_j) there is at most one call to FIND for which the output is not $\sqrt{n/k}$ orthogonal pairs. Hence, there are $k^2 = n^{2/3-2\alpha}$ such "failed" calls. To bound the number of "successful" calls to FIND, for which the output *is* $\sqrt{n/k}$ orthogonal pairs, note that after Step 1, with high probability each $a \in A$ is orthogonal to at most $n^{2\alpha}$ vectors $b \in B$, so the total number of orthogonal pairs is at most $n^{1+2\alpha}$. Each successful call eliminates $\sqrt{n/k} = n^{1/3+\alpha/2}$ orthogonal pairs, so there are at most $n^{2/3+3\alpha/2}$ successful calls. This bound dominates the bound on failed calls. Each call takes time $O((n/k)^{2-\epsilon})$, so the time complexity of Step 2 is asymptotically

$$n^{\left(\frac{2}{3}+\alpha\right)(2-\epsilon)} n^{\frac{2}{3}+\frac{3\alpha}{2}} = n^{2-\frac{\epsilon}{6}-\frac{\epsilon^2}{7}}.$$

Step 3 takes negligible time. Finally, in Step 4, there are at most $2n^{1-3\alpha/2}$ vectors $a \in A$ which are not the all-ones vector (since each of these is unmarked), so the complexity is $O(n^{2-3\alpha/2})$.

Hence, the overall time complexity is bounded by $O(n^{2-\epsilon/7})$. ◀

Single-Elimination Brackets Fail to Approximate Copeland Winner

Reyna Hulett
Department of Computer Science, Stanford University, CA, USA
rmhulett@stanford.edu

Abstract

Single-elimination (SE) brackets appear commonly in both sports tournaments and the voting theory literature. In certain tournament models, they have been shown to select the unambiguously-strongest competitor with optimum probability. By contrast, we reevaluate SE brackets through the lens of approximation, where the goal is to select a winner who would beat the most other competitors in a round robin (i.e., maximize the Copeland score), and find them lacking. Our primary result establishes the approximation ratio of a randomly-seeded SE bracket is $2^{-\Theta(\sqrt{\log n})}$; this is underwhelming considering a $\frac{1}{2}$ ratio is achieved by choosing a winner uniformly at random. We also establish that a generalized version of the SE bracket performs nearly as poorly, with an approximation ratio of $2^{-\Omega(\sqrt[4]{\log n})}$, addressing a decade-old open question in the voting tree literature.

2012 ACM Subject Classification Theory of computation → Solution concepts in game theory; Mathematics of computing → Approximation algorithms

Keywords and phrases Voting theory, mechanism design, query complexity, approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.13

Category APPROX

Funding Supported in part by an NSF Graduate Research Fellowship under grant DGE-1656518.

Acknowledgements I want to thank Benjamin Plaut and Mary Wootters for many helpful discussions.

1 Introduction

The *round robin* and the *single-elimination bracket* are two common formats for sporting competitions. In a round robin, every competitor plays against every other competitor once. The outcome of a round robin can be represented as a *tournament graph*, a directed complete graph where an edge from A to B means A defeats B. A single-elimination bracket can be represented by a balanced binary tree with the leaves labeled by a permutation of the competitors. Each internal node is then labeled with the winner of a game between the two children of that node, with the root node indicating the overall winner. (For simplicity, assume no ties, deterministic game outcomes, and $n = 2^m$ competitors for some integer $m \geq 2$.)

A round robin effectively gives us complete information; we learn the outcome of all $\binom{n}{2}$ possible games. However, it is not immediately clear how to translate this into a single winner unless one competitor beats every other competitor (known as a *Condorcet winner*). There are various possible solution concepts – such as the Slater set, the uncovered set, and the top cycle – but we will focus on the (far more popular) Copeland solution. Each competitor's *Copeland score* equals its out-degree in the tournament graph, i.e., the number of other competitors it defeats. This gives us a natural, quantitative measure of competitor strength; thus the Copeland winner(s), or Copeland set, is the competitor(s) with the maximum Copeland score.

An SE bracket leaves no such ambiguity in determining a unique winner. It also requires fewer games, and each game has higher stakes, which may explain the popularity of this format! But what are we trading off in exchange for these desirable qualities? Can we still expect a strong competitor to win? We will address this question by considering how well SE brackets approximate the maximum Copeland score, for both worst-case and random seeding.


© Reyna Hulett;
licensed under Creative Commons License CC-BY

1.1 Related Work

A slightly different version of this question is long resolved. Namely, that line of work assumes the game outcomes are probabilistic *but* there exists an unambiguously strongest competitor (who beats every other competitor with probability $> 1/2$). Competition formats can then be evaluated based on their probability of selecting this strongest competitor, relative to the number of games or rounds played. This evaluation criteria has variously been referred to as "predictive power" or "effectiveness" [12, 1, 14]. Under certain models in this setting, a balanced SE bracket has the highest predictive power of any competition format with at most $n-1$ games [12]. However, the predictive power of *any* knockout format (where a competitor is eliminated after a single loss) will in general be sub-constant in the number of competitors. This is one motivation for evaluating formats based on the expected "strength" of the winner, rather than just the (vanishing) probability of selecting a single strongest competitor.

The question of *seeding* an SE bracket has also received significant attention in the probabilistic setting, both in terms of *designing* a fair seeding [21] and *manipulating* the seeding to help a particular competitor [14, 20] – in general, it is NP-hard to find a seeding which maximizes a given competitor's win probability. Perhaps surprisingly, it is even NP-hard to determine whether there exists a winning seeding for a given competitor with *deterministic* game outcomes [2], although many special cases have been identified where a polynomial time algorithm exists [19, 16, 18, 17, 2, 10, 9]. We ignore questions of seeding in the present work, considering only worst-case and random seeding. However, it is worth noting that a winning seeding *does* exist for any Copeland winner [19].

Finding or approximating the Copeland winner(s) of a tournament graph with deterministic game outcomes has been studied more generally, although not specifically for SE brackets. Finding the whole Copeland set requires all $\binom{n}{2}$ games to be played in the worst-case [4, 7]. If we only wish to select a single Copeland winner, that still requires at least $\binom{n}{2} - 2$ games (for odd n) [7]. By contrast, finding the Condorcet winner (or determining none exists) requires only $2n - \lfloor \log n \rfloor - 2$ games to be played [3, 13]. The number of games required to *approximate* the Copeland winner is not well studied, but it is known that finding a competitor with Copeland score *exactly* $k \leq (n-1)/2$ requires $\Theta(nk)$ games to be played in the worst-case [3].

The most direct predecessor of the present work concerns approximating the Copeland winner using a broad category of tournament formats known as *voting trees*, which include SE brackets. Whereas an SE bracket corresponds to a balanced binary tree with n leaves, a voting tree can be any binary tree with any number of leaves (the same competitor can label multiple leaves). For $n \leq 7$ competitors, there exists a voting tree which can select a Copeland winner, but not so for 8 or more competitors [15]. However, voting trees can *approximate* the maximum Copeland score, with an approximation ratio of $2/3$ [8], although this result is non-constructive. The best-known upper bound on the approximation ratio achievable by voting trees is $3/4$ [5, 6]. The situation changes slightly if, instead of a single voting tree, we are allowed to specify a distribution over voting trees. For instance, we could consider a randomly-seeded SE bracket. In this case, we want the *expected* Copeland score of the winner to approximate the maximum Copeland score. Naturally, the $2/3$ lower bound still applies, but the best-known upper bound for randomized voting trees is $5/6$ [5]. In additional, for randomized voting trees there is a constructive lower bound with certain nice properties which obtains an approximation ratio approaching $1/2$ [5]. The relevant paper concludes by conjecturing that SE brackets, or certain sizes of balanced voting trees more generally, may be able to obtain good approximation ratios, although they note "[t]he analysis of this type of randomization is closely related to the theory of dynamical systems, and we expect it to be rather involved" [5]. We answer this conjecture in the negative.

It is worth noting that randomly-seeded single-elimination brackets have previously been assumed to select "strong" winners. The probability of a competitor winning such a bracket has variously been referred to as "a natural notion of player strength" [10], and proposed as a way to select a winner from a tournament graph [11]. This makes it especially surprising that SE brackets fail to approximate the maximum Copeland score.

1.2 Contributions

We divide our contributions into three primary categories.

In Section 3, we analyze SE brackets with worst-case seeding. Although it will be straightforward to see that they achieve an approximation ratio of only $\frac{\log n}{n-2}$, we provide context for this result by calculating the query complexity (number of games that must be played) to approximate the maximum Copeland score with a given approximation ratio. We argue that, for worst-case seeding, SE brackets can actually be considered optimal among formats with at most $n-1$ games satisfying a basic fairness criterion, in close analogy to the work of [12]. Additionally, our results suggest a "single-elimination into round robin" format as an optimal generalization of SE brackets to more than n games.

Our main result is described in Section 4. Namely, we establish that the approximation ratio of a randomly-seeded SE bracket is $2^{-\Theta(\sqrt{\log n})}$. (For comparision, this is not as bad as the worst non-zero approximation ratio $\Theta(\frac{1}{n})$ but is worse than, say, $\Theta(\frac{1}{\log n})$.) A central lemma used in this proof is based on a result of [5], although we have to redo their analysis more precisely for our purposes. Essentially, we consider a class of tournament graphs where the competitors fall into three categories: "weak", "mediocre", and "strong". We construct a randomized distribution over such tournament graphs, which allows us to use the lemma to show that "weak" competitors usually win the corresponding SE brackets. Additionally, we show that most of the probability mass concentrates on tournament graphs where the "weak" competitors really do have low Copeland score, and thus there exists a specific tournament graph with low approximation ratio. We obtain the corresponding lower bound on the approximation ratio by showing that sufficiently low-scoring competitors are few in number and likely to be eliminated early in an SE bracket.

Finally, in Section 5, we reuse the aforementioned lemma to upper-bound the approximation ratio of *all* randomly-seeded, balanced voting trees of sufficient size as $2^{-\Omega(\sqrt[4]{\log n})}$. This refutes the hope expressed in [5] that carefully choosing the size of a balanced voting tree could result in a good approximation ratio.

2 Preliminaries

Let $S = \{1, \ldots, n = 2^m\}$ be a set of competitors. A *tournament graph*, or just *tournament*, on S is a directed complete graph where S is the vertex set. For any two competitors $c_1, c_2 \in S$, $c_1 \neq c_2$, the corresponding edge is directed from the winner to the loser in a hypothetical game between the two. If c_1 is the winner, we say c_1 beats c_2, or $c_1 \to c_2$. Note that this implies there are no ties, and the game outcomes are deterministic, i.e., if $c_1 \to c_2$, then c_1 beats c_2 always. We use $\mathcal{T}(n)$ to denote the set of all tournaments on n competitors.

Given a tournament graph T, the *Copeland score* of a competitor c is the out-degree of the corresponding vertex, i.e., $d_+(c) = |\{s \in S \mid c \to s\}|$. The *Copeland winner(s)* of a tournament is the competitor(s) with the highest Copeland score. If some competitor has Copeland score $n-1$, she is called the *Condorcet winner*.

We will be considering various *competition formats*, which are (deterministic or randomized) algorithms that query edges of the tournament graph by running games between pairs of competitors, and return a single winner. For a competition format F, the *query complexity* is the worst-case number of games played under F. For a given set of n competitors S, we also define the *approximation ratio* for the maximum Copeland score as

$$\min_{T \in \mathcal{T}(n)} \frac{\mathbb{E}[d_+(F(T))]}{\max_{s \in S} d_+(s)}$$

where $F(T)$ denotes the winner of tournament T under format F (possibly randomized).

A *single-elimination bracket* is a competition format represented by a balanced binary tree with $n = 2^m$ leaves labeled by a permutation or *seeding* of the competitors. For each level of the tree, moving up from the leaves – that is, for each round of the bracket – for each internal node, it labels the node with the winner of a game between the node's two children. The root node's label indicates the winner. We will analyze both worst-case seeding (i.e., the deterministic competition format with an arbitrary labeling of the leaves) and random seeding (i.e., the randomized competition format where a random permutation of competitors is chosen to label the leaves).

3 Deterministic Approximation of Copeland Winner

We first consider how well SE brackets with worst-case seeding approximate the Copeland winner, and for comparison, we establish the deterministic query complexity required for any competition format with a given approximation ratio. This reveals that SE brackets require significantly more games than the optimal query complexity. However, in Section 3.1, we reanalyze the query complexity of approximation subject to a basic fairness constraint, the "Condorcet property." Under this constraint, SE brackets actually achieve optimal query complexity, and can be generalized into an asymptotically optimal "single-elimination into round robin" format for more than $n-1$ games.

We begin by calculating the approximation ratio of SE brackets.

▶ **Theorem 1.** *The single-elimination bracket on $n = 2^m$ competitors achieves a deterministic approximation ratio of exactly $\frac{\log n}{n-2}$ for the maximum Copeland score.*

Proof. Observe that (1) the SE winner must have a Copeland score of at least $\log n$, since she must beat one competitor per level for $\log n$ levels, and (2) if a Condorcet winner exists, he must be the SE winner, or conversely, a non-SE-winner has Copeland score at most $n-2$. Taken together, these observations imply that the approximation ratio of the SE bracket is *at least* $\frac{\log n}{n-2}$.

Now, we construct a tournament graph showing that the approximation ratio is *at most* $\frac{\log n}{n-2}$. Consider any tournament graph G on $n-1$ competitors with a Condorcet winner, c. Let the n^{th} competitor be s. Fix any seeding for the bracket. In order to win the bracket, s must defeat $\log n$ competitors, whose identities are fully determined by the graph G and the seeding. Thus we may assume s defeats exactly this set of $\log n$ competitors, loses to every other competitor, and wins the bracket. By construction, s has Copeland score $\log n$ and the maximum Copeland score is (at least) $n-2$, so the approximation ratio is (at most) $\frac{\log n}{n-2}$, as desired. ◀

In order to evaluate how good or bad this approximation ratio is, we will consider how well an arbitrary competition format can approximate the maximum Copeland score, trading off against the total number of games played. This can be thought of as the query complexity

of approximation. I.e., if we are allowed arbitrary and adaptive "queries", how many games must be played (outcomes queried) to find a competitor with at least $0 < r \leq 1$ times the maximum Copeland score?

▶ **Theorem 2.** *The deterministic query complexity to find a competitor with at least* $0 < r \leq 1$ *times the maximum Copeland score in a tournament on* n *vertices is* $\Theta(\max(1, rn)^2)$.

Proof. We will use the fact that, for any tournament on n vertices, the maximum Copeland score M lies in $[\frac{n-1}{2}, n-1]$, i.e., between the average Copeland score and the maximum possible.

The upper bound is simple: to obtain an approximation ratio r, pick an arbitrary $n' = \min(2\lceil r(n-1)\rceil, n)$ competitors, and query all games within this sub-tournament. If $n' = n$, you find the true Copeland winner, so the approximation ratio is $1 \geq r$. Otherwise, $n' = 2\lceil r(n-1)\rceil$, so the average Copeland score within this sub-tournament is $\frac{2\lceil r(n-1)\rceil - 1}{2} = \lceil r(n-1)\rceil - \frac{1}{2}$. The maximum Copeland score within this sub-tournament is at least the average score rounded up, so at least $\lceil r(n-1)\rceil \geq r(n-1) \geq rM$, as desired. The total number of games played is at most

$$\binom{2\lceil r(n-1)\rceil}{2} \leq 2\lceil rn\rceil^2 = O(\max(1, rn)^2).$$

For the lower bound, we will define a simple adversarial strategy for answering the queries: when two competitors are queried, give the win to whichever has *fewer* wins so far, breaking ties arbitrarily. The competitor c returned by the optimal algorithm must have been shown to beat at least $k = \lceil \frac{r(n-1)}{2} \rceil$ distinct other competitors, since any un-queried game could be a loss for c. Now, because of the adversary's strategy, when the algorithm discovers the i^{th} win for c, it must be beating a competitor which already had $i-1$ wins queried. Thus, counting the number of wins for c and for the k competitors she defeats, the algorithm must have queried at least

$$k + \sum_{i=1}^{k}(i-1) = \frac{k(k+1)}{2} \geq \left\lceil \frac{r(n-1)}{2} \right\rceil^2 \div 2 = \Omega(\max(1, rn)^2)$$

games. ◀

As a sanity check, we know that finding a competitor with maximum Copeland score requires at least $\binom{n}{2} - 2$ games to be played (for odd n) [4], and indeed plugging in $r = 1$ we get a query complexity of $\Theta(n^2)$.

Numerically, SE brackets do not look very good at this point! The sub-tournament strategy described above can obtain the same approximation ratio $r = \frac{\log n}{n-2}$ in only $\Theta(\log^2 n) \ll n-1$ games. Conversely, if we allow ourselves $n-1$ games, we should be able to obtain an approximation ratio of $\Theta(\frac{1}{\sqrt{n}}) \gg \frac{\log n}{n-2}$.

Of course, this sub-tournament strategy is a deeply unsatisfying format for any kind of competition or election. 64 teams qualify for March Madness; should we suggest the NCAA just pick 12 at random and then play a round robin?

3.1 "Fair" Deterministic Approximation

There are multiple reasons why we might prefer an SE bracket over this strange sub-tournament round robin, but the most glaring is fairness. An SE bracket may be more or less "fair" depending on how the competitors are seeded, but at least it doesn't eliminate a majority of the competitors from the get-go.

In this section, we will investigate the complexity of approximation restricted to *Condorcet* competition formats. That is, if one competitor beats every other competitor in the underlying tournament graph (i.e., if there is a Condorcet winner) then he must be chosen as the winner of the competition. This is only a slightly stronger requirement than insisting that no competitor be eliminated a priori, since it specifies an intuitively obvious win condition. Additionally, this "Condorcet property" is well-studied in voting theory, has previously been described for tournaments under the name "unbiasedness" [12], and is closely related to the concept of "admissibility" from the voting tree literature [5]. Clearly, the SE bracket is a Condorcet competition format, while the sub-tournament round robin discussed above is not.

As a warm-up, observe that any Condorcet format must query at least $n-1$ game outcomes. This holds because every competitor except the winner must have been observed to lose at least one game; otherwise, a non-winner could violate the Condorcet property. Thus if we want to obtain an approximation ratio of $\frac{\log n}{n-2}$ for the maximum Copeland score, the SE bracket is optimal among Condorcet competition formats. In fact, the following result implies that SE brackets achieve the optimal approximation ratio among all Condorcet formats with exactly $n-1$ games.

▶ **Theorem 3.** *The deterministic query complexity, restricted to Condorcet competition formats, to find a competitor with at least $0 < r \leq 1$ times the maximum Copeland score in a tournament on $n = 2^m \geq 4$ vertices is $n-1$ if $r \leq \frac{\log n}{n-2}$ or $n-1+\Theta(\max(1, r(n-2)-\log n)^2)$ otherwise.*

Proof. As noted above, $n-1$ games are required for any Condorcet format, so when $r \leq \frac{\log n}{n-2}$, the desired approximation ratio is achieved in the optimal $n-1$ games by an SE bracket. For the remainder of the proof, we consider $r > \frac{\log n}{n-2}$.

The upper bound is similar to the sub-tournament round robin approach from Theorem 2, except that we use a partial single-elimination bracket to select the competitors for the sub-tournament. Define $\delta = \max(1, r(n-2) - \log n)$. If $\delta \geq \frac{n}{8}$, then we simply run a round robin on all n competitors and return the Copeland winner. This is clearly a Condorcet format, achieves approximation ratio $1 \geq r$, and requires $\binom{n}{2} \leq \frac{n^2}{2} = n - 1 + O(\delta^2)$ games.

Otherwise, we will play the first $\log n - \lceil \log \delta \rceil - 3$ rounds of a single-elimination bracket, then run a round robin among the remaining $2^{\lceil \log \delta \rceil + 3}$ competitors and return a competitor with highest number of wins. Observe that this is a Condorcet format, and it returns a competitor with Copeland score at least $\log n - \lceil \log \delta \rceil - 3 + 2^{\lceil \log \delta \rceil + 2} \geq \log n + 2^{\lceil \log \delta \rceil} \geq \log n + \delta \geq r(n-2)$. Therefore, we obtain an r-approximation of the maximum Copeland score, because either (1) there is a Condorcet winner, she wins the tournament, and the ratio is $1 \geq r$, or (2) there is no Condorcet winner, so the maximum Copeland score is $M \leq n-2$. Finally, the number of games played is less than

$$n - 1 + \binom{8\delta}{2} = n - 1 + O(\delta^2).$$

For the lower bound, we will reuse the adversarial strategy from Theorem 2: whenever a query is made, the winner of the game will be whichever competitor has *fewer* wins so far, with ties broken arbitrarily. Suppose the competitor c chosen as the winner of the competition has had k wins queried. Because of the adversary's strategy, these k competitors must have already had $0, 1, 2, \ldots, k-1$ wins (out-edges), respectively, at the time they were beaten. In fact, the same logic extends to these k competitors and their wins, etc., forming a cascade of 2^k vertices. However, these 2^k vertices need not necessarily be distinct competitors (except the top $k+1$, which must). Moreover, every vertex except the winner must have at

least one in-edge (otherwise it could be a Condorcet winner), so the number of vertex-reuses in this cascade is at most the query complexity less $n-1$, since every reuse increases the in-degree of some vertex by 1.

We would like to lower-bound the query complexity in terms of n; to do this, we will initially frame it as lower-bounding n in terms of the query complexity, for fixed k. Let $i \in \mathbb{R}_{\geq 0}$ be such that the total number of games played will be $n-1+\lceil i(i+1)/2 \rceil$. Since k is fixed, we need a cascade of 2^k vertices; however $\lceil i(i+1)/2 \rceil$ can be "reuses." Note that if a vertex is reused, it *and* its children appear only once in the resulting cascade. Any valid reuse can be captured by "erasing" a sub-tree, with the interpretation that the dangling edge that led to that sub-tree now points somewhere else. However, none of the top $k+1$ vertices can be erased in this way, since they must be distinct. (Other vertices can be erased and have their edges pointed to one of these vertices, however.)

Thus, to lower-bound n, we can equivalently upper-bound the number of vertices erased with $\lceil i(i+1)/2 \rceil$ reuses, since there need to be at least enough distinct vertices to constitute all the non-erased vertices in the cascade. How can we maximize the number of vertices erased? The top two layers (top $k+1$ vertices) cannot be erased, so the optimal strategy is to erase vertices from the layer directly below, in order of decreasing size of their sub-trees, since it always removes more vertices to erase the root of a sub-tree than any of its children. In particular, we would first erase the 3^{rd} level sub-tree of size 2^{k-2}, then the two sub-trees of size 2^{k-3}, then the three sub-trees of size 2^{k-4}, etc. For ease of accounting, let us assume we remove all the 3^{rd}-level sub-trees of size at least $2^{k-\lceil i \rceil-1}$. Observe that this comes to $1+2+\cdots+\lceil i \rceil \geq \lceil i(i+1)/2 \rceil$ reuses. We need to erase at least 2^k-n vertices from the cascade, so

$$\begin{aligned}
2^k - n &\leq 2^{k-2} + 2 \times 2^{k-3} + \cdots + \lceil i \rceil \times 2^{k-\lceil i \rceil-1} \\
&= 2^{k-\lceil i \rceil-1}\left(2^{\lceil i \rceil-1} + 2 \times 2^{\lceil i \rceil-2} + \cdots + \lceil i \rceil \times 2^0\right) \\
&= 2^{k-\lceil i \rceil-1}\left(2^{\lceil i \rceil+1} - \lceil i \rceil - 2\right) \\
&= 2^k - (\lceil i \rceil + 2)2^{k-\lceil i \rceil-1} \\
\log n &\geq \log(\lceil i \rceil + 2) + k - \lceil i \rceil - 1 \\
\lceil i \rceil &\geq k - \log n + \log(\lceil i \rceil + 2) - 1 \geq k - \log n.
\end{aligned}$$

Note that this means any Condorcet competition format using $n-1+\lceil i(i+1)/2 \rceil$ games returns a winner with at most $k \leq \log n + \lceil i \rceil$ wins queried. What does this mean for our approximation ratio? By the pigeonhole principle, there is some non-winner with no more than $1+\left\lfloor \frac{\lceil i(i+1)/2 \rceil}{n-1} \right\rfloor$ losses queried. Thus the maximum Copeland score could be as high as $M \geq n-2-\left\lfloor \frac{\lceil i(i+1)/2 \rceil}{n-1} \right\rfloor$. In particular, observe that if only $n-1$ games are played, then $i=0$ and $r = \frac{k}{M} \leq \frac{\log n}{n-2}$. This confirms that for $r > \frac{\log n}{n-2}$, $n-1+\Omega(1)$ games are required, implying SE brackets achieve the optimal approximation ratio for Condorcet formats with $n-1$ games.

We have essentially calculated a bound on the approximation ratio in terms of i, but we want to turn this into an asymptotic bound on query complexity for a given approximation ratio. Assuming $0 < i \leq n-1$ (since we can only have $\binom{n}{2}$ games in total), we have

$$
\begin{aligned}
r &\le \frac{\log n + \lceil i \rceil}{n - 2 - \left\lfloor \frac{\lceil i(i+1)/2 \rceil}{n-1} \right\rfloor} \\
r(n-2) - \log n &\le r \left\lfloor \frac{\lceil i(i+1)/2 \rceil}{n-1} \right\rfloor + \lceil i \rceil \\
(r(n-2) - \log n)^2 &\le \left(r \left\lfloor \frac{\lceil i(i+1)/2 \rceil}{n-1} \right\rfloor + \lceil i \rceil \right)^2 \\
&\le \lceil i(i+1)/2 \rceil^2 \left(\frac{1}{n-1} + \frac{\lceil i \rceil}{\lceil i(i+1)/2 \rceil} \right)^2 \\
&\le \lceil i(i+1)/2 \rceil^2 \left(\frac{1}{n-1} + \min\left(1, \frac{2}{i}\right) \right)^2 \\
&\le \lceil i(i+1)/2 \rceil^2 \left(2 \min\left(1, \frac{2}{i}\right) \right)^2 \\
&\le 4 \lceil i(i+1)/2 \rceil \left(\frac{i(i+1)}{2} + 1 \right) \min\left(1, \frac{4}{i^2}\right) \le 16 \lceil i(i+1)/2 \rceil .
\end{aligned}
$$

Thus the query complexity is at least

$$
n - 1 + \lceil i(i+1)/2 \rceil \ge n - 1 + \frac{1}{16}(r(n-2) - \log n)^2 = n - 1 + \Omega(\max(1, r(n-2) - \log n)^2)
$$

as desired. ◀

Interestingly, this implies that a "single-elimination into round robin" format achieves asymptotically optimal query complexity, with very simple structure. The initial single-elimination rounds could still benefit from seeding (to make stronger competitors more likely to survive the early rounds), while the round-robin phase ensures the eventual winner is reasonably strong, regardless of any manipulation in the seeding. Both single-elimination and round robin are common formats for sporting competitions, but they are rarely if ever employed together in this order.

In the following section, we move on to analyze SE brackets with random seeding (rather than worst-case). Note, however, that coming up with a good *randomized* approximation of the Copeland winner is much easier than the deterministic case considered above. In fact, we can achieve an approximation ratio of $r = \frac{1}{2}$ with a query complexity of *zero* – the average Copeland score is $\frac{n-1}{2}$, so simply returning a random competitor achieves this objective! This makes it especially surprising that randomly-seeded SE brackets cannot even achieve a constant approximation of the maximum Copeland score.

4 SE Brackets Fail to Approximate Copeland Winner

In this section, we prove our main result: the approximation ratio of SE brackets for the maximum Copeland score is $2^{-\Theta(\sqrt{\log n})}$.

To obtain our upper bound on the approximation ratio, we consider tournament graphs consisting of 3 groups ("components") of competitors: a small set of "strong" competitors, a small set of "weak" competitors, and a majority of "mediocre" competitors. We assume every strong competitor beats every mediocre competitor, who beats every weak competitor; however, the weak beat the strong (a related concept has been analyzed under the term "choking" [10]). Note that the weak competitors will have low Copeland scores, while the strong have high scores. The idea is that, even though weak competitors have low scores,

they can beat the strongest competitors in the tournament and thus come out on top. In fact, the key observation follows one made in Theorem 5 of [5]: as the depth of a balanced bracket grows, the likely winner oscillates between these three components.

The construction of this upper bound is similar to the proof of Theorem 5 in [5], with a few key differences. First, they consider a tournament with only a single "strong" and a single "weak" competitor, and label each leaf of the bracket *independently and uniformly at random*. Since an SE bracket must be labeled with a random *permutation* of the competitors, we instead have to construct a distribution over tournament graphs with varying numbers of strong and weak competitors in order to simulate each leaf being labeled independently. Because of this, even after showing that a "weak" competitor is likely to win the bracket, we have to prove this still holds when we restrict our distribution to tournaments where the "weak" competitors really have low Copeland score, and thus there exists some specific tournament graph where the SE bracket has a poor approximation ratio.

Second, [5] shows that the winner of a bracket oscillates between these three components, but does not establish the rate of oscillation. Because we need to show a weak competitor is likely to win after *precisely* $\log n$ rounds, we have to repeat their analysis with significant additional bookkeeping.

The result of this bookkeeping is the following lemma, analogous to Lemma G.1 from [5]. Roughly, it says: Suppose after some number of rounds of an SE bracket, practically all the remaining competitors come from the "strong" component. Nevertheless, after a specific number of additional rounds, practically all the remaining competitors will come from the *"weak"* component. Furthermore, the "weak" competitors continue to dominate for many rounds before the oscillation repeats. The proof consists of analyzing a simple recursive formula for the likelihood of a "strong", "mediocre", or "weak" player winning an SE bracket after k rounds, in order to give painstaking bounds on the magnitude and rate of oscillation of these probabilities.

The rather lengthy and unenlightening proof has been relegated to the appendix.

▶ **Lemma 4.** *Let S be a set of competitors partitioned into three components C_1, C_2, C_3 such that every member of component C_i beats every member of component $C_{(i \bmod 3)+1}$. Fix probabilities $p_i^{(0)}$ summing to 1, and let $p_i^{(k)}$ denote the probability that a member of component C_i wins a balanced bracket of height k where each leaf is labeled independently according to $p_i^{(0)}$. If for some $K \in \mathbb{N}$ and $0 < \epsilon \leq 2^{-10}$, $\epsilon^2 \leq p_3^{(K)} \leq \epsilon$ and $\epsilon \leq p_1^{(K)} \leq 2\epsilon$, then there exists $K + \log(\frac{1}{\epsilon}) \leq K' \leq K + 3\log(\frac{1}{\epsilon})$ and $\epsilon^{2\log(\frac{1}{\epsilon})} \leq \delta \leq \epsilon^{\log(\frac{1}{\epsilon})/4}$ such that $\delta^2 \leq p_2^{(K')} \leq \delta$ and $\delta \leq p_3^{(K')} \leq 2\delta$. Furthermore, if $\epsilon \leq 2^{-75}$ then for any $K'' \in [K', K' + 2^5 \log(\frac{1}{\epsilon})]$, $p_2^{(K'')}, p_3^{(K'')} \leq \epsilon^{\log(\frac{1}{\epsilon})/2^5}$.*

We are now ready to prove our upper bound on the approximation ratio for SE brackets.

▶ **Theorem 5.** *The approximation ratio of a randomly-seeded single-elimination bracket on $n = 2^m$ competitors for the maximum Copeland score is $O(2^{-\sqrt{\log(n)/7}})$.*

Proof. For any $n = 2^m$ with $\log n \geq 2^{12}$, pick $0 < \delta \leq 2^{-18}$ such that $7\log^2 \frac{1}{\delta} \leq \log n \leq 8\log^2 \frac{1}{\delta}$. Define a distribution over tournaments $D(n, \delta)$, with $p_s^{(0)} = p_w^{(0)} = \delta, p_m^{(0)} = 1 - 2\delta$. $D(n, \delta)$ is supported over tournaments of size n with three (possibly empty) components s, m, w where s beats m, m beats w, and w beats s. Internally each component is a regular tournament, meaning the difference between the maximum and minimum Copeland scores is 1 or 0. For any fixed size of the components, summing to n, the weight of the corresponding tournament in $D(n, \delta)$ is equal to the probability that those fixed sizes are achieved by assigning each of n competitors independently to a component according to $p_s^{(0)}, p_m^{(0)}, p_w^{(0)}$.

We will now analyze the winner of a bracket where each leaf is labeled independently with a component according to $p_s^{(0)}, p_m^{(0)}, p_w^{(0)}$. Observe that this is equivalent to choosing a random tournament according to $D(n,\delta)$ and then labeling n leaves with a random permutation of the competitors. In particular, for any given tournament graph in $D(n,\delta)$, every permutation of the competitors appears with equal probability.

Letting $C_1 = s, C_2 = m$, and $C_3 = w$ we apply Lemma 4. Since $0 < \delta \leq 2^{-10}$ and $\delta^2 \leq p_w^{(0)} \leq \delta \leq p_s^{(0)} \leq 2\delta$, we obtain that $(\delta')^2 \leq p_m^{(K)} \leq \delta' \leq p_w^{(K)} \leq 2\delta'$ for some $\delta^{2\log(\frac{1}{\delta})} \leq \delta' \leq \delta^{\log(\frac{1}{\delta})/4}$ and $\log(\frac{1}{\delta}) \leq K \leq 3\log(\frac{1}{\delta})$. We apply Lemma 4 once more, now with $C_1 = w, C_2 = s$, and $C_3 = m$ and starting from K, to find that a weak competitor wins with overwhelming probability after K' rounds, with

$$0 \leq K' \leq K + 3\log(\frac{1}{\delta'}) \leq 3\log(\frac{1}{\delta}) + 6\log^2(\frac{1}{\delta}).$$

We will use the final part of Lemma 4 to increase this to a bracket of depth $\log n$. Observe that no more than $8\log^2(\frac{1}{\delta})$, but more than 0, additional rounds are required. Furthermore, note that

$$\delta' \leq \delta^{\log(\frac{1}{\delta})/4} \leq 2^{-18^2/4} < 2^{-75}$$

as required for this part of the lemma. Finally, the number of additional rounds required is at most

$$8\log^2(\frac{1}{\delta}) \leq 2^5 \log(\frac{1}{\delta'})$$

and thus by Lemma 4, after $\log n$ rounds we have

$$p_s^{(\log n)}, p_m^{(\log n)} \leq {\delta'}^{\log(\frac{1}{\delta'})/2^5} \leq 2^{-\log^2(2^{\log^2(\delta)/4})/2^5} = 2^{-(\log^2(\delta)/4)^2)/2^5} = 2^{-\log^4(\delta)/2^9}$$

so also $p_3^{(\log n)} \geq 1 - 2 \times 2^{-\log^4(\delta)/2^9}$.

We have established the winning probability of a "weak" competitor, over distribution $D(n,\delta)$. However, some tournaments with non-zero weight in the distribution have "weak" competitors with high Copeland score (those in which either the weak or strong component is large). Next, we bound the probability that this happens in order to establish a distribution $D'(n,\delta)$, where a "weak" competitor still wins almost always *and* the weak competitors all have low Copeland score.

First, we separate out the high-scoring weak and low-scoring weak cases from $D(n,\delta)$, where $\Pr[w]$ represents the probability of a weak competitor winning:

$$\begin{aligned}\Pr[w] = \Pr[w : |w|, |s| < 10\delta n] \Pr[|w|, |s| < 10\delta n] + \\ \Pr[w : |w| \text{ or } |s| \geq 10\delta n] \Pr[|w| \text{ or } |s| \geq 10\delta n]\end{aligned}$$

Rearranging,

$$\begin{aligned}&\Pr[w \text{ wins} : |w|, |s| < 10\delta n] \\ &\quad \geq \Pr[w \text{ wins} : |w|, |s| < 10\delta n] \Pr[|w|, |s| < 10\delta n] \\ &\quad = \Pr[w \text{ wins}] - \Pr[w \text{ wins} : |w| \text{ or } |s| \geq 10\delta n] \Pr[|w| \text{ or } |s| \geq 10\delta n] \\ &\quad \geq 1 - 2 \times 2^{-\log^4(\delta)/2^9} - \Pr[|w| \text{ or } |s| \geq 10\delta n] \\ &\quad \geq 1 - 2 \times 2^{-\log^4(\delta)/2^9} - \Pr[|s| \geq 10\delta n] - \Pr[|w| \geq 10\delta n] \\ &\quad \geq 1 - 2 \times 2^{-\log^4(\delta)/2^9} - 2e^{-\frac{9\delta n}{3}} := p\end{aligned}$$

using the Chernoff bound $\Pr[|C| \geq 10\delta n] \leq e^{-\frac{9\delta n}{3}}$ (since the expectation of $|w|, |s|$ is δn).

Let $D'(n,\delta)$ equal $D(n,\delta)$ restricted to tournaments where $|w|,|s| < 10\delta n$. A weak competitor wins the SE bracket on a tournament graph drawn from $D'(n,\delta)$ with probability at least p.

Finally, we observe that the Copeland score of any member of the weak component of any tournament with non-zero support on $D'(n,\delta)$ is less than $\frac{3}{2}10\delta n$ (consisting of less than $10\delta n$ edges to the strong component and less than $10\delta n/2$ edges within the weak component). Thus the expected Copeland score of the winner of a randomly-seeded SE bracket over a tournament drawn from $D'(n,\delta)$ is less than

$$\begin{aligned}
&p\frac{30\delta n}{2} + (1-p)(n-1) \\
&\quad \leq 15\delta n + 2n \times 2^{-\log^4(\delta)/2^9} + 2ne^{-3\delta n};
\end{aligned}$$

recall $\log n \leq 8\log^2(\frac{1}{\delta})$, so $\delta \leq 2^{-\sqrt{\log(n)/8}}$:

$$\begin{aligned}
&\leq 15 \times 2^{\log n - \sqrt{\log(n)/8}} + 2n \times 2^{-\log^4(2^{-\sqrt{\log(n)/8}})/2^9} + 2ne^{-3\times 2^{\log n - \sqrt{\log(n)/8}}} \\
&\leq 15 \times 2^{\log n - \sqrt{\log(n)/8}} + 2n \times 2^{-(\sqrt{\log(n)/8})^4/2^9} + 2ne^{-3\times 2^{\log(n)/2}} \\
&\leq 15 \times 2^{\log n - \sqrt{\log(n)/8}} + 2 \times 2^{\log n - (\sqrt{\log(n)/8})^4/2^9} + 2ne^{-3\sqrt{n}} \\
&\leq 15 \times 2^{\log n - \sqrt{\log(n)/8}} + 2 \times 2^{\log n - \sqrt{\log(n)/8}} + 2
\end{aligned}$$

This implies that some *individual* tournament with non-zero support on $D'(n,\delta)$ achieves expected Copeland score at most $O(2^{\log n - \sqrt{\log(n)/8}})$. Thus the approximation ratio is $O(2^{-\sqrt{\log(n)/8}})$, completing the proof. ◀

In fact, the upper bound shown above is "nearly" tight, as the following theorem establishes.

▶ **Theorem 6.** *The approximation ratio of a randomly-seeded single-elimination bracket on $n = 2^m$ competitors for the maximum Copeland score is $\Omega(2^{-\sqrt{2\log n}})$.*

Proof. For any $k < n$, at most $2k$ competitors in the tournament can have Copeland score less than k – otherwise, the average score amongst these competitors alone would be at least $\frac{(2k+1)-1}{2} = k$, a contradiction. If k is sufficiently small, it becomes quite likely that these few competitors will be eliminated early in a randomly-seeded SE bracket.

We will capture this idea by union-bounding over the probability that an individual competitor c with Copeland score $d_+(c) < k$ survives $\left\lfloor \log(\frac{n}{k}) \right\rfloor + 1$ rounds. Each round, c must face one of the $< k$ competitors he can beat (not including those he has already beaten) – even assuming every other competitor he can beat advances. Therefore,

$$\begin{aligned}
&\Pr\left(c \text{ survives } \left\lfloor \log\left(\frac{n}{k}\right)\right\rfloor + 1 \text{ rounds} : d_+(c) < k\right) \\
&\quad \leq \prod_{i=0}^{\lfloor \log(\frac{n}{k})\rfloor} \frac{k-i-1}{\frac{n}{2^i}-1} \\
&\quad \leq \prod_{i=0}^{\lfloor \log(\frac{n}{k})\rfloor} 2^i \frac{k}{n} \\
&\quad \leq 2^{\frac{\log(\frac{n}{k})(\log(\frac{n}{k})+1)}{2}} \left(\frac{k}{n}\right)^{\log(\frac{n}{k})+1} \\
&\quad = \left(\frac{k}{n}\right)^{\frac{-1-\log(\frac{n}{k})}{2}} \left(\frac{k}{n}\right)^{\log(\frac{n}{k})+1} = \left(\frac{k}{n}\right)^{\frac{\log(\frac{n}{k})+1}{2}}
\end{aligned}$$

Even if we assume that every one of these low-scoring competitors wins with the probability calculated above, and contributes nothing to the expected Copeland score of the SE winner, we still know that the remaining probability belongs to competitors with Copeland score at least k. Thus,

$$\mathbb{E}[d_+(\text{winner})] \geq k\left(1 - 2k\left(\frac{k}{n}\right)^{\frac{\log(\frac{n}{k})+1}{2}}\right).$$

Let us plug in $k = n2^{-\sqrt{2\log n}} = 2^{\log n - \sqrt{2\log n}}$. Then,

$$\begin{aligned}
\mathbb{E}[d_+(\text{winner})] &\geq 2^{\log n-\sqrt{2\log n}}\left(1 - 2 \times 2^{\log n-\sqrt{2\log n}}\left(2^{-\sqrt{2\log n}}\right)^{\frac{\sqrt{2\log n}+1}{2}}\right)\\
&\geq 2^{\log n-\sqrt{2\log n}}\left(1 - 2 \times 2^{\log n-\sqrt{2\log n}} \times 2^{-\log n-\sqrt{\frac{\log n}{2}}}\right)\\
&\geq 2^{\log n-\sqrt{2\log n}}\left(1 - 2^{-3\sqrt{\frac{\log n}{2}}+1}\right)\\
&\geq n\left(2^{-\sqrt{2\log n}} - 2^{-5\sqrt{\frac{\log n}{2}}+1}\right) \geq n2^{-\sqrt{2\log n}-1}
\end{aligned}$$

which establishes that the approximation ratio is $\Omega(2^{-\sqrt{2\log n}})$, as desired. ◀

Taken together, these bounds establish that the approximation ratio of randomly-seeded SE brackets for the maximum Copeland score is $2^{-\Theta(\sqrt{\log n})}$.

5 Balanced Voting Trees Fail to Approximate Copeland Winner

In this section, we derive an upper bound on the approximation ratio for the maximum Copeland score of a generalized version of SE brackets from the voting tree literature. Recall that a voting tree is any binary tree with leaves labeled by the n competitors. The *randomized perfect voting tree* of depth k (k-RPT) is a class of voting trees introduced by [5], consisting of a balanced binary tree of depth k, with each leaf labeled uniformly at random from the set of n competitors. The k-RPT is similar to an SE bracket, except (1) the number of leaves may be larger (or smaller) than the number of competitors, and (2) the random seeding process does not require every competitor to appear on the leaves. However, as noted by [5], as k grows, the probability of any competitor not appearing on the leaves vanishes.

[5] established that, for infinitely many k, the k-RPT has an approximation ratio of $O(1/n)$ for the maximum Copeland score – essentially the worst possible! However, they left open the question of whether, for some carefully chosen $k = f(n)$, the k-RPT might achieve a good approximation ratio. We certainly shouldn't expect the approximation ratio to be as low as $O(1/n)$ for *every* k, since technically the 1-RPT corresponds to randomly choosing a winner and so has approximation ratio $\frac{1}{2}$, while the $(\log n)$-RPT is closely related to the SE bracket which has a ratio of $2^{-\Theta(\sqrt{\log n})}$. However, we can at least show that the approximation ratio of a k-RPT for any $k \geq \log n$ is sub-constant.

▶ **Theorem 7.** *The approximation ratio of the k-RPT with $k \geq \log n$ is $O(2^{-\sqrt[4]{\log n}/4})$*

Proof. We use the same tournament structure as in the previous section, with strong, mediocre, and weak components s, m, w. Because the labeling of leaves in a k-RPT is uniformly random, there is no need to define a distribution over such tournaments; the components have fixed proportions $p_s^{(0)}, p_m^{(0)}, p_w^{(0)}$ to be specified later.

We again make repeated use of Lemma 4. In this setting, however, we do not have arbitrary control over ϵ, the probability of a competitor being weak; it must initially be some integer multiple of $1/n$. Thus we will first establish, for any sufficiently small ϵ, an infinite set of ranges for which the probability of a weak competitor winning must be high. We will then argue that for sufficiently large n, we can vary $\epsilon = \ell/n$ enough that these ranges collectively cover every $k \geq \log n$.

Let $\epsilon_0 \leq 2^{-2^6}$ equal $p_w^{(0)}$, i.e., it will represent the fraction of competitors that are weak; note that this satisfies the requirement $\epsilon_0 \leq 2^{-10}$ for Lemma 4. Each time we apply Lemma 4, we will obtain a new ϵ_i, so for instance, $\epsilon_1 \in [2^{-2\log^2(\epsilon_0)}, 2^{-\log^2(\epsilon_0)/4}]$. We claim that

$$\log \epsilon_i \in \left[-2^{2^i-1} \left(\log \frac{1}{\epsilon_0} \right)^{2^i}, -\left(\frac{1}{4}\right)^{2^i-1} \left(\log \frac{1}{\epsilon_0} \right)^{2^i} \right]$$

We can verify this by induction – it clearly holds for $i = 0$. Assume it holds for $i-1$. By Lemma 4,

$$\begin{aligned}
\log \epsilon_i &\in \left[-2\log^2(\frac{1}{\epsilon_{i-1}}), -\frac{1}{4}\log^2(\frac{1}{\epsilon_{i-1}}) \right] \\
&\in \left[-2\left(-2^{2^{i-1}-1} \left(\log \frac{1}{\epsilon_0} \right)^{2^{i-1}} \right)^2, -\frac{1}{4}\left(-\left(\frac{1}{4}\right)^{2^{i-1}-1} \left(\log \frac{1}{\epsilon_0} \right)^{2^{i-1}} \right)^2 \right] \\
&\in \left[-2 \times 2^{2^i-2} \left(\log \frac{1}{\epsilon_0} \right)^{2^i}, -\frac{1}{4}\left(\frac{1}{4}\right)^{2^i-2} \left(\log \frac{1}{\epsilon_0} \right)^{2^i} \right],
\end{aligned}$$

as desired.

Let t_i be the step (bracket depth) at which the i^{th} application of Lemma 4 begins, $t_0 = 0$. Note that $p_w^{(t_i)}$ is high when $i \bmod 3 = 2$. We claim that

$$t_{i+1} \in \left[\log(\frac{1}{\epsilon_i}), 4\log(\frac{1}{\epsilon_i}) \right].$$

The lower bound is immediate from Lemma 4 because the i^{th} oscillation takes at least $\log(\frac{1}{\epsilon_i})$ steps. The upper bound we again prove inductively; for t_1 it likewise holds directly from Lemma 4. Assuming it holds for t_i, and since $\log \epsilon_i \in [-2\log^2(\epsilon_{i-1}), -\frac{1}{4}\log^2(\epsilon_{i-1})]$,

$$\begin{aligned}
t_{i+1} &\leq t_i + 3\log(\frac{1}{\epsilon_i}) \\
&\leq 4\log(\frac{1}{\epsilon_{i-1}}) + 3\log(\frac{1}{\epsilon_i}) \\
&\leq 4\sqrt{4\log(\frac{1}{\epsilon_i})} + 3\log(\frac{1}{\epsilon_i}) \\
&\leq 4\log(\frac{1}{\epsilon_i})
\end{aligned}$$

where the last line holds because $\log(\frac{1}{\epsilon_i}) \geq \log(\frac{1}{\epsilon_0}) \geq 2^6$ by assumption.

Next, recall that by Lemma 4, the two smaller probabilities at t_{i+1} sum to at most $3\epsilon_{i+1}$. If we allow this sum to increase slightly, say to ϵ_i, we can go additional steps beyond t_{i+1}. Specifically, knowing the probability at most doubles each time step, for any $t \leq \frac{1}{8}\log^2(\frac{1}{\epsilon_i})$ we have

$$\begin{aligned}\log\left(2^t \times 3\epsilon_{i+1}\right) &\leq t - 1 + \log(\epsilon_{i+1}) \\ &\leq \frac{1}{8}\log^2(\frac{1}{\epsilon_i}) - 1 - \frac{1}{4}\log^2(\frac{1}{\epsilon_i}) \\ &\leq -\log(\frac{1}{\epsilon_i})\end{aligned}$$

where the last line holds because $\log(\frac{1}{\epsilon_i}) \geq \log(\frac{1}{\epsilon_0}) \geq 2^6$. Removing the logarithm, the above implies that for any $t \leq \frac{1}{8}\log^2(\frac{1}{\epsilon_i})$, the two smaller probabilities at time t_{i+1} still sum to at most ϵ_i at time $t_{i+1} + \lfloor t \rfloor$. Also, for $t = \frac{1}{8}\log^2(\frac{1}{\epsilon_i})$, observe that

$$\begin{aligned}t_{i+1} &\leq 4\log(\frac{1}{\epsilon_i}) \leq 2\left(2\log\frac{1}{\epsilon_0}\right)^{2^i} \\ t_{i+1} + \lfloor t \rfloor &\geq \log(\frac{1}{\epsilon_i}) + \frac{1}{8}\log^2(\frac{1}{\epsilon_i}) - 1 \\ &\geq \frac{1}{8}\log^2(\frac{1}{\epsilon_i}) \geq 2\left(\frac{\log(1/\epsilon_0)}{4}\right)^{2^{i+1}}.\end{aligned}$$

Thus for any $\epsilon_0 \leq 2^{-2^6}$ and $i \in \mathbb{N}$, we have established an interval on which the largest probability is at least $1 - \epsilon_i$. In particular, whenever $i + 1 = 2 \pmod 3$, this gives an interval on which a weak competitor wins with overwhelming probability.

Next, we want to show that, for any sufficiently large n, these intervals can be made to cover every depth $k \geq \log n$, even with the limitation that ϵ_0 must equal ℓ/n for some $\ell \in \mathbb{N}$. In fact, we claim that letting ℓ take on values $1, 2, \ldots, \left\lceil 2^{\log n - \sqrt[4]{\log n}/4} \right\rceil := L$ covers every $k \geq \log n$ for any n sufficiently large that $\left\lceil 2^{\log n - \sqrt[4]{\log n}/4} \right\rceil / n \leq 2^{-2^6}$ – this is necessary to ensure that $\epsilon_0 \in [1/n, L/n]$ will be at most 2^{-2^6} as assumed above.

First, let us verify that the lowest k contained in one of these intervals is sufficiently small. k will be smallest when i is small ($i + 1 = 2$) and when ϵ_0 is large ($\epsilon_0 = L/n$). Thus the first interval will start at

$$2\left(2\log\frac{n}{L}\right)^2 \leq 2\left(\log(2^{\sqrt[4]{\log n}/4})\right)^2 = \sqrt{\log n}/8 \ll \log n,$$

so indeed our overlapping intervals start below $k = \log n$.

Next, we need to establish that for a fixed i, the adjacent intervals with $\epsilon_0 = \ell/n, \epsilon_0 = (\ell - 1)/n$ overlap. That is, the lower bound of the later interval needs to be below the upper bound of the earlier. I.e., we require

$$\begin{aligned}2\left(2\log(\frac{n}{\ell - 1})\right)^2 &\leq 2\left(\log(\frac{n}{\ell})/4\right)^4 \\ 2^5\log(\frac{n}{\ell - 1}) &\leq \log^2(\frac{n}{\ell})\end{aligned}$$

and indeed,

$$2^5\log(\frac{n}{\ell - 1}) \leq 2^5\left(1 + \log\frac{n}{\ell}\right) \leq \log^2(\frac{n}{\ell})$$

where the final inequality holds because $\log(\frac{n}{\ell}) \geq 2^6$.

Finally, we will show that the latest interval for i, $\epsilon_0 = 1/n$, overlaps with the earliest interval for $i+3$, $\epsilon_0 = L/n$, and thus that there is no gap between intervals where a weak competitor can win with high probability. We require

$$2\left(2\log\frac{n}{L}\right)^{2^{i+3}} \leq 2\left(\frac{\log n}{4}\right)^{2^{i+1}}$$
$$\left(2\log\frac{n}{L}\right)^{4} \leq \frac{\log n}{4}$$

and indeed,

$$\left(2\log\frac{n}{L}\right)^{4} \leq \left(2\sqrt[4]{\log n}/4\right)^{4} = \frac{\log n}{16} < \frac{\log n}{4}$$

as desired.

Having shown that a weak competitor can win with overwhelming probability for sufficiently large n and any $k \geq \log n$, we can upper-bound the expected Copeland score of the winner as

$$\mathbb{E}[d_+(\text{winner})] \leq (1-\epsilon_i)(\ell+\ell/2)+\epsilon_i(n-1) \leq L+L/2+\frac{L}{n}(n-1) \leq 3L = 3\left\lceil 2^{\log n - \sqrt[4]{\log n}/4}\right\rceil.$$

Equivalently, the approximation ratio of any k-RPT is $O(2^{-\sqrt[4]{\log n}/4})$. ◀

Interestingly, although this bound holds for all $k \geq \log n$, significantly tighter bounds can be obtained for certain k by the same method. In particular, the bound is made much looser by increasing the size of the weak component up to L/n, which is only necessary when trying to cover every possible k. When the weak component has size $1/n$, we recover the $O(1/n)$ approximation ratio of [5]. Without matching lower bounds, it is unclear to what extent this oscillating approximation ratio is real versus an artifact of the proof method. Regardless, this upper bound settles the question of whether any k-RPT can obtain a decent approximation ratio.

6 Conclusion

In this work, we establish that randomly-seeded single-elimination brackets are surprisingly bad at approximating the maximum Copeland score, as is a generalized version of SE brackets from the voting theory literature, the k-RPT. However, we show that SE brackets have optimal approximation ratio for *worst-case*/deterministic seeding among Condorcet competition formats.

Despite their sub-constant approximation ratio, single-elimination brackets are widely used; perhaps quirks of their occurrence in practice could improve the approximation ratio? For instance, one could consider the impact of seeding based on some measure of competitor ability, or investigate whether SE brackets perform better on tournament graphs generated from some random model. Alternatively, one could investigate other existing competition formats (e.g., double-elimination, Swiss-system) to see if they better approximate the Copeland winner.

References

1 Micah Adler, Peter Gemmell, Mor Harchol-Balter, Richard M. Karp, and Claire Kenyon. Selection in the Presence of Noise: The Design of Playoff Systems. In *Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '94, pages 564–572. Society for Industrial and Applied Mathematics, 1994. URL: `http://dl.acm.org/citation.cfm?id=314464.314650`.

2 Haris Aziz, Serge Gaspers, Simon Mackenzie, Nicholas Mattei, Paul Stursberg, and Toby Walsh. Fixing a Balanced Knockout Tournament. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, AAAI'14, pages 552–558. AAAI Press, 2014. URL: http://dl.acm.org/citation.cfm?id=2893873.2893960.

3 Ramachandran Balasubramanian, Venkatesh Raman, and G Srinivasaragavan. Finding Scores in Tournaments. *Journal of Algorithms*, 24(2):380–394, 1997. doi:10.1006/jagm.1997.0865.

4 Palash Dey. Query Complexity of Tournament Solutions. *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2992–2998, 2017. URL: https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14180.

5 Felix Fischer, Ariel D. Procaccia, and Alex Samorodnitsky. On Voting Caterpillars: Approximating Maximum Degree in a Tournament by Binary Trees. In Ulle Endriss and Paul W. Goldberg, editors, *Proceedings of the 2nd International Workshop on Computational Social Choice*, pages 253–264, 2008.

6 Felix Fischer, Ariel D. Procaccia, and Alex Samorodnitsky. A New Perspective on Implementation by Voting Trees. *Random Structures & Algorithms*, 39(1):59–82, 2011. doi:10.1002/rsa.20336.

7 Dishant Goyal, Varunkumar Jayapaul, and Venkatesh Raman. Elusiveness of Finding Degrees. In Daya Gaur and N.S. Narayanaswamy, editors, *Algorithms and Discrete Applied Mathematics*, pages 242–253, Cham, 2017. Springer International Publishing.

8 Sean Horan. Implementation of Majority Voting Rules, 2013.

9 Michael Kim, Warut Suksompong, and Virginia Vassilevska Williams. Who Can Win a Single-Elimination Tournament? *SIAM Journal on Discrete Mathematics*, 31(3):1751–1764, 2017. doi:10.1137/16M1061783.

10 Michael Kim and Virginia Vassilevska Williams. Fixing Tournaments for Kings, Chokers, and More. *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 561–567, 2015. URL: https://www.ijcai.org/Proceedings/15/Papers/085.pdf.

11 Justin Kruger and Stéphane Airiau. Refinements and Randomised Versions of Some Tournament Solutions. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '17, pages 1584–1586. IFAAMS, 2017. URL: http://dl.acm.org/citation.cfm?id=3091125.3091370.

12 Willi Maurer. On Most Effective Tournament Plans With Fewer Games than Competitors. *The Annals of Statistics*, 3(3):717–727, 1975. URL: http://www.jstor.org/stable/2958441.

13 Ariel D. Procaccia. A Note on the Query Complexity of the Condorcet Winner Problem. *Information Processing Letters*, 108(6):390–393, 2008. doi:10.1016/j.ipl.2008.07.012.

14 Dmitry Ryvkin. The Predictive Power of Noisy Elimination Tournaments. CERGE-EI Working Papers wp252, The Center for Economic Research and Graduate Education - Economics Institute, Prague, March 2005. doi:10.2139/ssrn.849225.

15 Sanjay Srivastava and Michael A. Trick. Sophisticated Voting Rules: the Case of Two Tournaments. *Social Choice and Welfare*, 13(3):275–289, June 1996. doi:10.1007/BF00179232.

16 Isabelle Stanton and Virginia Vassilevska Williams. Manipulating Single-Elimination Tournaments in the Braverman-Mossel Model. In Ulle Endriss Edith Elkind and Jérôme Lang, editors, *Proceedings of the IJCAI-2011 Workshop on Social Choice and AI*, pages 87–92, 2011.

17 Isabelle Stanton and Virginia Vassilevska Williams. Manipulating Stochastically Generated Single-Elimination Tournaments for Nearly All Players. In Ning Chen, Edith Elkind, and Elias Koutsoupias, editors, *Internet and Network Economics*, pages 326–337, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

18 Isabelle Stanton and Virginia Vassilevska Williams. Rigging Tournament Brackets for Weaker Players. In Toby Walsh, editor, *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 357–364. IJCAI/AAAI, 2011. doi:10.5591/978-1-57735-516-8/IJCAI11-069.

19 Virginia Vassilevska Williams. Fixing a Tournament. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, AAAI'10, pages 895–900. AAAI Press, 2010. URL: `http://dl.acm.org/citation.cfm?id=2898607.2898751`.

20 Thuc Vu, Alon Altman, and Yoav Shoham. On the Agenda Control Problem for Knockout Tournaments. In Ulle Endriss and Paul W. Goldberg, editors, *Proceedings of the 2nd International Workshop on Computational Social Choice*, pages 415–426, 2008.

21 Thuc Vu and Yoav Shoham. Fair Seeding in Knockout Tournaments. *ACM Transactions on Intelligent Systems and Technology*, 3(1):9:1–9:17, October 2011. `doi:10.1145/2036264.2036273`.

A Proof of Lemma 4

▶ **Lemma 4.** *Let S be a set of competitors partitioned into three components C_1, C_2, C_3 such that every member of component C_i beats every member of component $C_{(i \bmod 3)+1}$. Fix probabilities $p_i^{(0)}$ summing to 1, and let $p_i^{(k)}$ denote the probability that a member of component C_i wins a balanced bracket of height k where each leaf is labeled independently according to $p_i^{(0)}$. If for some $K \in \mathbb{N}$ and $0 < \epsilon \leq 2^{-10}$, $\epsilon^2 \leq p_3^{(K)} \leq \epsilon$ and $\epsilon \leq p_1^{(K)} \leq 2\epsilon$, then there exists $K + \log(\frac{1}{\epsilon}) \leq K' \leq K + 3\log(\frac{1}{\epsilon})$ and $\epsilon^{2\log(\frac{1}{\epsilon})} \leq \delta \leq \epsilon^{\log(\frac{1}{\epsilon})/4}$ such that $\delta^2 \leq p_2^{(K')} \leq \delta$ and $\delta \leq p_3^{(K')} \leq 2\delta$. Furthermore, if $\epsilon \leq 2^{-75}$ then for any $K'' \in [K', K' + 2^5 \log(\frac{1}{\epsilon})]$, $p_2^{(K'')}, p_3^{(K'')} \leq \epsilon^{\log(\frac{1}{\epsilon})/2^5}$.*

Proof. Since we have labeled each leaf independently, the two children of a node are independent, so we can easily calculate $p_i^{(k)}$ recursively. For all $k \geq 0$,

$$p_i^{(k+1)} = \left(p_i^{(k)}\right)^2 + 2p_i^{(k)}p_{(i \bmod 3)+1}^{(k)} = p_i^{(k)}\left(p_i^{(k)} + 2p_{(i \bmod 3)+1}^{(k)}\right).$$

We will proceed by phases. Phase 1 will be the time during which $p_2^{(k)}$ shrinks to $1/2$; phase 2 will extend from there to the time when $p_2^{(k)}$ shrinks to less than $p_3^{(k)}$ (this will be K'); and phase 3 will be the additional $2^5 \log(\frac{1}{\epsilon})$ steps after K'.

Phase 1. Let $K_1 > K$ be the first step for which $p_1^{(k)} + p_3^{(k)} > 1/2$. Such a step must exist because for $K \leq k < K_1$, $p_1^{(k)} + p_3^{(k)} \leq 1/2$, and thus

$$p_3^{(k+1)} = p_3^{(k)}\left(p_3^{(k)} + 2p_1^{(k)}\right) \leq p_3^{(k)} \leq \epsilon,$$

i.e., $p_3^{(k)}$ is weakly decreasing on this interval. Thus also,

$$p_1^{(k+1)} = p_1^{(k)}\left(p_1^{(k)} + 2p_2^{(k)}\right) = p_1^{(k)}(1 - p_3^{(k)} + p_2^{(k)}) \geq p_1^{(k)}(1.5 - \epsilon) \geq p_1^{(k)}\sqrt{2}$$

for k in this interval, since $p_1^{(k)} + p_3^{(k)} \leq 1/2$ implies $p_2^{(k)} \geq 1/2$, and since $\epsilon \leq 2^{-10} < 1.5 - \sqrt{2}$. Therefore, $p_1^{(k)}$ is increasing by at least a constant factor every step, and so eventually $p_1^{(k)} + p_3^{(k)}$ will exceed $1/2$. Note also that $p_i^{(k+1)} \leq 2p_i^{(k)}$ for all i, k, so $p_1^{(k)}$ is increasing by at least a factor of $\sqrt{2}$ and at most a factor of 2 on this interval.

This leads to the following observations about phase 1:

1. $\epsilon 2^{(k-K)/2} \leq p_1^{(k)} \leq \epsilon 2^{k-K+1}$ for any $k \in [K, K_1]$
2. $\epsilon^2\left(\epsilon 2^{(K_1-K+3)/4}\right)^{K_1-K} \leq p_3^{(K_1)} \leq \epsilon\left(\epsilon 2^{(K_1-K+5)/2}\right)^{K_1-K}$
3. $\log(\frac{1}{\epsilon}) - 3 \leq K_1 - K \leq 2\log(\frac{1}{\epsilon})$

Observation 1 follows directly from our initial assumption on $p_1^{(K)}$ and the bounds on the factor by which it increases each step.

Observation 2 can be shown via observation 1 and our initial assumptions as follows, using the fact that $0 \le p_3^{(k)} \le p_1^{(k)}$ on this interval.

$$p_3^{(K_1)} = p_3^{(K)} \prod_{t=K}^{t=K_1-1} \left(p_3^{(t)} + 2p_1^{(t)}\right)$$

$$\begin{aligned}
p_3^{(K)} \prod_{t=K}^{t=K_1-1} 2p_1^{(t)} \le& p_3^{(K_1)} \le p_3^{(K)} \prod_{t=K}^{t=K_1-1} 3p_1^{(t)} \\
p_3^{(K)} \prod_{t=0}^{t=K_1-K-1} 2p_1^{(K)} 2^{t/2} \le& p_3^{(K_1)} \le p_3^{(K)} \prod_{t=0}^{t=K_1-K-1} 3p_1^{(K)} 2^{t} \\
p_3^{(K)} \left(2p_1^{(K)}\right)^{K_1-K} 2^{(K_1-K-1)(K_1-K)/4} \le& p_3^{(K_1)} \le p_3^{(K)} \left(3p_1^{(K)}\right)^{K_1-K} 2^{(K_1-K-1)(K_1-K)/2} \\
\epsilon^2 \left(\epsilon 2^{(K_1-K+3)/4}\right)^{K_1-K} \le& p_3^{(K_1)} \le \epsilon \left(\epsilon 2^{(K_1-K+5)/2}\right)^{K_1-K}
\end{aligned}$$

Finally, observation 3 is obtained from observation 1 based on how long it would take for $p_1^{(k)}$ to get to $1/2$ (or rather in the range $[1/2-\epsilon, 1]$). Specifically, plugging in $k = K_1$ to observation 1 and taking the log of both sides:

$$\begin{aligned}
\log \epsilon + \frac{K_1 - K}{2} \le& \log p_1^{(K_1)} \le \log \epsilon + K_1 - K + 1 \\
\log p_1^{(K_1)} - \log \epsilon - 1 \le& K_1 - K \le 2 \log p_1^{(K_1)} - 2 \log \epsilon \\
\log(\frac{1}{2} - \epsilon) - \log \epsilon - 1 \le& K_1 - K \le 2 \log 1 - 2 \log \epsilon \\
\log(\frac{1}{\epsilon}) - 3 \le& K_1 - K \le 2 \log(\frac{1}{\epsilon})
\end{aligned}$$

Phase 2. Let $K_2 > K_1$ be the first step for which $p_3^{(k)} > p_2^{(k)}$. We claim that $K' = K_2$ is as required in the statement of the lemma.

To show that such a step must exist, note that at step K_1, we have $1/4 < p_2^{(K_1)} < 1/2$, $p_3^{(K_1)} \le \epsilon$, and therefore $p_1^{(K_1)} > 1/2 - \epsilon$. Furthermore, since $p_3^{(k)} \le p_2^{(k)}$ on this interval, for $K_1 \le k < K_2$,

$$p_1^{(k+1)} = p_1^{(k)} \left(p_1^{(k)} + 2p_2^{(k)}\right) = p_1^{(k)} \left(1 - p_3^{(k)} + p_2^{(k)}\right) \ge p_1^{(k)}$$

i.e., $p_1^{(k)}$ is weakly increasing. In particular,

$$p_1^{(K_1+1)} = p_1^{(K_1)} \left(p_1^{(K_1)} + 2p_2^{(K_1)}\right) \ge (1/2 - \epsilon)(1/2 - \epsilon + 1/4) > 0.6 > 1/2.$$

Thus for every step after the first, $p_3^{(k)}$ is increasing by a factor of $p_3^{(k)} + 2p_1^{(k)} > 1.2$, while $p_2^{(k)}$ is multiplied by a factor of $p_2^{(k)} + 2p_3^{(k)} = 1 - p_1^{(k)} + p_3^{(k)} < 1$, and they will eventually cross.

We make the following observations about phase 2:

1. $\left(\frac{1}{4}\right)^{2^{k-K_1}} \le p_2^{(k)} \le \left(\frac{1}{2}\right)^{2^{k-K_1-1}}$ for any $k \in [K_1+1, K_2]$
2. $1.5^{k-K_1-2} p_3^{(K_1)} \le p_3^{(k)} \le 2^{k-K_1} p_3^{(K_1)}$ for any $k \in [K_1+2, K_2]$
3. $\log\log(\frac{1}{\epsilon}) \le K_2 - K_1 \le \log(\frac{1}{\epsilon})$

Observation 1 follows from the fact that $\frac{1}{4} \leq p_2^{(K_1)} \leq \frac{1}{2}$ and $p_3^{(K_1)} \leq \epsilon \leq 2^{-10}$. Recall that no probability can more than double in a single step. Thus

$$\begin{aligned}
p_2^{(K_1+1)} &\leq p_2^{(K_1)}\left(p_2^{(K_1)} + 2p_3^{(K_1)}\right) \leq \frac{1}{2}\left(\frac{1}{2} + 2^{-9}\right) \leq 0.251 \\
p_2^{(K_1+2)} &\leq p_2^{(K_1+1)}\left(p_2^{(K_1+1)} + 2p_3^{(K_1+1)}\right) \leq 0.251(0.251 + 2^{-8}) \leq 0.064 \\
p_2^{(K_1+3)} &\leq p_2^{(K_1+2)}\left(p_2^{(K_1+2)} + 2p_3^{(K_1+2)}\right) \leq 0.064(0.064 + 2^{-7}) \leq 0.0046.
\end{aligned}$$

Therefore, since $p_2^{(k+1)} = p_2^{(k)}\left(p_2^{(k)} + 2p_3^{(k)}\right)$, we have for any $k \geq K_1 + 3$,

$$\begin{aligned}
\left(p_2^{(k)}\right)^2 \leq & p_2^{(k+1)} \leq 3\left(p_2^{(k)}\right)^2 \\
\left(\frac{1}{4}\right)^{2^{k-K_1}} \leq & p_2^{(k+1)} \leq (3 \times 0.0046)^{2^{k-K_1-3}} \leq \left(\frac{1}{2}\right)^{2^{k-K_1-1}}.
\end{aligned}$$

Since $p_2^{(k)}$ is decreasing we have $p_3^{(k)} \leq p_2^{(k)} \leq \frac{1}{8}$ for $K_1 + 2 \leq k \leq K_2$. Thus $p_3^{(k)}$ increases by at most a factor of 2, and at least a factor of $2p_1^{(k)}$ which is at least 1.5 after the first two steps. ($p_3^{(k)}$ may decrease by a factor no smaller than $1 - \epsilon$ in the first step, but the next step more than cancels this out.) This yields observation 2.

Observation 3 we'll just show by plugging in the given values for $K_2 - K_1$ into the bounds for $p_2^{(k)}$ and $p_3^{(k)}$ and showing that they either must have, or must not have crossed by the given time.

First let us verify that after another $\log(\frac{1}{\epsilon})$ steps it must be the case that $p_3^{(k)} > p_2^{(k)}$. Specifically, if we assume that $K_1 + \log(\frac{1}{\epsilon}) < K_2$, we obtain the following contradiction:

$$\begin{aligned}
p_3^{(K_1+\log(\frac{1}{\epsilon}))} &\geq 1.5^{\log(\frac{1}{\epsilon})-2} p_3^{(K_1)} \geq 1.5^{\log(\frac{1}{\epsilon})-2}\epsilon^2\left(\epsilon 2^{(K_1-K+3)/4}\right)^{K_1-K} \\
&\geq 1.5^{\log(\frac{1}{\epsilon})-2}\epsilon^2\left(\epsilon\left(\frac{1}{\epsilon}\right)^{1/2} 2^{3/4}\right)^{2\log(\frac{1}{\epsilon})} \\
&= \left(\frac{2}{3}\right)^2\left(\frac{3}{2}\right)^{\log(\frac{1}{\epsilon})}\epsilon^{\log(\frac{1}{\epsilon})+2} 2^{3\log(\frac{1}{\epsilon})/2} \\
&> \left(\frac{1}{2}\right)^{\log(\frac{1}{\epsilon})(\log(\frac{1}{\epsilon})+2)} > \left(\frac{1}{2}\right)^{1/2\epsilon} \geq p_2^{(K_1+\log(\frac{1}{\epsilon}))}
\end{aligned}$$

since $1/2\epsilon > \log^2(\frac{1}{\epsilon}) + 2\log(\frac{1}{\epsilon})$ for $\epsilon \leq 2^{-7}$. This establishes the upper bound on $K_2 - K_1$.

To establish that $K_2 - K_1 \geq \log\log(\frac{1}{\epsilon})$, we need to show that after that many steps, it still holds that $p_3^{(k)} \leq p_2^{(k)}$.

$$\begin{aligned}
p_3^{(K_1+\log\log(\frac{1}{\epsilon}))} &\leq 2^{\log\log(\frac{1}{\epsilon})} p_3^{(K_1)} \leq 2^{\log\log(\frac{1}{\epsilon})+2}\epsilon\left(\epsilon 2^{(\log(\frac{1}{\epsilon})-3+5)/2}\right)^{\log(\frac{1}{\epsilon})-3} \\
&= \log(\frac{1}{\epsilon})\epsilon\left(2\epsilon^{1/2}\right)^{\log(\frac{1}{\epsilon})-3} \\
&= \frac{1}{8}\log(\frac{1}{\epsilon})\epsilon^{\log(\frac{1}{\epsilon})/2-3/2} \\
&\leq \epsilon^{\log(\frac{1}{\epsilon})/4} \leq \left(\frac{1}{4}\right)^{\log(\frac{1}{\epsilon})} \leq p_2^{(K_1+\log\log(\frac{1}{\epsilon}))}
\end{aligned}$$

where the last line holds for $\epsilon \leq 2^{-10}$. Thus phase 2 must proceed for more than $\log\log(\frac{1}{\epsilon})$ steps.

Note also that observation 3 (together with observation 3 of phase 1) implies that $K + \log(\frac{1}{\epsilon}) \leq K' \leq K + 3\log(\frac{1}{\epsilon})$, as required.

Finally, we need to establish that there exists a δ as required in the statement of the lemma. If $p_3^{(K'-1)} \leq p_2^{(K')} \leq p_3^{(K')}$, let $\delta = p_3^{(K')}$. Then trivially $\delta \leq p_3^{(K')} \leq 2\delta$ and also

$$\delta^2 \leq \delta/2 \leq p_3^{(K'-1)} \leq p_2^{(K')} \leq p_3^{(K')} = \delta$$

since $\delta \leq 1/2$. Otherwise, $p_2^{(K')} < p_3^{(K'-1)}$; in this case, let $\delta = p_3^{(K'-1)}$. Again trivially, $\delta \leq p_3^{(K')} \leq 2\delta$. Additionally,

$$\delta^2 = \left(p_3^{(K'-1)}\right)^2 \leq \left(p_2^{(K'-1)}\right)^2 \leq p_2^{(K')} \leq p_3^{(K'-1)} = \delta.$$

Now to lower-bound δ, using observations 2 and 3, we have

$$\begin{aligned}
2\delta &\geq p_3^{(K')} \geq 1.5^{\log\log(\frac{1}{\epsilon})-2} p_3^{(K_1)} \\
&\geq \left(\frac{2}{3}\right)^2 \log(\frac{1}{\epsilon})^{1/2}\epsilon^2 \left(\epsilon 2^{(2\log(\frac{1}{\epsilon})+1)/4}\right)^{2\log(\frac{1}{\epsilon})} \\
&\geq \left(\frac{2}{3}\right)^2 \log(\frac{1}{\epsilon})^{1/2}\epsilon^{\log(\frac{1}{\epsilon})+3/2} \geq 2\epsilon^{2\log(\frac{1}{\epsilon})} \\
\delta &\geq \epsilon^{2\log(\frac{1}{\epsilon})}
\end{aligned}$$

where the penultimate line holds for $\epsilon \leq 2^{-4}$.

As for an upper bound:

$$\begin{aligned}
\delta &\leq p_3^{(K')} \leq p_3^{(K_1)} 2^{\log(\frac{1}{\epsilon})} \\
&\leq \epsilon \left(\epsilon 2^{(\log(\frac{1}{\epsilon})-3+5)/2}\right)^{\log(\frac{1}{\epsilon})-3} 2^{\log(\frac{1}{\epsilon})} \\
&= \left(2\epsilon^{1/2}\right)^{\log(\frac{1}{\epsilon})-3} = 2^{-3}\epsilon^{(\log(\frac{1}{\epsilon})-5)/2} \leq \epsilon^{\log(\frac{1}{\epsilon})/4}
\end{aligned}$$

where the last line holds for $\epsilon \leq 2^{-10}$. Thus we have shown that $K' = K_2$ is as required in the statement of the lemma.

Phase 3. Finally, we want to establish that $p_3^{(k)}$ and $p_2^{(k)}$ stay relatively small for $2^5\log(\frac{1}{\epsilon})$ steps past K', provided ϵ is small enough. In particular, we know that $p_3^{(K')} > p_2^{(K')}$, and that no probability can more than double at each time step. Thus

$$\begin{aligned}
p_3^{(K'+t)}, p_2^{(K'+t)} &\leq 2^t p_3^{(K')} \leq 2^t \left(2\epsilon^{1/2}\right)^{\log(\frac{1}{\epsilon})-3} \\
&= 2^{t-3}\epsilon^{(\log(\frac{1}{\epsilon})-5)/2} \\
&\leq \epsilon^{\log(\frac{1}{\epsilon})/2^5}
\end{aligned}$$

where the last line holds if

$$\begin{aligned}
2^{t-3}\epsilon^{-5/2} &\leq \epsilon^{-(2^4-1)\log(\frac{1}{\epsilon})/2^5} \\
t - 3 - \frac{5}{2}\log(\epsilon) &\leq (2^4-1)\log^2(\epsilon)/2^5 \\
t &\leq \frac{2^4-1}{2^5}\log^2(\epsilon) + \frac{5}{2}\log(\epsilon) + 3
\end{aligned}$$

In particular, since we want this to hold up to $t = 2^5\log(\frac{1}{\epsilon})$, $\epsilon \leq 2^{-75}$ suffices. This completes the proof. ◀

Routing Symmetric Demands in Directed Minor-Free Graphs with Constant Congestion

Timothy Carpenter
Dept. of Computer Science & Engineering, The Ohio State University, Columbus, OH, USA
carpenter.454@osu.edu

Ario Salmasi
Dept. of Computer Science & Engineering, The Ohio State University, Columbus, OH, USA
salmasi.1@osu.edu

Anastasios Sidiropoulos
Dept. of Computer Science, University of Illinois at Chicago, USA
sidiropo@uic.edu

Abstract

The problem of routing in graphs using node-disjoint paths has received a lot of attention and a polylogarithmic approximation algorithm with constant congestion is known for undirected graphs [Chuzhoy and Li 2016] and [Chekuri and Ene 2013]. However, the problem is hard to approximate within polynomial factors on directed graphs, for any constant congestion [Chuzhoy, Kim and Li 2016].

Recently, [Chekuri, Ene and Pilipczuk 2016] have obtained a polylogarithmic approximation with constant congestion on directed planar graphs, for the special case of symmetric demands. We extend their result by obtaining a polylogarithmic approximation with constant congestion on arbitrary directed minor-free graphs, for the case of symmetric demands.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Routing, Node-disjoint, Symmetric demands, Minor-free graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.14

Category APPROX

Funding This work was supported by NSF under CAREER award 1453472 and grant CCF 1815145.

1 Introduction

Routing in graphs along disjoint paths is a fundamental problem with numerous applications in various domains [1, 2, 3, 23, 24]. Disjoint path problems have been well-studied in both the directed and undirected setting, and it is known that the directed formulations of these problems are generally more difficult to approximate [14, 11]. The recent work of [5, 6] has brought to light a more tractable formulation of the directed version of these problems by considering routing symmetric demand pairs with constant congestion.

Two of the most well-known and studied disjoint path problems are the node-disjoint paths problem (NDP) and the edge-disjoint paths problems (EDP). In these problems, the goal is to connect a set of node pairs through node- or edge-disjoint paths in an undirected graph. It is known that the decision version of NDP is NP-complete [20], and it has been shown to be fixed parameter tractable [26]. But there remain gaps in our understanding of their approximability. For both EDP and NDP on n-node graphs, the state of the art is an $O(\sqrt{n})$-approximation [9], [22]. For planar graphs, a slightly better bound of $\tilde{O}(n^{9/19})$-approximation is known [13]. Even for the case of the grid, only a $\tilde{O}(n^{1/4})$-approximation for NDP is known [12]. For hardness of approximation, it is known that both NDP and EDP are $2^{\Omega\left(\sqrt{\log n}\right)}$-hard to approximate, unless all problems in NP have algorithms with running time $n^{\log n}$ [14].


© Timothy Carpenter, Ario Salmasi, and Anastasios Sidiropoulos;
licensed under Creative Commons License CC-BY



Progress has been made on relaxed versions of these problems. One such relaxation is the all-or-nothing flow problem (ANF), where a subset $\mathcal{M}' \subseteq \mathcal{M}$ is routed if there is a feasible multicommodity flow routing one unit of flow for each pair in $\mathcal{M}'$. Poly-logarithmic approximations are known for ANF [8, 7]. Another relaxation is to allow some small constant congestion on the nodes or edges. For this relaxation, poly-logarithmic approximations have been obtained for EDP with congestion 2 [15], and for NDP with congestion $O(1)$ [4].

It is natural to extend the study of disjoint path problems to directed graphs. However, these problems are known to be significantly harder on directed graphs. Even the case of ANF with constant congestion c allowed has an $n^{\Omega(1/c)}$ inapproximability bound [11]. However, a more tractable case is found by considering symmetric demand pairs. The study of maximum throughput routing problems in directed graphs with symmetric demand pairs began in [5]. In this setting the graph G is directed, and routing a source-destination pair (s_i, t_i) requires finding a path from s_i to t_i and a path from t_i to s_i. We let Sym-Dir-ANF be the analogue of ANF, and Sym-Dir-NDP be the analogue of NDP in this setting. A poly-logarithmic approximation for Sym-Dir-ANF is obtained in [5]. Subsequently, in [6] a randomized poly-logarithmic approximation with constant congestion on planar graphs for Sym-Dir-NDP is obtained.

1.1 Our contribution

We consider the problem of routing symmetric demands along node-disjoint paths in directed graphs. We refer to this problem as Sym-Dir-NDP. Letting $G = (V, E)$ be a directed graph with unit node capacities and $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\} \subseteq V \times V$ be a set of source-destination pairs, we say that $(G, \mathcal{M})$ is an instance of Sym-Dir-NDP. Routing a pair (s_i, t_i) requires finding a path from s_i to t_i, and from t_i to s_i. A solution to an instance of Sym-Dir-NDP is a routing strategy maximizing the number of pairs routed through disjoint paths. We refer to a solution having congestion ζ, if no vertex is used in more than ζ paths. Our contribution generalizes the result from [6] from the class of directed planar graphs to arbitrary directed minor-free graphs. We now formally state our results and briefly highlight the methods used. Our main result is the following.

▶ **Theorem 6.** *Let G be an H-minor free graph. There is a polynomial time randomized algorithm that, with high probability, achieves an $\Omega\left(\frac{1}{h^7\sqrt{h}\log^{5/2}(n)}\right)$-approximation with congestion 5 for Sym-Dir-NDP instances in G, where h is an integer dependent only on H.*

The approximation algorithm in this theorem is obtained by extending the algorithm of [6]. For an instance $(G, \{(s_1, t_1), \ldots, (s_k, t_k)\})$ of Sym-Dir-NDP, we say that the set $\mathcal{T} = \{s_1, \ldots, s_k\} \cup \{t_1, \ldots, t_k\}$ is the set of terminals. Speaking broadly, the algorithm obtained in Theorem 6 consists of the following steps.

1. Using a multicommodity flow based LP relaxation and the well-linked decomposition of [6], reduce to an instance in which the terminals $\mathcal{T}$ are α-well-linked for a fixed constant α.
2. Find a large routing structure connected to a large proportion of the terminals.
3. Use the routing structure to connect a large number of the source-destination pairs.

From here on, we shall refer to the routing structure as the crossbar. The reduction we use in Step 1 allows us to reduce an instance of Sym-Dir-NDP to an instance on an Eulerian graph of small maximum degree, and where the terminals are α-well-linked. This comes at the cost of then having a randomized algorithm for the original instance. This reduction comes from [6], and while there it is used for planar graphs, we were fortunate in that it

can also be used for general graphs. The routing scheme of Step 3 is also thanks to [6], and relies on a similar crossbar construction. Our main contribution to this line of research is in finding an appropriate crossbar construction for Step 2.

To build our crossbar, we would ideally find a "flat" grid minor so that some constant fraction of the terminal pairs can be routed along node-disjoint paths to the interface of the grid minor (a "flat" grid minor is one in which the grid minor is connected with the rest of the graph only through the outer face). Then we would have the following sets of node-disjoint paths along which to route the terminal pairs: the paths from the terminals to the interface, the paths from terminals to terminals implied by the node-well-linked property of the terminals, the concentric cycles of the grid minor, and the paths connecting the outermost and innermost cycles of the grid minor. From these, just as in [6] we can construct a routing scheme with congestion 5. To find a suitable flat grid minor, we combine results of [10] and [28] to show that flat grid minors of a suitable size can be found. We then show that if for the flat grid minor produced we cannot route a large enough fraction of the terminals to the interface then there exists some vertex which can be eliminated from the graph without destroying a potential solution to the problem. Thus, we find and test flat grid minors until one suitable to be used in the crossbar is found.

2 Notation and Preliminaries

We now introduce some notation and definitions that are used throughout the paper.

Directed and undirected graphs

From any directed graph G we can obtain an undirected graph G^{UN} as follows. We set $V(G^{\mathsf{UN}}) = V(G)$ and $E(G^{\mathsf{UN}}) = \{\{u, v\} : (u, v) \in E(G) \vee (v, u) \in E(G)\}$. We refer to G^{UN} as the *underlying undirected graph* of G.

Flat subgraphs

We say that a planar subgraph H of an undirected graph G is *flat* if there exists a planar drawing Φ of H such that for any $\{u, v\} \in E(G)$, with $u \in V(H)$ and $v \in V(G) \setminus V(H)$, we have that u is on the outer face of Φ.

Well-linked sets

Let G be a directed (resp. undirected) graph. A set $X \subseteq V(G)$ is node-well linked in G if for any two disjoint subsets $Y, Z \subset X$ of equal size, there exist $|Y|$ node-disjoint directed (resp. undirected) paths from Y to Z, such that each vertex in Y is the start of exactly one path, and each vertex in Z is the end of exactly one path. For some $\alpha \in (0, 1)$, we say that X is α-node well-linked if for any two disjoint subsets $Y, Z \subset X$ of equal size, there exist $|Y|$ directed (resp. undirected) paths from Y to Z such that no vertex is in more than $1/\alpha$ of these paths; In other words, we allow a node congestion of $1/\alpha$ for these paths.

Directed and undirected treewidth

For a directed graph G, we will denote by $\mathsf{dtw}(G)$ the *directed treewidth of* G, and we will denote by $\mathsf{tw}(G^{\mathsf{UN}})$ the (undirected) *treewidth of* G^{UN}. Directed treewidth is a global connectivity measure introduced in [19, 25], and just as undirected treewidth is defined by the minimum width tree decomposition, directed treewidth is defined by the minimum size of

what is termed an arboreal decomposition. Instead of providing the full definitions of directed and undirected treewidth here, we only ask the reader to make a note of the following two important facts:

- If G is an Eulerian directed graph with max degree Δ, then $\mathsf{tw}(G^{\mathsf{UN}}) \leq \mathsf{dtw}(G) = O(\Delta \cdot \mathsf{tw}(G^{\mathsf{UN}}))$ [19].
- If a directed graph G contains an α-well-linked set X, then $\mathsf{dtw}(G) = \Omega(\alpha|X|)$ [25].

Clique-sums

Let G_1 and G_2 be two graphs. A *clique-sum* of G_1 and G_2 is any graph that is obtained by identifying a clique in G_1 with a clique of the same size in G_2, and then possibly removing some edges in the resulting shared clique. An *h-clique-sum*, or *h-sum* for short, is a clique-sum where the identified cliques have at most h vertices.

Nearly-embeddable and minor-free graphs

We say that a graph is (a, g, k, p)*-nearly embeddable* if it is obtained from a graph of Euler genus g by adding a apices and k vortices of pathwidth p. We say that a graph is h-nearly embeddable if it is (a, g, k, p)-nearly embeddable for some $a, g, k, p \leq h$. The following is implicit in [27].

▶ **Theorem 1** (Robertson and Seymour [27]). *Let H be any graph. Every H-minor-free graph can be obtained by at most h-clique-sums of graphs that are h-nearly embeddable graphs, where h is a non-negative integer dependent on H.*

Note that the result of the above theorem is existential. Demaine, Hajiaghayi, and Kawarabayashi in [16] provide an algorithm to compute this decomposition in polynomial time, for any fixed minor H.

3 The Algorithm for Minor-Free Graphs

We first use the following result of [6] to reduce the problem to the case of Eulerian graphs with small degrees. Note that this result is stated for planar graphs in [6], but the proof does not use planarity, and thus can be stated for general graphs.

▶ **Lemma 2** (Chekuri, Ene & Pilipczuk [6]). *Suppose that there is a polynomial time algorithm for $\Omega(1)$-node-well-linked instances of Sym-Dir-NDP in directed Eulerian graphs of maximum degree Δ that achieves a $\beta(\Delta)$-approximation with congestion c. Then there is a polynomial time randomized algorithm that, with high probability, achieves a $\beta(O(\log^2 k)) \cdot O(\log^6 k)$-approximation with congestion c for arbitrary instances of Sym-Dir-NDP in directed graphs, where k is the number of pairs in the instance.*

Now we describe how to construct the crossbar in minor-free graphs, assuming that we are given a $m \times m$ flat grid minor Γ, for some large enough m, and a family of λm node-disjoint paths connecting the set of terminals and the interface of Γ, for some constant λ. The following is our main technical result, which is similar to the one in [6] for planar graphs. We use here a generalized notion of *enclosed* for flat grids in non-planar graphs. Let H be a directed graph with a flat grid minor η. Let u^{out} be an arbitrary vertex not contained in η. Let C be some cycle contained within η. We say a vertex u is *enclosed* by C if all paths in H^{UN} from u to u^{out} intersect C. We now find the desired concentric cycles in G. The proof is deferred to Section 4.

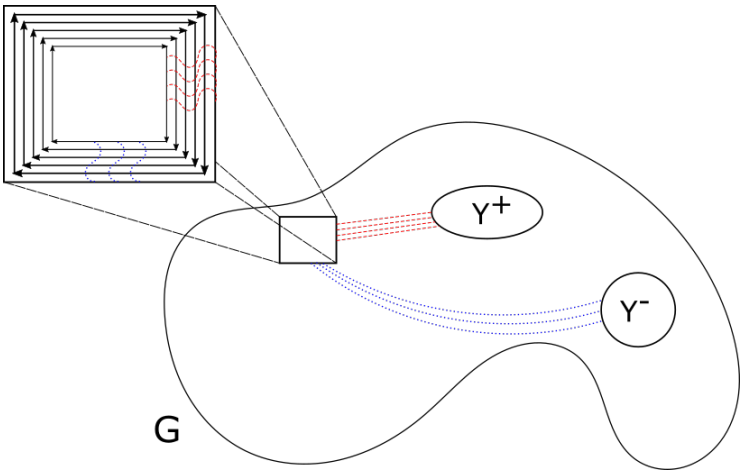


Figure 1 An example for case (1) of Theorem 3. The red paths are the node disjoint paths in P^+, going from Y^+ to the innermost of the concentric cycles, and the blue paths correspond to the node disjoint paths in P^-, going from Y^- to the innermost of the concentric cycles.

▶ **Theorem 3.** *Let G be a directed minor-free graph of maximum in-degree of at most Δ. Let X be an α-node-well-linked set in G with $|X| = \Omega\left(\frac{\Delta^2}{\alpha}\right)$. Let $m = \Omega\left(\frac{\alpha|X|}{\beta}\right)$, where β is a non-negative number dependent on G. Suppose that we can find a $m \times m$ flat grid minor Γ of G^{UN}, and a family of λm node-disjoint paths connecting X and the interface of Γ in G^{UN}, for some $0 < \lambda \leq 1$. One can in polynomial time find a set of $\Omega\left(\frac{\alpha|X|}{\beta\Delta}\right)$ concentric directed cycles going in the same direction w.r.t. a planar embedding of Γ (all clockwise or counter-clockwise), sets $Y^+, Y^- \in X$ with $|Y^+| = |Y^-| = \Omega\left(\frac{\alpha^2|X|}{\beta\Delta^2}\right)$, and families P^+ and P^- of node-disjoint paths such that one of the following holds.*

(1) *None of the cycles enclose any vertex of $Y^+ \cup Y^-$, the family P^+ consists of $|Y^+|$ node-disjoint paths from Y^+ to the innermost cycle, and the family P^- consists of $|Y^-|$ node-disjoint paths from the innermost cycle to Y^- (See Figure 1).*

(2) *All cycles enclose $Y^+ \cup Y^-$, the family P^+ consists of $|Y^+|$ node-disjoint paths from $|Y^+|$ to the outermost cycle, and the family P^- consists of $|Y^-|$ node-disjoint paths from the outermost cycle to Y^-.*

In order to obtain such a crossbar, we need to find a flat grid minor of large enough size. The following Lemma provides us the desired flat grid minor, and the proof is deferred to Section 6.2.

▶ **Lemma 4.** *Let H be any graph and let G be an H-minor-free directed graph with treewidth t. Let X be an α-node-well-linked set in G with $|X| = \Omega\left(\frac{\Delta^2}{\alpha}\right)$. One can in polynomial time find a $r \times r$ flat grid minor Γ in G^{UN}, with $r = \Omega\left(\frac{t}{h^7\sqrt{h}\log^{5/2}(n)}\right)$, and a family of r node-disjoint paths connecting X and the interface of Γ, where h is an integer dependent only on the structure of H.*

Once we obtain a crossbar as described above, we can route a large subset of terminal pairs.

▶ **Lemma 5.** *Given the crossbar described in Theorem 3, one can get an $O\left(\frac{\Delta^2}{\beta\alpha^3}\right)$-approximation algorithm with congestion 5 for Sym-Dir-NDP in instances for which the input graph is minor-free and Eulerian with maximum in-degree Δ, the set of terminals is α-node-well-linked for some $\alpha \leq 1$, and β is dependent only on H.*

Proof. By applying the same routing scheme as in the one in [6], we get the desired result. ◀

Now we are ready to state the main result of this paper.

▶ **Theorem 6.** *Let G be a H-minor-free graph. There is a polynomial time randomized algorithm that, with high probability, achieves an $\Omega\left(\frac{1}{h^7\sqrt{h}\log^{5/2}(n)}\right)$-approximation with congestion 5 for Sym-Dir-NDP instances in G, where h is an integer dependent only on H.*

Proof. This is immediate by Lemmas 2, 4, 5, and Theorem 3. ◀

4 The Crossbar Construction

In this section we discuss the construction of the crossbar stated in Theorem 3. Before we give the proof of this Theorem we establish some auxiliary facts. Throughout this subsection, we assume that we are given the input of Theorem 3.

▶ **Lemma 7.** *One can in polynomial time find an integer $r = \Omega\left(\frac{\alpha|X|}{\beta}\right)$ and a sequence of node-disjoint concentric undirected cycles $C_1, C_2, \ldots, C_r$ in G^{UN}, with C_1 being the outermost and C_r being the innermost cycle.*

Proof. Let t be the treewidth of G^{UN}. Since X is α-node-well-linked in G, X is also α-node-well-linked in G^{UN}. Thus, $t = \Omega(\alpha|X|)$. Let Γ be a flat $m \times m$ grid minor of G^{UN}, as given in the input of Theorem 3. By losing a constant factor, we can construct a flat sub-divided $r \times r$ wall in G^{UN}, with $r = \Omega\left(\frac{\alpha|X|}{\beta}\right)$. Let C_1 be the outermost cycle of Γ, and for each $i \in \{2, \ldots, r\}$, let C_i be the outermost cycle of $\Gamma \setminus \cup_{1 \le j < i} V(C_i)$. ◀

As in [6], for a vertex set $Q \subseteq V(G^{\mathsf{UN}})$, a vertex $v \notin Q$, and an integer $\ell \ge 2\Delta$, we say that a vertex set S is a (v, Q, ℓ)-*isle* if $v \in S$, $G^{\mathsf{UN}}[S]$ is connected, $S \cap Q = \emptyset$, and $|N_{G^{\mathsf{UN}}}(S)| \le \ell$. Let $C_1, \ldots, C_r$ be the sequence of node-disjoint concentric undirected cycles in G^{UN} obtained from Lemma 7. We set isles S^{out} and S^{in} by choosing an arbitrary vertex v^{out} in C_1, and an arbitrary vertex v^{in} in C_r. Letting $\ell = \lfloor r/(4\Delta+2) \rfloor$, then S^{out} is the $(v^{\mathsf{out}}, X, \ell)$-isle and S^{in} is the $(v^{\mathsf{in}}, X, \ell)$-isle obtained. We also need that S^{out} and S^{in} are separated by many cycles. For this, we use the following Lemma of [6], the proof of which is slightly modified.

▶ **Lemma 8.** *The isle S^{out} does not contain any vertex that is enclosed by $C_{\ell+1}$, and the isle S^{in} does not contain any vertex that is not strictly enclosed by $C_{r-\ell}$.*

Proof. The proofs for S^{in} and S^{out} are symmetrical, so we focus on the case of S^{out}. Assume that S^{out} contains a vertex enclosed by $C_{\ell+1}$, and we will find a contradiction. Since $v^{out} \in S^{\mathsf{out}}$, S^{out} is connected in G^{UN}, and Γ is a flat wall, it must be that S^{out} contains a vertex from every cycle C_i, $1 \le i \le \ell+1$. Since $|N_{G^{\mathsf{UN}}}(S^{\mathsf{out}})| \le \ell$, for some $1 \le i \le \ell+1$ we have that $V(C_i)$ is completely contained in S^{out}. However, there are $r > \ell$ vertex-disjoint paths in G^{UN} connecting C_i with X. Thus, either $S^{\mathsf{out}} \cap X \neq \emptyset$ or $|N_{G^{\mathsf{UN}}}(S^{\mathsf{out}})| > \ell$, both of which are contradictions. ◀

We are almost ready to prove the main result of this section. We will make use of the following Lemma, which is implicit in [6]. Note that sets S'^{in} and S'^{out}, the concentric cycles $C'_1, \ldots, C'_{r'}$, and integers r' and Δ' in the next Lemma are defined for a planar graph G' as described in [6].

▶ **Lemma 9.** *Let G' be an Eulerian, planar directed graph, with sets $S'^{\mathsf{in}}, S'^{\mathsf{out}}$ separated by concentric cycles $C'_1, \ldots, C'_{r'}$, and let $\ell' = \lfloor r'/(4\Delta'+2) \rfloor$, where Δ' is the the maximum in-degree of G'. Then one can in polynomial time find $\lceil \ell'/2 \rceil$ node-disjoint directed concentric*

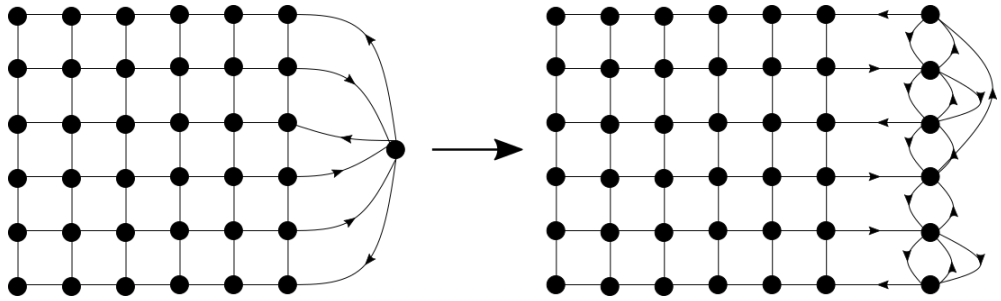


Figure 2 Maintaining an Eulerian graph with bounded degree.

cycles, all going in the same direction (all clockwise or all counter-clockwise), such that all vertices of S'^{in} are strictly enclosed by the innermost cycle, and all vertices of S'^{out} are not enclosed by the outermost cycle, or vice versa, with the roles of S'^{in} and S'^{out} swapped.

We will use Lemma 9 to find concentric cycles in minor-free graphs. We first generalize the notion of *enclosed* for flat grids in non-planar graphs. Let H be a directed graph with a flat grid minor η. Let u^{out} an arbitrary vertex not contained in η. Let C be some cycle contained within η. We say a vertex u is *enclosed* by C if all paths in H^{UN} from u to u^{out} intersect C. We now find the desired concentric cycles in G.

▶ **Lemma 10.** *One can in polynomial time find $\lceil \ell/2 \rceil$ node-disjoint directed concentric cycles in G, all going in the same direction (all clockwise or all counter-clockwise), such that all vertices of S^{in} are enclosed by the innermost cycle, and all vertices of S^{out} are not enclosed by the outermost cycle, or vice versa, with the roles of S^{in} and S^{out} swapped.*

Proof. We proceed by creating G' from G as follows. Let

$$Z = \{v \in V(G) : v \in V(C_1) \text{ or } v \text{ is not in the component of } G \setminus V(C_1) \text{ containing } C_2\}.$$

Let $G' = G/Z$, i.e. G' is the graph created by identifying all vertices in Z to a single vertex z. Since G is Eulerian, G' is also Eulerian. Furthermore, we can delete any self-loops on z, and G' is still Eulerian. Since $C_1, \ldots, C_r$ are contained within a flat grid minor of G, G' is therefore a planar graph. The only impediment to directly applying Lemma 9 is that the in-degree δ of z might be greater than Δ. We can eliminate this by replacing z with a path P of length δ, with edges directed both ways between adjacent vertices. We then connect the vertices formerly connected to z to vertices in P, maintaining the planarity of G'. Then, to restore G' as an Eulerian graph, for the vertices in P with an imbalance between in- and out-degree we can create a new edge (See Figure 2).

After these modifications, G' is an Eulerian, planar digraph with maximum in-degree Δ. Let $S'_{\mathsf{in}} = S^{\mathsf{in}}$ and $S'_{\mathsf{out}} = (S^{\mathsf{out}} \cap V(G')) \cup \{z\}$. We now apply Lemma 9 using G', S'_{in}, and S'_{out} to find $\lceil \ell/2 \rceil$ node-disjoint directed concentric cycles, all going in the same direction, and all vertices of S'_{in} are strictly enclosed by the innermost cycle, and all vertices of S'_{out} are not enclosed by the outermost cycle. Clearly, each of these cycles exists in G, all vertices of S^{in} are strictly enclosed by the innermost cycle, and all vertices of S^{out} are not enclosed by the outermost cycle. ◀

We are now ready to obtain the proof of Theorem 3.

Proof of Theorem 3. By Lemma 10, we can finish the construction of the crossbar with the same argument as in [6]. ◀

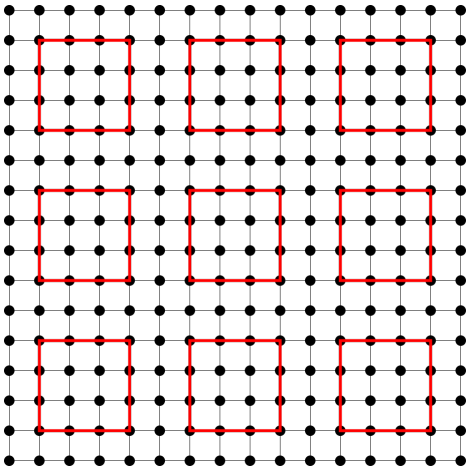


Figure 3 Decomposition of the grid minor.

5 Graphs of Bounded Genus

In this section we describe an algorithm to construct a flat grid minor Γ of large enough size in graphs of bounded genus. The following is implicit in the work of Chekuri and Sidiropoulos [10].

▶ **Lemma 11.** *Let G be an undirected graph of Euler genus $g \geq 1$, with treewidth $t \geq 1$. There is a polynomial time algorithm that computes a $r' \times r'$-grid as a minor, with $r' = \Omega\left(\frac{t}{g^3 \log^{5/2} n}\right)$. Furthermore, the algorithm does not require a drawing of G as part of the input.*

We need to find a flat grid minor for our purpose. Thomassen in [28] shows that if a graph of genus g contains a $m \times m$-grid as a minor, then it contains a $k \times k$ flat grid minor, where $m > 100k\sqrt{g}$. With some minor modifications, we can use this result to obtain the following.

▶ **Lemma 12.** *Let G be an undirected graph of Euler genus $g \geq 1$, and let H be a $m \times m$ grid minor of G. Let $k < \frac{m}{100\sqrt{g}}$ be an integer. Then one can compute a $k \times k$ flat grid minor of G in polynomial time.*

Proof. Thomassen in [28] shows that in order to find the desired flat grid minor, it is enough to construct a family of pairwise disjoint subgraphs $Q_1, Q_2, \cdots, Q_{2g+2}$ of H, satisfying the following conditions.

(1) Each Q_i is a $k \times k$ sub-grid of H.

(2) For any i, j with $1 \leq i < j \leq 2g+2$, we have the following. If x_i and x_j are on the outer cycles of Q_i and Q_j respectively, and they have neighbors $y_i \in V(H) \setminus V(Q_i)$ and $y_j \in V(H) \setminus V(Q_j)$ respectively, then H has a path P_{ij} from x_i to x_j such that $V(P_{ij}) \cap \left(\bigcup_{r=1}^{2g+2} V(Q_r)\right) = \{x_i, x_j\}$.

Since we have $m > 100k\sqrt{g}$, this construction can be easily done as shown in Figure 3, and thus one of the Q_i's is flat, as desired. ◀

▶ **Lemma 13.** *Let G be an undirected graph of Euler genus $g \geq 1$, with treewidth $t \geq 1$. There exists a polynomial time algorithm that computes a $r \times r$-grid as a minor, with $r = \Omega\left(\frac{t}{g^3\sqrt{g}\log^{5/2} n}\right)$. Moreover, the algorithm does not require a drawing of G as part of the input.*

Proof. This is immediate by Lemmas 11 and 12. ◀

Note that computing a large grid minor in the graph is not enough. We need to make sure that a large number of terminals can reach the interface of the grid minor. The following Lemma will provide for us the desired grid minor. The proof of this Lemma is deferred to Appendix A.

▶ **Lemma 14.** *Let $\mathcal{F}$ be some minor-closed family of graphs, let $\alpha \leq 1$, and $\beta > 0$. Suppose that there exists a polynomial-time algorithm which given, some $G' \in \mathcal{F}$ and some α-node-well-linked set X' in G', outputs some $r' \times r'$ flat grid minor Γ' in G', for some $r' = \Omega(\alpha|X'|/\beta)$. Then there exists a polynomial-time algorithm which, given some $G \in \mathcal{F}$ and some α-node-well-linked set X in G, outputs some $r \times r$ flat grid minor Γ in G, for some integer $r = \Omega(\alpha|X|/\beta)$, and a family of λr node-disjoint paths in G connecting X to the interface of Γ, for some constant $0 < \lambda < 1$.*

▶ **Lemma 15.** *Let G be an undirected graph of genus g, and let $\alpha \leq 1$. Let X be an α-node-well-linked set in G. One can, in polynomial time, find some $r \times r$ flat grid minor Γ in G, for some integer $r = \Omega\left(\frac{\alpha|X|}{g^3\sqrt{g}\log^{5/2} n}\right)$, and a family of λr node-disjoint paths connecting X and the interface of Γ, for some $0 < \lambda \leq 1$.*

Proof. This is immediate by combining Lemmas 13 and 14. ◀

Now by Lemmas 2, 5, and 15 we get the following result.

▶ **Theorem 16.** *Let G be a graph of genus g. There is a polynomial time randomized algorithm that, with high probability, achieves an $\Omega\left(\frac{1}{g^3\sqrt{g}\log^{5/2}(n)}\right)$-approximation with congestion 5 for Sym-Dir-NDP instances in G.*

6 Minor Free Graphs

In this section we present the flat grid minor construction for minor-free graphs. We first consider the problem on nearly embeddable graphs, and we extend our solution to arbitrary minor-free graphs by dealing with sums of constant size.

6.1 Nearly Embeddable Graphs

In this subsection we work on nearly embeddable graphs. First we reduce the problem to the case of zero apices.

▶ **Lemma 17** (Reduction to $(0, g, k, p)$-nearly embeddable graphs)**.** *Suppose that there is a polynomial time algorithm for Sym-Dir-NDP in $(0, g, k, p)$-nearly embeddable graphs that achieves a β-approximation with congestion c. Then there is a polynomial time algorithm for Sym-Dir-NDP in (a, g, k, p)-nearly embeddable graphs that achieves a β/a-approximation with congestion c.*

Proof. Let G be an (a, g, k, p)-nearly embeddable graph, and suppose that we are given a Sym-Dir-NDP instance $M = \{s_1t_1, \cdots, s_mt_m\}$ in G. Let $A \subseteq V(G)$ be the set of apices in G. Let $G' = G \setminus A$. Clearly, G' is a $(0, g, k, p)$-nearly embeddable graph. Let $M' \subseteq M$ be the subset of source-terminal pairs that do not intersect A. M' forms a Sym-Dir-NDP instance in G', and thus we can get a β-approximation solution S' with congestion c. Since $|M| \leq |M'| + a$, we have that S' is a β/a-approximation solution with congestion c for M in G, as desired. ◀

Next we provide an algorithm for Sym-Dir-NDP in $(0, g, k, p)$-nearly embeddable graphs. Let G be an $(0, g, k, p)$-nearly embeddable graph, and let S be the bounded genus subgraph of G on the surface; that is, S is obtained from G by deleting all vortices. Let $X \subseteq V(G)$ be the set of terminals. Note that by using Lemma 2 we can reduce the problem to the case where X is α-well-linked for some $\alpha \leq 1$. The following is implicit in [17].

▶ **Lemma 18** (Demaine and Hajiaghayi [17]). *Let $t \geq 1$ be the treewidth of G^{UN}, and let t' be the treewidth of S^{UN}. Then we have $t' \geq \frac{t}{(p+k)^3}$.*

▶ **Lemma 19.** *One can in polynomial time find a $r \times r$ flat grid minor Γ in G^{UN},with $r = \Omega\left(\frac{t}{g^3\sqrt{g}(p+k)^3 \log^{5/2} n}\right)$.*

Proof. By Lemma 18 we have that the treewidth of S^{UN} is at least $\frac{t}{(p+k)^3}$. S^{UN} is a graph of Euler genus g, and thus by Lemma 13 we get the desired result. ◀

▶ **Lemma 20.** *One can in polynomial time find some $r \times r$ flat grid minor Γ in G^{UN}, for some integer $r = \Omega\left(\frac{t}{g^3\sqrt{g}(p+k)^3 \log^{5/2} n}\right)$, and a family of r node-disjoint paths connecting X and the interface of Γ.*

Proof. This is immediate by Lemmas 19 and 14. ◀

Now by combining Lemmas 2, 3, 20, the crossbar construction and routing scheme in Section 4, we get the following result.

▶ **Lemma 21.** *Let G be a $(0, g, k, p)$-nearly embeddable graph. There is a polynomial time randomized algorithm that, with high probability, achieves an $\Omega\left(\frac{1}{g^3\sqrt{g}(p+k)^3 \log^{5/2} n}\right)$-approximation with congestion 5 for Sym-Dir-NDP instances in G.*

▶ **Theorem 22.** *Let G be a (a, g, k, p)-nearly embeddable graph. There is a polynomial time randomized algorithm that, with high probability, achieves an $\Omega\left(\frac{1}{ag^3\sqrt{g}(p+k)^3 \log^{5/2} n}\right)$-approximation with congestion 5 for Sym-Dir-NDP instances in G.*

Proof. This follows immediately by Lemmas 21 and 17. ◀

6.2 Dealing with h-sums

In this subsection we are going to prove Lemma 4. Let G be a minor-free graph, with treewidth t. Let $X \subseteq V(G)$ be the set of terminals. The following is implicit in [18].

▶ **Lemma 23** ([18]). *Let G_1, G_2 be two undirected graphs, and let G_3 be an h-sum of G_1 and G_2 for some integer $h > 0$. Let t_1, t_2, and t_3 be the treewidth of G_1, G_2, and G_3 respectively. Then we have $t_3 \leq \max\{t_1, t_2\}$.*

We are now ready to prove our result for computing flat grid minors in minor-free graphs.

Proof of Lemma 4. By using Theorem 1, we get a decomposition of G^{UN} into h-sums of h-nearly-embeddable graphs. By Lemma 23, we have that at least one summand G' has treewidth at least t. Now G' is a h-nearly-embeddable graph with treewidth t, and thus by Lemma 20 we get the desired flat grid minor. ◀

References

1 Alok Aggarwal, Amotz Bar-Noy, Don Coppersmith, Rajiv Ramaswami, Baruch Schieber, and Madhu Sudan. Efficient Routing and Scheduling Algorithms for Optical Networks. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '94, pages 412–423, Philadelphia, PA, USA, 1994. Society for Industrial and Applied Mathematics.

2 B. Awerbuch, R. Gawlick, T. Leighton, and Y. Rabani. On-line Admission Control and Circuit Routing for High Performance Computing and Communication. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, FOCS '94, pages 412–423, Washington, DC, USA, 1994. IEEE Computer Society.

3 Andrei Z. Broder, Alan M. Frieze, and Eli Upfal. Existence and Construction of Edge-Disjoint Paths on Expander Graphs. *SIAM Journal on Computing*, 23(5):976–989, 1994.

4 Chandra Chekuri and Alina Ene. Poly-logarithmic Approximation for Maximum Node Disjoint Paths with Constant Congestion. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 326–341, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics.

5 Chandra Chekuri and Alina Ene. The all-or-nothing flow problem in directed graphs with symmetric demand pairs. *Mathematical Programming*, 154(1):249–272, December 2015.

6 Chandra Chekuri, Alina Ene, and Marcin Pilipczuk. Constant Congestion Routing of Symmetric Demands in Planar Directed Graphs. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 7:1–7:14, 2016.

7 Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. The All-or-nothing Multicommodity Flow Problem. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 156–165, New York, NY, USA, 2004. ACM.

8 Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. Multicommodity Flow, Well-linked Terminals, and Routing Problems. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 183–192, New York, NY, USA, 2005. ACM.

9 Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. An $O(\sqrt{n})$ approximation and integrality gap for disjoint paths and unsplittable flow. *Theory of Computing*, 2:2006, 2006.

10 Chandra Chekuri and Anastasios Sidiropoulos. Approximation algorithms for Euler genus and related problems. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 167–176. IEEE, 2013.

11 Julia Chuzhoy, Venkatesan Guruswami, Sanjeev Khanna, and Kunal Talwar. Hardness of Routing with Congestion in Directed Graphs. In *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*, STOC '07, pages 165–178, New York, NY, USA, 2007. ACM.

12 Julia Chuzhoy and David H. K. Kim. On Approximating Node-Disjoint Paths in Grids. In Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 187–211, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

13 Julia Chuzhoy, David HK Kim, and Shi Li. Improved approximation for node-disjoint paths in planar graphs. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 556–569. ACM, 2016.

14 Julia Chuzhoy, David HK Kim, and Rachit Nimavat. New hardness results for routing on disjoint paths. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 86–99. ACM, 2017.

15 Julia Chuzhoy and Shi Li. A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 233–242. IEEE, 2012.

16 Erik D Demaine, Mohammad Taghi Hajiaghayi, and Ken-ichi Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *Foundations of Computer Science, 2005. FOCS 2005. 46th Annual IEEE Symposium on*, pages 637–646. IEEE, 2005.

17 Erik D Demaine and MohammadTaghi Hajiaghayi. Linearity of grid minors in treewidth with applications through bidimensionality. *Combinatorica*, 28(1):19–36, 2008.

18 Erik D Demaine, MohammadTaghi Hajiaghayi, Naomi Nishimura, Prabhakar Ragde, and Dimitrios M Thilikos. Approximation algorithms for classes of graphs excluding single-crossing graphs as minors. *Journal of Computer and System Sciences*, 69(2):166–195, 2004.

19 Thor Johnson, Neil Robertson, P.D. Seymour, and Robin Thomas. Directed Tree-Width. *Journal of Combinatorial Theory, Series B*, 82(1):138–154, 2001.

20 R M Karp. On the complexity of combinatorial problems. *Networks*, 5:45–68, 1975.

21 Ken-ichi Kawarabayashi and Anastasios Sidiropoulos. Polylogarithmic approximation for minimum planarization (almost). In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 779–788. IEEE, 2017.

22 Stavros G Kolliopoulos and Clifford Stein. Approximating disjoint-path problems using greedy algorithms and packing integer programs. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 153–168. Springer, 1998.

23 D. Peleg and E. Upfal. Constructing disjoint paths on expander graphs. *Combinatorica*, 9(3):289–313, September 1989.

24 Prabhakar Raghavan and Eli Upfal. Efficient Routing in All-optical Networks. In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 134–143, New York, NY, USA, 1994. ACM.

25 B. Reed. Introducing Directed Tree Width. *Electronic Notes in Discrete Mathematics*, 3(Supplement C):222–229, 1999. 6th Twente Workshop on Graphs and Combinatorial Optimization.

26 N. Robertson and P.D. Seymour. Graph Minors. XIII. The Disjoint Paths Problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

27 Neil Robertson and Paul D Seymour. Graph minors. XVI. Excluding a non-planar graph. *Journal of Combinatorial Theory, Series B*, 89(1):43–76, 2003.

28 Carsten Thomassen. A simpler proof of the excluded minor theorem for higher surfaces. *Journal of Combinatorial Theory, Series B*, 70(2):306–311, 1997.

A Missing Proofs

Proof of Lemma 14. Let t be the treewidth of G. Since X is α-node-well-linked in G, we have that $t = \Omega(\alpha|X|)$. Let Γ_0 be an $r' \times r'$ flat grid minor in G, for some $r' = \Omega(\alpha|X|/\beta)$. If there is a family of λr_0 node-disjoint paths connecting X and the the interface of Γ_0, then we are done. Otherwise, we will find an *irrelevant* vertex; that is a vertex $v \in V(G)$ such that deleting v from G does not affect the well-linkedness of X. Therefore, we can delete v from G, and recursively call the process for finding flat grid minors, until we get the desired one.

Suppose that there is not a family of λr_0 node-disjoint paths connecting X and the interface of Γ_0. First we find a $r'_0 \times r'_0$ sub-grid Γ'_0 of Γ_0 such that $r'_0 = O(r_0)$ and Γ'_0 contains at most $\frac{\lambda r_0}{\alpha}$ terminals. For any minor H of G, and for every $v \in V(H)$, let $\eta(v) \subseteq V(G)$ be the subset of vertices in G corresponding to v. Let also $X_H = X \cap \eta(H)$. Since there is not a family of λr_0 node-disjoint paths connecting X and the interface of Γ_0, we can find a cut $C \subseteq E(G)$ in G, separating X_{Γ_0} and the interface of Γ_0, with $|C| < \lambda r_0$. Now let $A_1, A_2, \ldots, A_m$ be the connected components of $G \setminus C$ that contain vertices of X_{Γ_0} (See Figure 4). We may assume w.l.o.g. that $|V(A_1)| \geq |V(A_2)| \geq \ldots \geq |V(A_m)|$. Now let $Y, Z \subset X$ be two disjoint subsets of X of equal size such that $X_{A_1} \subset Y$ and $X_{A_i} \subset Z$ for any $i \in \{2, 3, \ldots, m\}$. Since X is α-node-well-linked, there exist a family $\mathcal{P}$ of $|Y|$ paths from Y

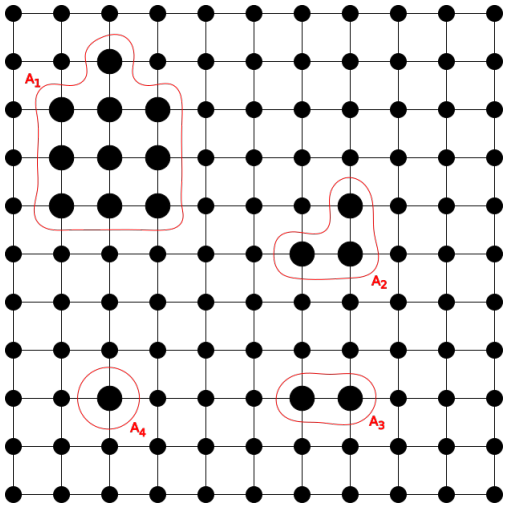


Figure 4 The connected components of $G \setminus C$ in Γ'_0.

to Z such that no vertex is in more than $1/\alpha$ of these paths. However, we have $X_{A_1} \subset Y$ and $X_{A_i} \subset Z$ for any $i \in \{2, 3, \ldots, m\}$, and thus we have $|V(X_{A_2}) \cup \ldots \cup V(X_{A_m})| \leq |C|\frac{1}{\alpha} < \frac{\lambda r_0}{\alpha}$. Therefore, we can find a $\frac{r_0}{4} \times \frac{r_0}{4}$ sub-grid Γ'_0 of Γ_0 such that Γ'_0 does not intersect X_{A_1}, and moreover there are at most $\frac{\lambda r_0}{\alpha}$ number of terminals in $\eta(\Gamma'_0)$.

If there is a family of $\lambda r'_0$ node-disjoint paths connecting X and the interface of Γ'_0, then we are done. Otherwise, we find an irrelevant vertex. We use a similar technique as in [21]. Let Γ''_0 be the $r''_0 \times r''_0$ sub-grid of Γ'_0 obtained by deleting the first and last $r'_0/4$ rows and columns of Γ'_0. By the construction, we know that Γ''_0 contains at most $\frac{\lambda r_0}{\alpha}$ terminals. We may assume w.l.o.g. that r'_0 is a power of 2, and thus r''_0 is a power of 2 as well. We construct a hierarchical partitioning of Γ''_0 into smaller sub-grids as follows. For every $i, j \in \{1, 2, \ldots, r''_0\}$, let $v_{i,j}$ be the vertex in the i'th row and j'th column of Γ''_0. For any $i, j, h \in \{1, 2, \ldots, r''_0\}$, let

$$H_{i,j,h} = \bigcup_{a=\max\{1,i-h-1\}}^{\min\{i+h,r''_0\}} \bigcup_{b=\max\{1,j-h-1\}}^{\min\{j+h,r''_0\}} \{v_{a,b}\}.$$

We also define $\ell(H_{i,j,h}) = 2\mathrm{h}$. For every $q \in \{0, 1, \ldots, \log r''_0\}$, we define two partitions of Γ''_0 into $q \times q$ sub-grids as follows. Let

$$\mathcal{H}_{q,1} = \bigcup_{i=0}^{r''_0/2^{q+1}} \bigcup_{j=0}^{r''_0/2^{q+1}} \{H(i2^{q+1}, j2^{q+1}, 2^q)\},$$

and

$$\mathcal{H}_{q,2} = \bigcup_{i=0}^{r''_0/2^{q+1}} \bigcup_{j=0}^{r''_0/2^{q+1}} \{H(i2^{q+1} + 2^q, j2^{q+1} + 2^q, 2^q)\}.$$

Let $\mathcal{H} = \bigcup_{q=0}^{\log r''_0} \bigcup_{i=1}^{2} \mathcal{H}_{q,i}$. For every $H \in \mathcal{H}$, let $w(H)$ be the number of terminals in $\eta(H)$. Let also $w(\Gamma''_0)$ be the number of terminals in $\eta(\Gamma''_0)$. We say that some $H \in \mathcal{H}$ is *dense* if $w(H) \geq \ell(H)/100$. Let $\delta(\Gamma''_0)$ be the interface of Γ''_0. We say that some $v \in V(\Gamma''_0)$ is *good* if v is not contained in any dense $H \in \mathcal{H}$, and there is no terminals in $\eta(v)$. First we

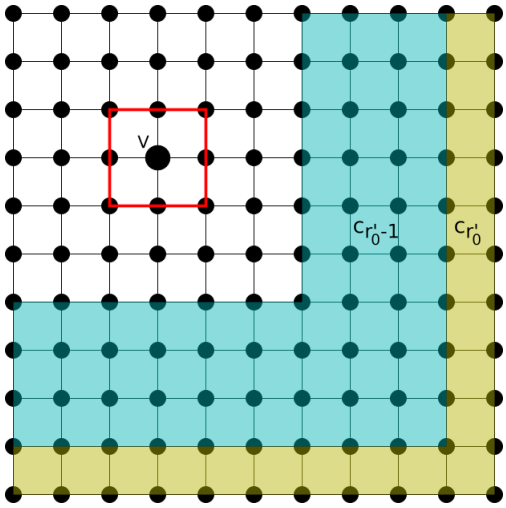


Figure 5 Sets C_q.

show that there exists a good vertex in Γ''_0. We count the number of vertices in Γ''_0 that are contained in at least one dense $H \in \mathcal{H}$. Let $\mathcal{H}_{q,j} \in \mathcal{H}$ for some $q \in \{0, \ldots, \log r''_0\}$ and $j \in \{1, 2\}$, and let $H \in \mathcal{H}_{q,j}$. H is dense if and only if $w(H) \geq \ell(H)/100 = 2^{q+1}/100$. We know that $w(\Gamma''_0) \leq r''_0/10000$, and thus if $2^{q+1} > r''_0/100$, then there are no dense $H \in \mathcal{H}_{q,j}$. Now suppose that $2^{q+1} \leq r''_0/100$, and thus $q < \log r''_0 - 7$. Let $i \in \{8, \ldots, \log r''_0\}$, and let $q = \log r''_0 - i$. Let $H' \in \mathcal{H}_{q,1}$. We have that $\ell(H') = 2^{q+1} = r''_0/2^{i-1}$. In order for H' to be dense it must be that $w(H') \geq \frac{\ell(H)}{100} = \frac{r''_0}{100 \cdot 2^{i-1}}$. Note that we have $w(\Gamma''_0) \leq r''_0/10000$, and therefore there can be at most $2^{i-1}/100$ dense $H' \in \mathcal{H}_{q,1}$. With a similar argument, we can show that there can be at most $2^{i-1}/100$ dense $H' \in \mathcal{H}_{q,2}$. Now we have

$$\begin{aligned}\left|\bigcup_{H \in \mathcal{H}: H \text{ is dense}} H\right| &\leq 2 \cdot \sum_{i=8}^{\log r''_0} \left(\frac{r''_0}{2^{i-1}}\right)^2 \cdot \frac{2^{i-1}}{100} \\ &= \frac{(r''_0)^2}{50} \cdot \sum_{i=8}^{\log r''_0} \frac{1}{2^{i-1}} \\ &< \frac{(r''_0)^2}{50}.\end{aligned}$$

This means that there exist at least $\frac{49(r''_0)^2}{50}$ vertices in Γ''_0 that are not contained in any dense $H \in \mathcal{H}$, and since there are at most $r''_0/10000$ terminals in $\eta(\Gamma''_0)$, there must exist a good vertex in Γ''_0, as desired. Furthermore, this vertex can be found in polynomial time. Let $v \in V(\Gamma''_0)$ be a good vertex.

We claim that vertices in $\eta(v)$ are irrelevant. For every $q \in \{0, 1, \ldots, \log r''_0\}$ and $i \in \{1, 2\}$, let $H_{q,i} \in \mathcal{H}_{q,i}$ be a sub-grid that contains v. By the construction, for every $q \in \{0, 1, \ldots, \log r''_0\}$, we have that either $d_{\Gamma''_0}(v, \delta(H_{q,1})) \geq 2^{q-1}$ or $d_{\Gamma''_0}(v, \delta(H_{q,2})) \geq 2^{q-1}$. Let $B_q \in \{H_{q,1}, H_{q,2}\}$ be such that $d_{\Gamma''_0}(v, \delta(B_q)) \geq 2^{q-1}$. For every $q \in \{1, \ldots, \log r''_0\}$, let $C_q = B_q \setminus B_{q-1}$, and let also $C_{\log r''_0+1} = V(\Gamma'_0) \setminus V(\Gamma''_0)$ (See Figure 5).

Let $Y, Z \subset X$ be two disjoint subsets of X of equal size. Since X is α-node well-linked, we know that there exists a family $\mathcal{P}$ of $|Y|$ paths from Y to Z such that no vertex is in more than $1/\alpha$ of these paths. If none of these paths use v, then we are done. Otherwise, we try to re-route these paths to obtain a new family $\mathcal{P}'$ of paths, such that no path is using

v, and no vertex is in more than $1/\alpha$ of the paths in $\mathcal{P}'$. First we look at the paths $P \in \mathcal{P}$ with both endpoints outside of Γ'_0; that is the endpoints of P do not belong to $\eta(\Gamma'_0)$. Let $\mathcal{P}^* \subseteq \mathcal{P}$ be the set of all such paths. We re-route them in a way such that they do not intersect $\eta(\Gamma''_0)$. Note that by the construction, at most $\lambda r'_0$ of paths in $\mathcal{P}^*$ can intersect $\eta(\Gamma'_0)$. For these paths, we can re-route their intersection with $\eta(\Gamma'_0)$ in $\eta(\Gamma'_0) \setminus \eta(\Gamma''_0)$, and thus they will not intersect $\eta(\Gamma''_0)$. Now let $\mathcal{P}^{**} \subseteq \mathcal{P}$ be the set of paths with one endpoint outside of $\eta(\Gamma'_0)$, and one endpoint inside of $\eta(\Gamma'_0)$. Let $P = (a_1, a_2, \ldots, a_p) \in \mathcal{P}^{**}$, where $a_1 \notin \eta(\Gamma'_0)$ and $a_p \in \eta(\Gamma'_0)$. Let $a_f \in V(P)$ be the first intersection of P and $\eta(\Gamma'_0)$; that is $f \in \{1, 2, \ldots, p\}$ is the minimum number such that $a_f \in \eta(\Gamma'_0)$. Let $P' = (a_f, \ldots, a_p)$. We replace P with P' in $\mathcal{P}$. Note that again there are at most $\lambda r'_0$ such paths in $\mathcal{P}$. Now we are only dealing with paths with both endpoints in $\eta(\Gamma'_0)$. For all such paths, we use an inductive argument to re-route them. For any $i, j \in \{1, 2, \ldots, \log r''_0 + 1\}$, let $\mathcal{P}_{i,j} \subseteq \mathcal{P}$ be the paths with one endpoint in $\eta(C_i)$, and the other endpoint in $\eta(C_j)$. By the construction, for any $i \in \{1, 2, \ldots, \log r''_0 + 1\}$, we know that there are at most $2^i/20$ terminals in $\eta(C_i)$, and thus $|\mathcal{P}_{i,i}| \leq 2^i/20$. For all such paths, we can re-route them such that they stay inside C_i. We start with $\mathcal{P}_{\log r''_0+1, \log r''_0+1}$, and re-route all these paths such that they only use vertices in $C_{\log r''_0+1}$. Again, by the construction, we have that $\left| \bigcup_{j=1}^{\log r''_0} \mathcal{P}_{\log r''_0+1, j} \right| \leq r''_0/10$. For all $P \in \bigcup_{j=1}^{\log r''_0} \mathcal{P}_{\log r''_0+1, j}$, similar to the paths in $\mathcal{P}^{**}$, we can replace them with paths with one endpoint on the boundary of $C_{\log r''_0}$, and recursively follow the same argument for paths with both endpoints in $\eta\left(\bigcup_{j=1}^{\log r''_0} C_j\right)$ and so on. Therefore, by applying the same re-routing pattern, we can get a new set of paths $\mathcal{P}'$ such that no path uses vertex v, as desired.

Now let $G_1 = G \setminus v$. Since v is an irrelevant vertex in G, we have that X is α-node-well-linked in G_1, and thus we have that the treewidth of G_1 is $\Omega(\alpha|X|)$. Therefore, we can find a $r'_1 \times r'_1$ flat grid minor Γ_1 in G_1, for some $r'_1 = \Omega\left(\frac{\alpha|X|}{\beta}\right)$. If there exists a family of $\lambda r'_1$ node-disjoint paths connecting X and the interface of Γ_1, we are done. Otherwise, we recursively follow the same approach to find an irrelevant vertex v_1 in G_1, and let $G_2 = G_1 \setminus v_1$ and so on. This recursive call stops in $O(n)$ steps, because for each $i \geq 1$, G_i is a graph of treewidth $\alpha|X|$. Therefore, for some $j \geq 1$, we can find a $r_j \times r_j$ flat grid minor Γ_j of G_j, for some $r_j = \Omega\left(\frac{\alpha|X|}{\beta}\right)$, such that there exists a family of λr_j node-disjoint paths connecting X and the interface of Γ_j. Note that Γ_j is also a flat grid minor of G, and this completes the proof. ◀

Rainbow Coloring Hardness via Low Sensitivity Polymorphisms

Venkatesan Guruswami
Carnegie Mellon University, Pittsburgh, PA, USA
guruswami@cmu.edu

Sai Sandeep
Carnegie Mellon University, Pittsburgh, PA, USA
spallerl@andrew.cmu.edu

Abstract

A k-uniform hypergraph is said to be r-rainbow colorable if there is an r-coloring of its vertices such that every hyperedge intersects all r color classes. Given as input such a hypergraph, finding a r-rainbow coloring of it is NP-hard for all $k \geq 3$ and $r \geq 2$. Therefore, one settles for finding a rainbow coloring with fewer colors (which is an easier task). When $r = k$ (the maximum possible value), i.e., the hypergraph is k-partite, one can efficiently 2-rainbow color the hypergraph, i.e., 2-color its vertices so that there are no monochromatic edges. In this work we consider the next smaller value of $r = k - 1$, and prove that in this case it is NP-hard to rainbow color the hypergraph with $q := \lceil \frac{k-2}{2} \rceil$ colors. In particular, for $k \leq 6$, it is NP-hard to 2-color $(k-1)$-rainbow colorable k-uniform hypergraphs.

Our proof follows the algebraic approach to promise constraint satisfaction problems. It proceeds by characterizing the polymorphisms associated with the approximate rainbow coloring problem, which are rainbow colorings of some product hypergraphs on vertex set $[r]^n$. We prove that any such polymorphism $f : [r]^n \to [q]$ must be C-fixing, i.e., there is a small subset S of C coordinates and a setting $a \in [q]^S$ such that fixing $x_{|S} = a$ determines the value of $f(x)$. The key step in our proof is bounding the sensitivity of certain rainbow colorings, thereby arguing that they must be juntas. Armed with the C-fixing characterization, our NP-hardness is obtained via a reduction from smooth Label Cover.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases inapproximability, hardness of approximation, constraint satisfaction, hypergraph coloring, polymorphisms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.15

Category APPROX

Related Version A full version of the paper is available at [18], `https://eccc.weizmann.ac.il/report/2019/094/`.

Funding Research supported in part by NSF grants CCF-1422045, CCF-1526092, and CCF-1814603.

Acknowledgements We thank Amey Bhangale, Joshua Brakensiek, Jakub Opršal, and Xinyu Wu for useful discussions. We would also like to thank anonymous reviewers for helpful comments.

1 Introduction

Graph and hypergraph coloring are one of the most studied problems in Graph Theory and Theoretical Computer Science. Even though there is a simple algorithm to check if a given graph is 2-colorable or not, checking if a 3-uniform hypergraph can be colored with two colors so that no hyperedge is monochromatic is one of the classic NP-hard problems. This raises the question of identifying if 2-coloring is easy on special hypergraphs of interest. For example, if a k-uniform hypergraph is k-partite, i.e., the vertices can be partitioned into k parts so


© Venkatesan Guruswami and Sai Sandeep;
licensed under Creative Commons License CC-BY



that every hyperedge intersects each part, then there are simple algorithms to properly color the hypergraph with two colors. Suppose we know that a k-uniform hypergraph is promised to be $k-1$-partite, can we color it with two colors?

An equivalent way to formulate this question is in terms of rainbow coloring. A k-uniform hypergraph is said to be *r-rainbow colorable* if there is a coloring of vertices with r colors such that all the r colors appear in every edge. Unlike usual coloring, rainbow coloring becomes harder as we have more colors. Note that r-partiteness is the same thing as r-rainbow colorability. As mentioned above, a k-uniform hypergraph that is promised to be k-rainbow colorable can be efficiently colored with two colors. One big hammer approach for this is to use semidefinite programming and find a unit vector for each vertex such that the sum of the vectors in each edge sum to zero, and then use random hyperplane rounding. But the 2-coloring can also be performed by a simple random walk algorithm – start with an arbitrary coloring, and as long as there is a monochromatic edge, pick an arbitrary one and flip the color of a random vertex in it. This process will converge to a 2-coloring in a quadratic number of iterations with high probability [26].

If we relax the k-rainbow colorability assumption slightly to that of $(k-1)$-rainbow colorability, there are no known efficient algorithms for 2-coloring. It is tempting to conjecture that in fact this task is hard (in fact, even if we are allowed c colors for any constant c; this was shown assuming the V Label Cover conjecture in [9]). If we relax the rainbow colorability assumption further, Austrin, Bhangale, and Potukuchi proved that it is NP-hard to 2-color a k-uniform hypergraph when it is promised to be $(k-2\sqrt{k})$-rainbow colorable [2]. They also showed that it is NP-hard to 2-color a 4-uniform hypergraph even if it is 3-rainbow colorable. In this work, we focus on hardness results for the $(k-1)$-rainbow colorable case, as this promise is the closest to k-partiteness which makes 2-coloring easy. While we can't show hardness of 2-coloring, we show that rainbow coloring with $\lceil \frac{k-2}{2} \rceil$ colors is hard. Formally, our main result is the following.

▶ **Theorem 1.** *Fix an integer $k \geq 4$. Given a k-uniform hypergraph that is promised to be $(k-1)$-rainbow colorable, it is NP-hard to rainbow color it with $\lceil \frac{k-2}{2} \rceil$ colors.*

As a corollary, we also get the following, which extends the similar result of [2] for the $k = 4$ case (their techniques did not generalize beyond the 4-uniform case).

▶ **Theorem 2.** *For $k \leq 6$, given a k-uniform hypergraph that is promised to be $()k-1)$-rainbow colorable, it is NP hard to 2-color it.*

1.1 Techniques

There have been broadly three lines of attack on proving hardness for graph and hypergraph coloring problems.

1. The first line of work gives reductions from Label Cover analyzed using Fourier-analytic techniques of the sort originally pioneered by Håstad [19]. Early applications of this method showed strong hardness results for coloring 2-colorable hypergraphs of low uniformity with any constant number of colors [15, 20, 22, 31]. This approach, augmented with the invariance principle of [28] and some of its extensions such as [13,27,33], was used to prove further hardness results for hypergraph coloring [5, 16] and strong conditional hardness results for graph coloring [13]. These methods usually also prove a stronger statement about finding independent sets in the graphs or hypergraphs. For rainbow

coloring, it is proved in [16] by combining together many of these techniques that a $(k/2)$-rainbow colorable k-uniform hypergraph cannot be colored with any constant number of colors in polynomial time unless P = NP.

2. A less extensive line of work proceeds via combinatorial gadgets that are analyzed using ideas based on the chromatic number of Kneser graphs and similar results. The first exemplar of this approach was the hardness of $O(1)$-coloring 2-colorable 3-uniform hypergraphs shown in [14]. Unlike the analytic results for 4-uniform hypergraphs mentioned above, this result does not show hardness of finding large independent sets. (This was later shown in [23] using the analytic approach, albeit under the d-to-1 conjecture.) A few recent results have revived this combinatorial approach, by re-deriving and improving some previous hardness results for hypergraph coloring using simpler proofs [3, 6].
3. The third and most recent line of work adapts the universal algebraic method behind the complexity classification of constraint satisfaction problems that culminated in the resolution of the Feder-Vardi CSP dichotomy conjecture [11, 35]. Here, the coloring problem is viewed as a Promise Constraint Satisfaction Problem (PCSP), and its associated "polymorphisms" are then analyzed. [1] In the cases when the polymorphisms are severely limited, one can show hardness via a reduction from Label Cover. The approach to study PCSP using polymorphisms originated in [4] and was used to show hardness results for graph and hypergraph coloring in [8]. The algebraic theory was further developed significantly in [12] leading among other results to a proof of NP-hardness of 5-coloring 3-colorable graphs. Recently, [24] and [34] used topological ideas to make further progress.

In this work, we follow the algebraic approach to prove Theorem 1. In fact, our main motivation is to understand Promise CSPs better. A promise CSP (defined formally in Section 2) is a relaxation of the traditional CSP where one is allowed to find an assignment that satisfies a relaxed version of the predicates underlying the CSP. Approximate graph coloring with more colors than the promised chromatic number is a quintessential example of a promise CSP. Rainbow coloring with fewer colors also naturally falls in this framework. As proved in [10, 12], as with normal CSP, the complexity of a promise CSP is captured by its associated polymorphisms. Polymorphisms (defined formally in Section 2) of a PCSP are ways to combine multiple solutions of an instance satisfying the stronger predicate to obtain a solution to the instance satisfying the weaker predicate. The high-level principle behind the algebraic approach is that the problem should be easy when it has a rich enough set of polymorphisms that include functions with strong symmetries, and hard when all its polymorphisms are somehow skewed and lack symmetries. This has been fully established for CSPs – when there are polymorphisms which obey weak near-unanimity, the CSP is polytime solvable, and otherwise NP-complete.[2] The hardness part of this dichotomy is easier and was known for a while [25]; the much harder algorithmic part was established only recently in [11, 35].

For promise CSPs, which form a much richer class, our current understanding is rather limited, for both the algorithmic and hardness sides. It is not clear (to even conjecture) what kind of lack in symmetries in the polymorphisms might dictate hardness, and how one might show the corresponding hardness. A simple (but rather limited) sufficient condition

[1] The proof in [13] also implicitly studies polymorphisms and proves that they must have a small number of coordinates with sizeable influence and thus are not too symmetric. This influence-type characterization interfaces better with Unique Games or other highly structured forms of Label Cover.

[2] For the case of Boolean CSPs, the CSP is hard iff all polymorphisms are essentially at most unary, i.e., either the dictator function, its complement or a constant function.

for hardness is when all the polymorphisms are dictators that depend on a single coordinate. In [4], it has been proved that if all the polymorphisms of a PCSP are juntas[3], then the PCSP is NP-hard. This is the basis of the hardness results for $(2+\epsilon)$-SAT [4] and 3-coloring graphs that admit a homomorphism to C_k for any fixed odd integer k [24]. The recent hardness of 5-coloring 3-colorable graphs in [12] proceeds by showing that the absence of arity 6 polymorphisms with the so-called Olšák symmetry implies NP-hardness, and then verifying that 3 vs. 5-coloring lacks such polymorphisms.

It turns out that the polymorphisms of rainbow coloring can have Olšák symmetries *and* be non-juntas. We will get around this by proving that these polymorphisms are C-fixing in the sense that there exists a constant number of coordinates and an assignment to them such that if we fix these coordinates to the assignment, the value of the function is fixed. This is also studied as certificate complexity in Boolean function analysis [1]. We then prove that if the polymorphisms of a PCSP are C-fixing, then the PCSP is NP-hard.

In order to prove that the polymorphisms have low certificate complexity, we use the connection between sensitivity and certificate complexity of functions. These two ways of characterizing the complexity of functions are well studied in the context of Boolean functions. It is worth emphasizing that for our purposes, all we need is to show that low sensitivity (even sensitivity 2 suffices) implies constant certificate complexity, and thus we are not interested in optimal gaps between sensitivity and certificate complexity. The famous sensitivity vs. block sensitivity conjecture [29] states that these two parameters are in fact polynomially related. In one of the earliest works related to this problem, Simon [32] proved that certificate complexity is at most exponential in sensitivity. We extend this to larger domains and then use it to prove that the polymorphisms that we study have low certificate complexity. We remark that in a striking breakthrough, Huang [21] recently proved the sensitivity vs block sensitivity conjecture for Boolean domains.

The second step is to then use the C-fixing property to show NP-hardness of the PCSP. This is done by the usual paradigm of reducing from Label Cover using polymorphism tests (better known as long code tests) of functions associated with vertices of the Label Cover instance. A more structured form of the C-fixing property where the C variables are fixed to the same value, is used in [10] to show NP-hardness of certain Boolean PCSPs. However, in order to prove NP-hardness using our more general notion of C-fixing, we end up needing stronger properties of the Label Cover instance. As a result, our reduction is from the *smooth* Label Cover problem that was introduced and shown to be NP-hard in [22], and has found many applications in inapproximability since.

A natural question is to understand how far we can push these techniques. Our NP hardness reduction from smooth Label Cover works when the polymorphisms of the PCSP in hand are C-fixing for some constant C. As k increases, the polymorphisms of PCSP of 2-coloring a k-uniform hypergraph that is promised to be $(k-1)$-rainbow colorable get richer. When k is at most 6, the polymorphisms are C-fixing. At $k=7$, we show that there is a polymorphism that is not C-fixing for any constant C. In fact, one would need C to be linear in the arity of polymorphisms which also rules out using smooth Label Cover with very strong soundness.

1.2 Prior work on rainbow coloring and related problems

Various notions of approximate coloring with rainbow colorability guarantees have been studied in the literature. Bansal and Khot [5] prove that when the input hypergraph is promised to be almost k-rainbow colorable, it is Unique Games hard to color it with

[3] A C-junta is a function that depends on at most C inputs.

$O(1)$ colors. Sachdeva and Saket [30] establish NP-hardness of $O(1)$ coloring a k-uniform hypergraph when it is promised to be almost $(k/2)$-rainbow colorable. This was extended by Guruswami and Lee [16] to perfectly $(k/2)$-rainbow colorable hypergraphs. Guruswami and Saket [17] prove similar results assuming stronger forms of rainbow colorability in the completeness case. In [2], Austrin, Bhangale, and Potukuchi proved that it is NP-hard to 2-color a k-uniform hypergraph when it is promised to be $(k-2\sqrt{k})$-rainbow colorable. On the other hand, when the hypergraph is promised to be $(k-\sqrt{k})$-rainbow colorable, Bhattiprolu, Guruswami and Lee [7] give algorithms to color the hypergraph with two colors that miscolors at most $k^{-\Omega(k)}$ fraction of edges; this beats the 2^{k+1} fraction achieved by random coloring that is the best possible for general 2-colorable hypergraphs [19]. Brakensiek and Guruswami [9] put forth a problem called V label cover (to possibly serve as a perfect completeness variant surrogate for Unique games), and under its conjectured inapproximability proved that it is hard to color a k-uniform $(k-1)$-rainbow colorable hypergraph with $O(1)$ colors.

A related notion of hypergraph coloring is strong coloring where we color a k-uniform hypergraph with $s > k$ colors such that in any edge, all the k vertices are colored with distinct colors. Brakensiek and Guruswami [8] prove that it is NP-hard to 2-color a k-uniform hypergraph that is promised to be strongly colorable with $\lceil \frac{3k}{2} \rceil$ colors. Assuming the V Label Cover conjecture, it is hard to $O(1)$-color k-uniform hypergraphs with strong chromatic number at most $k + \sqrt{k}$ [9].

1.3 Outline

We start with a few notations and definitions in Section 2. In Section 3, we study polymorphisms of rainbow coloring. We first prove a result on sensitivity and certificate complexity and use it to prove properties of polymorphisms of the PCSP that we are studying. Then, we use these in Section 4 to prove NP hardness. Finally, we conclude in Section 5 by mentioning some open questions.

2 Preliminaries

2.1 Notations

We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. Vectors are represented using bold face letters. We abuse the notation of k-ary relation A to use it both as a set $A \subseteq [q]^k$ and indicator function $A : [q]^k \to \{0, 1\}$.

2.2 PCSP and Polymorphisms

We will now formally define CSP, PCSP, and Polymorphisms.

▶ **Definition 3** (CSP). *Given a k ary relation $A : [q]^k \to \{0, 1\}$ over $[q]$, the Constraint Satisfaction problem associated with A takes input as a set of variables $V = \{a_1, a_2, \ldots, a_n\}$ which are to be assigned values from $[q]$. There are m constraints $(e_1, e_2, \ldots, e_m)$ each consisting of $e_i = ((e_i)_1, (e_i)_2, \ldots, (e_i)_k) \subseteq V^k$ that indicate that the corresponding assignment should belong to A. The problem is to check if we can satisfy all the constraints or not.*

In general, we can have multiple relations $A_1, A_2, \ldots, A_m$, and different constraints can use different relations. We denote such a CSP by CSP$(A_1, A_2, \ldots, A_m)$.

Promise CSP (PCSP) is a gap or promise version of CSP. Here, we have a pair of relations such that one is a relaxed form of other and given a CSP instance, and the objective is to decide if there is a satisfying assignment from stronger relation or we cannot even satisfy

the CSP using the relaxed relation. One canonical example of PCSP is the promise graph coloring: Given a graph G, distinguish between the case that G can be 3-colored versus G cannot even be colored with five colors. We can formally define PCSP as below:

▶ **Definition 4** (Promise CSP). *In the promise CSP problem, we have a set of pairs of relations $(A_1, B_1), (A_2, B_2), \ldots, (A_m, B_m)$ such that for every i, A_i is a subset of $[q_1]^{k_i}$ and B_i is a subset of $[q_2]^{k_i}$. Furthermore, there is a homomorphism $h : [q_1] \to [q_2]$ such that, for all i, $x \in A_i$ implies $h(x) \in B_i$ for all $x \in [q_1]^{k_i}$. Given a CSP$(A_1, A_2, \ldots, A_m)$ instance, the goal is to distinguish between the two cases:*

1. *There is a solution to the instance assigning values from $[q_1]$ that satisfies every contraint when viewed as CSP$(A_1, A_2, \ldots, A_m)$.*
2. *There is no solution to the instance assigning values from $[q_2]$ that satisfies every constraint when viewed as CSP$(B_1, B_2, \ldots, B_m)$.*

We now turn our attention towards rainbow coloring which is the PCSP that we study in this paper. In RAINBOW(k, r, q) problem, the input is a k-uniform hypergraph. The goal is to distinguish between the cases when the hypergraph is rainbow colorable with r colors and when it cannot be rainbow colorable with q colors. More formally, we can define the problem as below:

▶ **Definition 5** (RAINBOW(k, r, q)). *In the RAINBOW(k, r, q) promise CSP, $q \leq r \leq k$, we have the relation pair (A, B) defined as follows:*

- *$A : [r]^k \to \{0, 1\} : A(x_1, x_2, \ldots, x_k) = 1$ if and only if $\{x_1, x_2, \ldots, x_k\} = [r]$.*
- *$B : [q]^k \to \{0, 1\} : B(y_1, y_2, \ldots, y_k) = 1$ if and only if $\{y_1, y_2, \ldots, y_k\} = [q]$.*

Note that we need q, r to be at most k since we cannot rainbow color a k-uniform hypergraph with more than k colors. We also need the condition that $q \leq r$ for the promise problem to make sense: If the hypergraph is r rainbow colorable, we can infer that it is already $q < r$ rainbow colorable too. Thus, the promise problem is to identify if the hypergraph is r rainbow colorable or it *cannot even be* rainbow colorable with q colors. Furthermore, in this paper we will be only dealing with near perfect completeness case when hypergraph is $(k-1)$-partite i.e. $r = k - 1$.

Associated with every promise CSP, there are polymorphisms. Polymorphisms capture the symmetries in the PCSP. They are ways in which we combine solutions to obtain new solutions that are still valid.

▶ **Definition 6** (Polymorphisms). *For a PCSP problem $(A, B), A : [q_1]^k \to \{0, 1\}, B : [q_2]^k \to \{0, 1\}$, a polymorphism is a function f from $[q_1]^n \to [q_2]$, where n is the arity of the polymorphism that satisfies the property $(f(\boldsymbol{v}_1), f(\boldsymbol{v}_2), \ldots, f(\boldsymbol{v}_k)) \in B$ for all $(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k)$ such that for all $i \in [n], ((\boldsymbol{v}_1)_i, (\boldsymbol{v}_2)_i, \ldots, (\boldsymbol{v}_k)_i) \in A$.*

In the above, we defined polymorphisms for a single relation PCSP. When the PCSP has multiple relations, the polymorphism should satisfy the above property for all the relations. Informally, the arity n polymorphisms are precisely the functions $f : [q_1]^n \to [q_2]$ such that for every $k \times n$ matrix M with elements from $[q_1]$ whose columns are satisfying tuples of A, the k tuple obtained by applying f to the rows of M should be in B. We refer the reader to [10, 12] for a detailed introduction to PCSP and various examples of polymorphisms.

We now direct our attention to polymorphisms of RAINBOW(k, r, q). By definition, the polymorphisms of hypergraph coloring PCSPs turn out to be colorings of certain tensor product hypergraphs. Fix n to be arity of the polymorphisms. We can infer that the polymorphisms of RAINBOW(k, r, q) are proper q-rainbow colorings of the following k-uniform hypergraph $\mathtt{RH}_n(k, r)$:

- The vertex set of hypergraph is the set $V = [r]^n$.
- A k element set $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$, where each $\mathbf{v}_i \in [r]^n$ is an edge if and only if for every $j \in [n]$, the set $\{(\mathbf{v}_1)_j, (\mathbf{v}_2)_j, \ldots, (\mathbf{v}_k)_j\}$ is equal to $[r]$.

That is, a set of k vectors from $[r]^n$ forms an edge if in the matrix obtained by plugging these vectors as rows, all the r elements from $[r]$ occur in every column.

2.3 Complexity measures of functions

Finally, we define the notions of sensitivity and C-fixing of functions.

▶ **Definition 7** (Sensitivity at $\mathbf{x}$). *For a function $f : [r]^n \to [q]$, and an input $\boldsymbol{x} \in [r]^n$, the sensitivity of f at $\boldsymbol{x}$, denoted by $S(f, \boldsymbol{x})$ is defined as the number of coordinates i such that changing $\boldsymbol{x}$ at i can change the value of f i.e. $S(f, \boldsymbol{x}) = \left|\{i \in [n] | \exists a : f(\boldsymbol{x}) \neq f(\boldsymbol{x} : x_i \leftarrow a)\}\right|$.*

▶ **Definition 8** (Sensitivity). *The sensitivity of a function $f : [r]^n \to [q]$, denoted by $S(f)$ is defined as the maximum sensitivity of f over all $\boldsymbol{x}$ in $[r]^n$ i.e. $S(f) = \max_{\boldsymbol{x}} S(f, \boldsymbol{x})$.*

▶ **Definition 9** (C-fixing). *A function f from $[r]^n$ to $[q]$ is said to be C-fixing for some integer C if there exists a set $S = \{s_1, s_2, \ldots, s_C\} \subseteq [n]$ and a vector $\boldsymbol{\alpha} \subseteq [r]^n$ such that $f(\boldsymbol{x}) = c$ whenever $\boldsymbol{x}_{s_i} = \boldsymbol{\alpha}_{s_i}$ for all integers $1 \leq i \leq C$, for some fixed $c \in [q]$.*

3 Polymorphisms

In this section, we will analyze the properties of polymorphisms of rainbow coloring. In order to do so, we will prove that low sensitivity implies low certificate complexity. Using this, we will establish that the polymorphisms for $\mathsf{RAINBOW}(k, k-1, \lceil \frac{k-2}{2} \rceil)$ are C-fixing. Along the way, we will study rainbow colorings of various hypergraphs related to $\mathtt{RH}_n(k, r)$. Finally, we will show that our techniques cannot prove hardness of $\mathsf{RAINBOW}(7, 6, 2)$ by presenting a polymorphism that is not C-fixing for any constant C.

3.1 Sensitivity vs certificate complexity

We extend a lemma of [32] that proves that if a function has low sensitivity then the function is C fixing, to larger domains. The proof is along the same lines as the original proof.

▶ **Lemma 10.** *Let $f : [r]^n \to [q]$ be a function with sensitivity s, and let $b \in [q]$ such that $f^{-1}(b)$ is non empty. Then, $\left|f^{-1}(b)\right| \geq r^{n-s}$.*

Proof. Fix s, and induct on n. The case $n = s$ is trivial. Let $\mathbf{x} \in [r]^n$ be such that $f(\mathbf{x}) = b$. Since $s < n$, there is a coordinate in $\mathbf{x}$ that is not sensitive. Without loss of generality, let it be 1, and let $\mathbf{x} = (x_1, \mathbf{y})$. As the first coordinate is not sensitive for $\mathbf{x}$, we can conclude that $f(\alpha, \mathbf{y}) = b$ for all $\alpha \in [r]$.

Consider the set of functions $g_i : [r]^{n-1} \to [q], g_i(\mathbf{u}) = f(i, \mathbf{u}), i \in [r]$. Note that for each such g_i, the set $g_i^{-1}(b)$ is non-empty. In addition, for every $i \in [r]$, sensitivity of g_i is at most the sensitivity of f. Thus, by induction, we know that each such g_i has at least r^{n-1-s} elements $\mathbf{u}$ in $[r]^{n-1}$ such that $g_i(\mathbf{u}) = b$. Note that every such $\mathbf{u}$ gives $f(i, \mathbf{u}) = b$. By combining over all is, we can conclude that there are at least $r \cdot r^{n-1-s} = r^{n-s}$ elements $\mathbf{x} \in [r]^n$ such that $f(\mathbf{x}) = b$. ◀

▶ **Lemma 11.** *Let $f : [r]^n \to [q]$ be a function with sensitivity $s < n/2$. Then, it is C-fixing for $C = s(r-1)r^{2s+1}$.*

Proof. We will actually prove a stronger statement that f is a C-junta. Let A denote the set of coordinates with non-zero influence in f i.e. the coordinates that are sensitive for some input. Our goal is to upper bound the cardinality of A.

For a function $f : [r]^n \to [q]$, let the set of sensitive edges $E(f)$ be defined as the set of pairs of elements $\mathbf{x}, \mathbf{y} \in [r]^n$ such that $f(\mathbf{x}) \neq f(\mathbf{y})$, and $\mathbf{x}, \mathbf{y}$ differ on exactly one coordinate. From the sensitivity bound on f, we can deduce that

$$|E(f)| \leq s(r-1)r^n \tag{1}$$

Fix an arbitrary coordinate $i \in A$. There are elements $\mathbf{x}, \mathbf{y} \in [r]^n$ such that $x_i = \alpha, y_i = \beta, \alpha \neq \beta, f(\mathbf{x}) \neq f(\mathbf{y})$, and $\mathbf{x}, \mathbf{y}$ differ only in ith coordinate. Define a function $g : [r]^{n-1} \to \{0, 1\}$ as $g(\mathbf{z})$ is 1 if and only if $f(\alpha, \mathbf{z}) = f(\mathbf{x})$, and $f(\beta, \mathbf{z}) = f(\mathbf{y})$ where we use the notation $(\alpha, \mathbf{z})$ to denote the vector in $[r]^n$ obtained by inserting α in ith position to $\mathbf{z} \in [r]^{n-1}$. Now, since $f(\alpha, \mathbf{z})$ and $f(\beta, \mathbf{z})$ are both sensitive to at most s coordinates, $g(\mathbf{z})$ is sensitive to at most $2s$ coordinates. Also note that $g^{-1}(1)$ is non-empty. Thus, by Lemma 10, we can conclude that $|g^{-1}(1)|$ is at least r^{n-1-2s}. In other words, each sensitive coordinate contributes at least r^{n-2s-1} edges to $E(f)$. Thus, we can conclude that

$$|E(f)| \geq |A| r^{n-2s-1} \tag{2}$$

Combining Equation (1) and Equation (2), we get

$$|A| \leq s(r-1)r^{2s+1} \tag{3}$$

which proves the required claim. ◀

3.2 Low sensitivity polymorphisms of rainbow coloring

We now turn our attention towards our main goal in this section: to show that polymorphisms of $\mathsf{RAINBOW}(k, k-1, q)$ are C-fixing for $q = \lceil \frac{k-2}{2} \rceil$. As we have already mentioned earlier, the polymorphisms of rainbow coloring themselves are rainbow colorings of certain tensor product hypergraphs. To be precise, the n-ary polymorphisms of $\mathsf{RAINBOW}(k, r, q)$ are precisely q-rainbow colorings of $\mathtt{RH}_n(k, r)$. Thus our new goal is to prove that for any integer $q \geq 2$, any q-rainbow coloring of $\mathtt{RH}_n(2q+2, 2q+1)$ is a C-fixing function.

In order to achieve this, we will first define certain hypergraphs similar to $\mathtt{RH}_n(k, r)$.

▶ **Definition 12.** *$\mathtt{H}_n(r, s) = (V, E)$ is a r-uniform hypergraph where the vertex set V is equal to $[r]^n$. A set of vectors $(\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_r)$ is an edge if and only if*

1. *In every coordinate $i \subseteq [n]$, at least $r-1$ elements occur i.e. $\left|\bigcup_j (\boldsymbol{u}_j)_i\right| \geq r-1 \quad \forall i \in [n]$.*
2. *All the r elements occur in at least $n-s$ coordinates i.e. $\left|\bigcup_j (\boldsymbol{u}_j)_i\right| = r$ for at least $n-s$ choices of i in $[n]$.*

The reason behind studying these hypergraphs is that the q-rainbow colorings of $\mathtt{RH}_n(2q+2, 2q+1)$ are very closely related to q-rainbow colorings of $\mathtt{H}_n(2q+1, c)$ for any absolute constant c. In fact if we can prove that q-rainbow colorings of $\mathtt{H}_n(2q+1, c)$ are C-fixing, it implies that q-rainbow colorings of $\mathtt{RH}_n(2q+2, 2q+1)$ are $\max(C, c)$-fixing. This is formally proved in Lemma 16. Thus our modified objective is to argue that q-rainbow colorings of $\mathtt{H}_n(2q+1, c)$ are C-fixing. In order to do so, we inductively relate q-rainbow colorings of $\mathtt{H}_n(t, c)$ and $\mathtt{H}_n(t-1, c-1)$. As a base case, we have the following lemma:

▶ **Lemma 13.** *For all integers $q \geq 2$ and $n \geq 1$, the hypergraph $\mathtt{H}_n(2q-1, 1)$ cannot be rainbow colored with q colors.*

Proof. We will use induction on q. For the case $q = 2$, rainbow coloring with 2 colors is the same as proper coloring the hypergraph with 2 colors. The fact that $\mathtt{H}_n(3, 1)$ cannot be two colored follows from [2] (Lemma 3.2 with $d = 3$).

Suppose for contradiction that f is a valid q-rainbow coloring of $\mathtt{H}_n(2q-1, 1)$. Let $r = 2q - 1$ denote the uniformity of the hypergraphs. Consider the set of r vectors in $[r]^n$: $\{\bigcup_{i\in[r]}(i, i, \ldots, i)\}$. As there are at most $q < r$ colors, some two elements of this set should have same f value. Without loss of generality, let $f(r-1, r-1, \ldots, r-1) = f(r, r, \ldots, r) = c$ for some $c \in [q]$. Consider the $(r-2)$-uniform hypergraph $H = \mathtt{H}_n(r-2, 1)$. Note that every edge in H together with $\mathbf{u} = (r-1, r-1, \ldots, r-1)$ and $\mathbf{v} = (r, r, \ldots, r)$ forms an edge in $\mathtt{H}_n(r, 1)$. Thus, all the $q-1$ colors in $[q] \setminus \{c\}$ occur in every edge of coloring of $\mathtt{H}_n(r-2, 1)$ using f. This implies that we can get a a valid $(q-1)$-rainbow coloring of $\mathtt{H}_n(r-2 = 2(q-1)-1, 1)$ by restricting f to $[r-2]^n$, and replacing the color c using arbitrary color from $[q] \setminus \{c\}$. However, by the induction hypothesis, such a coloring cannot exist, and thus we have arrived at a contradiction. ◀

Now, we will use this to argue about q-rainbow colorings of $\mathtt{H}_n(2q+1, 3)$ via q-rainbow colorings of $\mathtt{H}_n(2q, 2)$. Consider the hypergraph $\mathtt{H}_n(2q, 2)$. A trivial way to q-rainbow color this hypergraph is to pick a coordinate $i \in [n]$, and partition the set $[2q]$ into q disjoint sets of size two, let's say $A_1, A_2, \ldots, A_q$ and assign the value $p \in [q]$ to $f(\mathbf{x})$ for $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ if and only if $x_i \in A_p$. It turns out that this is the only way to q-rainbow color $\mathtt{H}_n(2q, 2)$. We prove it in the lemma below:

▶ **Lemma 14.** *Let f be a q-rainbow coloring of $\mathtt{H}_n(r = 2q, 2)$. Then, there exists an index $i \in [n]$, sets $A_1, A_2, \ldots, A_q \subseteq [r]$ mutually disjoint and each of size 2, such that $f(\boldsymbol{x}) = j$ iff $x_i \in A_j$.*

Proof. First we will prove that the sensitivity of f is at most 1. Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ be an arbitrary vector in $[r]^n$. Consider a $(r-1)$-uniform hypergraph $H(\mathbf{x})$ defined on $([r] \setminus \{x_1\}) \times ([r] \setminus \{x_2\}) \times \ldots \times ([r] \setminus \{x_n\})$. We add a $r-1$ vector set as edge of $H(\mathbf{x})$ if and only if it has at most one coordinate where there are missing elements i.e. all the $[r] \setminus \{x_i\}$ occur in all but one coordinate i, and in that coordinate, at most one value is missing.

Note that $H(\mathbf{x})$ is isomorphic to $\mathtt{H}_n(2q-1, 1)$. From Lemma 13, we know that $H(\mathbf{x})$ cannot be rainbow colored with q colors. Thus, when we view f as a coloring of $H(\mathbf{x})$, there is an edge that has a color missing. Let it be denoted by $e = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{r-1})$. Let j be the coordinate where there is a missing element in e. If there is no coordinate with a missing element, j can be arbitrary. Without loss of generality, let color $1 \subseteq [q]$ be missing in e. Note that $\{\mathbf{x}\} \cup e$ is an edge of $\mathtt{H}_n(r, 1)$, and thus an edge of $\mathtt{H}_n(r, 2)$ as well. Since f is a proper q-rainbow coloring of $\mathtt{H}_n(2q, 2)$, we can conclude that $f(\mathbf{x}) = 1$. In fact, we can actually deduce something stronger. Let $\mathbf{y} \in [r]^n$ such that $\mathbf{x}$ and $\mathbf{y}$ differ on exactly one coordinate $j' \in [r]^n, j' \neq j$. Note that $\{\mathbf{y}\} \cup e$ is also a valid edge of $\mathtt{H}_n(2q, 2)$ since it has at most two coordinates where there are missing elements i.e. j' and j. Thus, $f(\mathbf{y}) = 1 = f(\mathbf{x})$. Thus, for every $\mathbf{x}$, in except for one coordinate, changing the value of the coordinate preserves the value of $\mathbf{x}$. In other words, the sensitivity of f is at most 1.

Using this, we will now prove that f is a dictator. Let i be an influential coordinate of f i.e. there exists $\mathbf{x}, \mathbf{y} \in [r]^n$ differing only in ith coordinate such that $f(\mathbf{x}) \neq f(\mathbf{y})$. We claim that $f(\mathbf{u}) = f(\mathbf{x})$ for all $\mathbf{u} = (u_1, u_2, \ldots, u_n) \in [r]^n$ such that $u_i = x_i$, and $f(\mathbf{u}) = f(\mathbf{y})$ if $u_i = y_i$. We will prove this by induction on the number of coordinates in which $\mathbf{x}$ and $\mathbf{u}$ differ excluding i. Since f has sensitivity at most 1, the only sensitive coordinate of $\mathbf{x}$ and $\mathbf{y}$ is i. Thus, for any $\mathbf{u}$ differing only in one coordinate from $\mathbf{x}$ (other than i) such that $u_i = x_i$ or y_i will have corresponding f value. Suppose that the statement holds for all $\mathbf{u}$ differing from $\mathbf{x}$ in t coordinates excluding i.

Now, let $\mathbf{u}$ differ from $\mathbf{x}$ in $t+1$ coordinates excluding i. We can find $\mathbf{v} \in [r]^n, \mathbf{w} \in [r]^n$ such that $\mathbf{v}$ and $\mathbf{x}$ differ in t coordinates excluding i, $v_i = x_i$; $\mathbf{w}$ and $\mathbf{y}$ differ in t coordinates excluding i, $w_i = y_i$, and one of $\mathbf{v}$ and $\mathbf{w}$ differs from $\mathbf{u}$ in at most one coordinate. By the induction hypothesis, $f(\mathbf{v}) = f(\mathbf{x}), f(\mathbf{w}) = f(\mathbf{y})$. Since $\mathbf{v}$ and $\mathbf{w}$ differ in a single coordinate i, i is the only sensitive coordinate of $\mathbf{v}$ and $\mathbf{w}$. Thus, $f(\mathbf{u})$ is equal to either $f(\mathbf{v})$ or $f(\mathbf{w})$ depending on $u_i = x_i$ or y_i. This completes the inductive proof.

To complete the proof that f is a dictator, we will use this to show that there cannot be two influential coordinates. Suppose that there are two influential coordinates i and j. From the previous argument, we can infer that there are assignments $i_1, i_2, j_1, j_2 \in [r]$ such that assigning these to corresponding coordinates fixes the value of f. Also note that assigning i as i_1 and i_2 fixes f to different values. Similarly, assigning j as j_1 and j_2 fixes f to different values. This gives rise to contradiction since if we set coordinate i to i_1, f should be fixed irrespective of j is equal to j_1 or j_2. Thus, there can be only one influential coordinate for f, or in other words, f is a dictator.

Let p be the dictator coordinate of f i.e. there exists a function $g : [r] \to [q]$ such that $f(\mathbf{x}) = g(x_p)$. From the definition of the hypergraph $\mathtt{H}_n(r, 2)$, for every $j \in [r]$, the set $\{\bigcup_i g(x_i)\} \setminus \{g(x_j)\}$ should be equal to $[q]$. This proves that there exists sets $A_1, A_2, \ldots, A_q \subseteq [r]$ each of size two, and mutually disjoint such that $g(\alpha) = j$ if and only if $\alpha \in A_j$, which proves the required claim. ◀

We finish the chain of inductive arguments by proving a key property of q-rainbow colorings of $\mathtt{H}_n(2q+1, 3)$.

▶ **Lemma 15.** *Let $f : [2q+1]^n \to [q]$ be a q-rainbow coloring of $\mathtt{H}_n(r = 2q+1, 3)$. Then, there exists an index $i \in [n]$, and $\alpha \in [r]$ such that $S(f, \boldsymbol{x}) \leq 2$ for all $\boldsymbol{x} \in [r]^n$ such that $x_i = \alpha$.*

Proof. Let $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in [r]^n$ be an arbitrary vector in $[r]^n$. Similar to the previous lemma, we define the complement hypergraph associated with $\mathbf{x}$. Consider a $(r-1)$-uniform hypergraph $H(\mathbf{x})$ defined on $([r] \setminus \{x_1\}) \times ([r] \setminus \{x_2\}) \times \ldots \times ([r] \setminus \{x_n\})$. We add a $r-1$ vector set as edge of $H(\mathbf{x})$ if and only if it has at most two coordinates where there are missing elements i.e. all the $[r] \setminus \{x_i\}$ occur in all but two coordinates i, and in these two coordinates, at least $r-2$ values occur. Note that $H(\mathbf{x})$ is isomorphic to $\mathtt{H}_n(r-1, 2)$.

We can view $f : [2q+1]^n \to [q]$ as a q-coloring of $H(\mathbf{x})$. If f is not a valid q-rainbow coloring of $H(\mathbf{x})$, by the same argument as in Lemma 14, we can deduce that $S(f, \mathbf{x}) \leq 2$. If f is a valid q-rainbow coloring of $H(\mathbf{x})$, we will use the properties proved in Lemma 14. Let us define a function $g : [r]^n \to [n] \cup \{\perp\}$ such that for a vector $\mathbf{x} \in [r]^n$,

1. If f is a valid q-rainbow coloring of $H(\mathbf{x})$, then Lemma 14 implies that there exists a coordinate $i \in [n]$ such that f is a dictator in ith coordinate in $H(\mathbf{x})$. In this case, set $g(\mathbf{x}) = i$.
2. If f is not a valid q-rainbow coloring of $H(\mathbf{x})$, let $g(\mathbf{x}) = \perp$.

First, we will prove that there exists an index $i \in [n]$ such that $g(\mathbf{x}) \in \{i, \perp\}$ for all $\mathbf{x} \in [r]^n$. Suppose $g(\mathbf{x}) = i \in [n]$, and $g(\mathbf{y}) = j \in [n]$ where $\mathbf{x}, \mathbf{y} \in [r]^n$ and $i \neq j$. Since $g(\mathbf{x}) = i$, there exist sets $S_1, S_2, \ldots, S_n \subseteq [r]$ such that f is a dictator on ith coordinate in $S = S_1 \times S_2 \times \ldots \times S_n \subseteq [r]^n$. In particular, there is a subset $A \subseteq S_i$ such that $|A| = 2$, and $f(\mathbf{x})$, $\mathbf{x} \in S$, is equal to 1 if and only if $x_i \in A$. Similarly, there exist sets $T_1, T_2, \ldots, T_n \subseteq [r]$ such that f is a dictator on jth coordinate in $T = T_1 \times T_2 \times \ldots \times T_n \subseteq [r]^n$. There exists a subset $B \subseteq T_j$ such that $|B| = 2$, and $f(\mathbf{x}), \mathbf{x} \in T$ is equal to $c \neq 1$ if and only if $x_j \in B$ for some $c \in [q]$. Combining the both, let $U_i = S_i \cap T_i$, $|U_i| \geq r - 2 \, \forall \, i \in [n]$. We can deduce

that f is a dictator in both i and j coordinates in $U = U_1 \times U_2 \times \ldots \times U_n$. This implies that f is a constant function in U. Recall that there are two assignments in S_i that make f equal to 1 and two assignments in T_j that make f equal to $c \neq 1$. Thus, $f(\mathbf{x}')$ is equal to 1 for some $\mathbf{x}' \in U$ and $f(\mathbf{y}') = c \neq 1$ for some $\mathbf{y}' \in U$. This contradicts the fact that f is a constant function in U. Thus, there exists an index $i \in [n]$ such that $g(\mathbf{x})$ is either equal to i or is equal to $\perp$ for all $\mathbf{x} \in [r]^n$. Without loss of generality let that be the first coordinate i.e. for all $\mathbf{x} \in [r]^n$, $g(\mathbf{x}) \in \{1, \perp\}$.

Consider the case when $g(\mathbf{x}) = \perp$ for every $\mathbf{x} \in [r]^n$. In this case, we know that $S(f, \mathbf{x}) \leq 2$ for all $\mathbf{x} \in [r]^n$. In particular, we can set α arbitrary and say that $S(f, \mathbf{x}) \leq 2$ whenever $x_1 = \alpha$. So we are only left with the case when there exists a $\mathbf{x} \in [r]^n$ such that $g(\mathbf{x}) = 1$. We will now prove that there exists $\alpha \in [r]$ such that $g(\mathbf{x}) = \perp$ whenever $x_1 = \alpha$, thus proving the required sensitivity bound.

Suppose for contradiction that for every $\alpha \in [r]$, there exists $\mathbf{x} \in [r]^n$ such that $x_1 = \alpha$, and $g(\mathbf{x}) = 1$. Consider a pair $\mathbf{u}, \mathbf{v} \in [r]^n$ such that $u_1 = \alpha, v_1 = \beta \neq \alpha$ and $g(\mathbf{u}) = g(\mathbf{v}) = 1$. Let $\mathbf{u} = (u_1, u_2, \ldots, u_n)$, $S_i = [r] \backslash \{u_i\}$ and f is dictator on 1st coordinate in $S = S_1 \times S_2 \times \ldots \times S_n$. There is a function $h_1 : S_1 \to [q]$ such that $f(\mathbf{x}) = h_1(x_1)$ if $\mathbf{x} \in S$ and $|h_1^{-1}(c)| = 2 \, \forall c \in [q]$. Similarly, let $\mathbf{v} = (v_1, v_2, \ldots, v_n)$, $T_i = [r] \setminus \{v_i\}$ and f is dictator on first coordinate in $T = T_1 \times T_2 \times \ldots \times T_n$. There is a function $h_2 : T_1 \to [q]$ such that $f(\mathbf{x}) = h_2(x_1)$ if $\mathbf{x} \in T$ and $|h_2^{-1}(c)| = 2 \quad \forall c \in [q]$. Let $U_i = S_i \cap T_i$. Note that $U = U_1 \times U_2 \times \ldots U_n$ is non empty and f is dictator on 1st coordinate in U as well. Note that $|U_i| \geq r - 2$ for all $i \in [n]$. Thus, we can conclude that if $\gamma \in U_1$, then $h_1(\gamma) = h_2(\gamma)$.

Applying this to all pairs $\mathbf{u}, \mathbf{v}$ such that $g(\mathbf{u}) = g(\mathbf{v}) = 1$, we can infer that there exists a function $h : [r] \to [q]$ that satisfies the property that for all $\mathbf{x} \in [r]^n$ such that $g(\mathbf{x}) = 1$, let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, $S_i = [r] \setminus \{x_i\}$, $S = S_1 \times S_2 \times \ldots \times S_n$, then $f(\mathbf{y}) = h(y_1)$ for all $\mathbf{y} \in S$. As $r = 2q + 1 > 2q$, there exists $b \in [q]$ such that $|h^{-1}(b)| \geq 3$. Let $\gamma \in [r]$ be such that $h(\gamma) \neq b$. From our assumption that for every $\alpha \in [r]$ there exists $\mathbf{x} \in [r]^n$ such that $g(\mathbf{x}) = 1$ and $x_1 = \alpha$, there exists $\mathbf{u} \in [r]^n$ such that $u_1 = \gamma$ and $g(\mathbf{u}) = 1$. Now, let $\mathbf{u} = (u_1, u_2, \ldots, u_n)$, $S_i = [r] \setminus \{u_i\}$, $S = S_1 \times S_2 \times \ldots \times S_n$, and we know that $f(\mathbf{x}) = h(x_1)$ if $\mathbf{x} \in S$, and $\left|h^{-1}(c) \cap S_1\right| = 2 \, \forall c \in [q]$. However, this contradicts the fact that $h(u_1) = h(\gamma) \neq b$, and $|h^{-1}(b)| = 3$. Thus, there exists $\alpha \in [r]$ such that $g(\mathbf{x}) = \perp$ for all $\mathbf{x} \in [r]^n$ such that $x_1 = \alpha$. ◀

Finally, we will use the previous hypergraph coloring properties to argue about polymorphisms of rainbow coloring.

▶ **Lemma 16.** *There exist constant $C = C(q)$ independent of n such that every $f : [2q+1]^n \to [q]$ that is an n-ary polymorphism of RAINBOW$(2q+2, 2q+1, q)$ i.e. f is a proper q-rainbow coloring of $\mathtt{RH}_n(2q+2, 2q+1)$ is C-fixing.*

Proof. Let $r = 2q + 1$. Let $f : [r]^n \to [q]$ be a polymorphism of RAINBOW$(2q + 2, 2q + 1, q)$. We can view f as a q-rainbow coloring of $\mathtt{H}_n(r, 3)$ as the vertex set of $\mathtt{RH}_n(r + 1, r)$ and of $\mathtt{H}_n(r, 3)$ is equal to $[r]^n$. If it is not a valid q-rainbow coloring, there is an edge in which not all q colors appear. Let that edge be $e = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ and $c \in [q]$ be a missing color in $\{f(\mathbf{v}_1), f(\mathbf{v}_2), \ldots, f(\mathbf{v}_r)\}$. Since this edge is part of $H_n(r, 3)$, except for 3 values of i, for all other i, the set $\big((\mathbf{v}_1)_i, (\mathbf{v}_2)_i, \ldots, (\mathbf{v}_r)_i\big)$ is equal to $[r]$. Let the missing coordinates be the set $S = \{i_1, i_2, i_3\}$. Now consider an element $\mathbf{u}$ of $[r]^n$ such that it has missing values of e in S. From the definition of $\mathtt{RH}_n(r + 1, r)$, we can deduce that the set $e \cup \mathbf{u}$ is an edge of $\mathtt{RH}_n(r + 1, r)$. Since f is a valid q-rainbow coloring of $\mathtt{RH}_n(r + 1, r)$, $f(\mathbf{u})$ is equal to c. Note that this should hold irrespective of what values $\mathbf{u}$ has in coordinates outside S. This proves that f is C-fixing with $C = 3$.

On the other hand if f is a valid q-rainbow coloring of $\mathtt{H}_n(r,3)$, using Lemma 15, we can deduce that there exists an index $i \in [n]$, and $\alpha \in [r]$ such that $S(f,\mathbf{x}) \leq 2$ whenever $x_i = \alpha$. Now, we can consider a function $g : [r]^{n-1} \to [q]$ which on an input $\mathbf{y} \in [r]^{n-1}$, is equal to $f(\mathbf{x}), \mathbf{x} = \mathbf{y}, x_i \leftarrow \alpha \in [r]^n$ i.e. we first insert α in ith position to $\mathbf{y}$ and then apply f. Note that g has sensitivity at most 2. From Lemma 11, we can conclude that g is C-fixing for $C = 2(r-1)\cdot r^5$. In other words, g is fixed by assigning values to a set of C indices. This implies that f is also $C' = C + 1$-fixing since we can first assign ith index to α, then use C-fixing property of g. ◀

3.3 High sensitivity polymorphism of RAINBOW$(7,6,2)$

We show that there exists a function $f : [6]^n \to \{0,1\}$ that is a polymorphism of RAINBOW$(7,6,2)$ that is not C-fixing for any constant C. We start with a dictator but add just enough noise such that the function still remains being a polymorphism, but it is no longer C-fixing. Let $wt(\mathbf{x})$ denote the number of $i \in [n], i > 1$ such that $x_i = 1$. Let $S \subseteq [6]^n$ denote the set of $\mathbf{x} \in [6]^n$ such that $wt(\mathbf{x}) > \frac{2n}{3}$. Let $h : [6]^n \to \{0,1\}$ be noise function defined below. For a given $\mathbf{x} \in [6]^n$, we define $f(\mathbf{x})$ as follows:

1. If $\mathbf{x} \notin S$
   a. If $x_1 \leq 3$, $f(\mathbf{x}) = 0$
   b. Else, $f(\mathbf{x}) = 1$
2. Else $f(\mathbf{x}) = h(\mathbf{x})$.

A choice of noise function that works is inverting the original function: $h(\mathbf{x})$ is defined as 1 if and only if $x_1 \leq 3$.

▶ **Proposition 17.** *The function $f : [6]^n \to \{0,1\}$ defined above is a polymorphism of RAINBOW$(7,6,2)$ and it is not C-fixing for any $C < \frac{n}{3}$.*

Proof. Any polymorphism of RAINBOW$(7,6,2)$ is a proper 2-rainbow coloring of $\mathtt{RH}_n(7,6)$. Recall that rainbow coloring with two colors is the same as standard hypergraph coloring with two colors.

Polymorphism. In any set of 7 vectors E in $[6]^n$ such that all the 6 elements occur in all the coordinates, at most two vectors can be in S. This is because, in any set of three vectors in S, there exists a coordinate in which all three values are equal to 1. Thus, there are vectors $\mathbf{x} \notin S$ with $x_1 \leq 3$ and vector $\mathbf{y} \notin S$ such that $y_1 \geq 3$ in E, which together ensures that E is not monochromatic.

C-fixing. Suppose that there exists a set $T = \{t_1, t_2, \ldots, t_m\} \subseteq [n]$ and $(\alpha_1, \alpha_2, \ldots, \alpha_m) \subseteq [6]^m$ such that $f(\mathbf{x}) = b$ for all $\mathbf{x}$ such that $x_i = \alpha_i$ for all $1 \leq i \leq m$, for some fixed $b \in \{0,1\}$. We will prove that $|T| \geq \frac{n}{3}$. Suppose for contradiciton that $|T| < \frac{n}{3}$. First, if $1 \notin T$, we can set all coordinates outside T to be equal to $\beta \neq 1$, and in this case $f(\mathbf{x}) = x_1$, which cannot be fixed if $1 \notin T$. Thus $1 \in T$. Next, if all the coordinates outside T are all equal to 1, then $f(\mathbf{x})$ is equal to noise function, which is different from the case when the rest are equal to $\beta \neq 1$. Thus, if f is indeed a C-fixing function, for the C-fixing assignment, the value of f should be independent of the assignment to the coordinates outside T. However, that is not the case as the value of f changes when we set all the coordinates outside T to be 1 or $\beta \neq 1$. ◀

4 NP-Hardness

In this section, we will use the properties of polymorphisms proved so far to argue about NP hardness of rainbow coloring PCSP. We will prove the below theorem:

▶ **Theorem 18.** *Suppose that there exists a constant C such that for all integers $n \geq 1$, every n-ary polymorphism of RAINBOW$(k, k-1, q)$ is C-fixing. Then, the corresponding decision problem RAINBOW$(k, k-1, q)$ is NP hard.*

Before delving into the proof of Theorem 18, we first mention that this theorem together with Lemma 16 implies Theorem 1. In Lemma 16, we have proved that for every $q \geq 2$, the polymorphisms of RAINBOW$(2q+2, 2q+1, q)$ are C-fixing. This fact combined with Theorem 18 implies that RAINBOW$(2q+2, 2q+1, q)$ is NP hard for every $q \geq 2$. This already proves Theorem 1 when k is even. When k is odd, we can combine Lemma 14 and Lemma 16 to prove that the polymorphisms of RAINBOW$(k = 2q+1, 2q, q)$ are C-fixing. We can combine this with Theorem 18 to prove Theorem 1 when k is odd.

The rest of this section is dedicated to proving Theorem 18. Like various other hardness of approximation results, we will use the standard label cover with long code framework. We reduce *smooth* label cover introduced in [22] to rainbow coloring PCSP. First we define Label Cover problem below:

▶ **Definition 19** (Label Cover)**.** *In an instance of Label Cover problem, we are given a tuple $(G = (L, R, E), \Sigma, \Pi)$ where*

1. *G is a bipartite multi graph between vertex sets L and R*
2. *Each vertex in G has to be assigned a label from Σ*
3. *For each edge $e = (u, v) \in E$, there is a projection constraint Π_e from u to v that is a function from Σ to itself. This corresponds to a constraint between u and v.*

A labelling of graph is a function $\sigma : L \cup R \to \Sigma$ that assigns a label to each vertex of G. A labelling σ is said to satisfy constraint Π_e if and only if $\Pi_e(\sigma(u)) = \sigma(v)$.

We refer to L and R as left and right vertices respectively. We are now ready to define Gap Label Cover.

▶ **Definition 20** ($(1, \epsilon_{LC})$ Gap Label Cover)**.** *In $(1, \epsilon_{LC})$ Gap Label Cover, we are given a Label Cover instance $(G = (L, R, E), \Sigma, \Pi)$, and the goal is to distinguish between the following two cases:*

1. *There is a labelling $\sigma : G \to \Sigma$ that satisfies all the constraints.*
2. *No labelling can satisfy ϵ_{LC} fraction of constraints.*

As mentioned earlier, we need stronger properties of the Label Cover instance that we are starting with. One such property is smoothness.

▶ **Definition 21** (Smoothness)**.** *A Label Cover instance $(G = (L, R, E), \Sigma, \Pi)$ is said to be $(J, \epsilon)-$ smooth if for any vertex $u \in L$ and a set of labels $S \subseteq \Sigma, |S| \leq J$, over a uniformly random neighbor $v \in R$, $Pr(\left|\bigcup_{s \in S} \Pi_{u,v}(s)\right| < |S|) \leq \epsilon$.*

The following is a special case of Theorem 1.17 in [33].

▶ **Theorem 22.** *For every $\epsilon, \epsilon_{LC} > 0$ and $J \in \mathbb{Z}_+$, there exists $n = n(\epsilon, \epsilon_{LC}, J)$ such that $(1, \epsilon_{LC})$ Gap Label Cover with $|\Sigma| = n$ that is promised to be (J, ϵ)-smooth is NP hard.*

We now prove Theorem 18.

Reduction. We start with $(1, \epsilon_{LC})$ Gap Label Cover instance $(G = (L, R, E), \Sigma, \Pi)$ that is promised to be (C, ϵ)-smooth, for ϵ and ϵ_{LC} to be set later, and output a PCSP instance. The reduction described here is the same as the general one from Label Cover to PCSP in e.g. [12]. Let n denote the label size $n = |\Sigma|$. For each vertex $v \in L \cup R$, we add a set of nodes K_v of size $[k-1]^n$, indexed by n length vectors. We add two types of constraints:

1. Coloring constraints: Inside every vertex of the Label Cover instance, we add the following constraints among the $[k-1]^n$ nodes. We add the constraint that the promise relation should be satisfied in the set T of k nodes $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$ in $[k-1]^n$, if for every $i \in [n]$, the set $\{\bigcup_j (\mathbf{x}_j)_i\}$ has cardinality $k-1$.
2. Equality constraints: For every constraint $\Pi_e : u \to v$ of the Label Cover, we add a set of equality constraints between nodes $\mathbf{x} \in K_u$, $\mathbf{y} \in K_v$ if for all $i \in [n]$, $\mathbf{x}_i = \mathbf{y}_{\Pi_e(i)}$.

Note that the Coloring constraints give rise to rainbow colorings of k-uniform hypergraphs. It is yet unclear how we can justify adding equality constraints. One way to handle the equality constraints is to have a single node for all the vertices corresponding to equality constraint. However, this fails if we want to add a coloring constraint that involves two copies of the same vertex. A neater way to get around this is to argue that adding equality constraints does not change the set of polymorphisms, and thus the hardness of the predicate remains the same with or without equality constraints. This simple fact is proved in e.g. [10, 12].

Completeness. If the label cover instance is satisfiable, then PCSP instance that is being output can be satisfied by assignment from $[k-1]$. Suppose $\sigma : L \cup R \to \Sigma$ is a labeling that satisfies all the constraints of the Label Cover. For every vertex $\mathbf{x} \in K_u$ corresponding to the vertex $u \in L \cup R$, we can assign the value $\mathbf{x}_{\sigma(u)}$. In other words, in every long code, we are assigning corresponding dictator function. The coloring constraints are defined precisely such that this assignment satisfies the constraints. The equality constraints also follow since the labeling σ satisfies all the constraints of the Label Cover.

Soundness. If the Label Cover is not ϵ_{LC} satisfiable, we need to show that there is no assignment of the PCSP instance in $[q]$ that satisfies all the constraints. Taking contrapositive, if there is an assignment in $[q]$ to PCSP instance that satisfies all the constraints, then we will prove that there is an assignment to the Label Cover instance that can satisfy a c fraction of constraints, for an absolute constant c. Taking $\epsilon_{LC} < c$, we can arrive at a contradiction, thus proving that there is no assignment in $[q]$ to PCSP that satisfies all the constraints.

Let $f_v : [k-1]^n \to [q]$ denote the assignment to the PCSP instance that satisfies all the constraints for $v \in L \cup R$. From the coloring constraints, we can infer that f_v is a n-ary polymorphism of $\mathsf{RAINBOW}(k, k-1, q)$. Thus, it is C-fixing for a constant C independent of n.

For every vertex $v \in L \cup R$ of the Label Cover instance, we will assign a set of labels $A(v) \subseteq [n]$. For vertices v in L, $A(v)$ is the C-fixing set. Since the Label Cover instance is smooth, we will only consider the constraints where all these labels go to distinct labels on the right under projections. We can set the smoothness parameter ϵ to be 0.1 for example, and we will be left with $\frac{9}{10}$ fraction of original constraints. We will prove that there is an assignment that satisfies a c fraction of these constraints, for an absolute constant c, which will prove the original soundness claim. Thus in all the remaining constraints, the set of labels in $A(v)$ go to distinct labels on the right. Thus, for a vertex $v \in R$, each constraint (u, v) gives rise to C coordinates $\Pi_{u,v}(A(u))$. Note that these C coordinates are in fact C-fixing for v for every constraint (u, v). For a given $v \in R$, there are several such C-fixing sets. Let the set of these C-fixing sets be denoted by $B(v) = \{S_1, S_2, \ldots\}$ where each $S_i \subseteq [n]$ is a C-fixing set of f_v. Now we define $A(v)$ for $v \in R$ to be the set of union of an arbitrary fixed maximal disjoint sets in $B(v)$.

In order to prove that there is a good labeling to the Label Cover, we assign a label to every vertex v from $A(v)$ uniformly at random and prove that it satisfies a constant fraction of constraints with non-zero probability. We will, in fact, show that the random assignment satisfies a constant fraction of constraints in expectation. We prove this in two steps. First, we show that for every constraint (u, v) of the Label Cover, there exists $x \in A(u), y \in A(v)$ such that $\Pi_{u,v}(x) = y$. This follows from the definitions of $A(v)$: suppose the projection of $A(u)$ is disjoint from $A(v)$. In that case, we can add the projection of $A(u)$ to $A(v)$ to get a larger set in v, which contradicts the fact that $A(v)$ is the maximal such union of disjoint projections. This implies that the uniformly random labelling satisfies each constraint (u, v) of Label Cover with probability at least $\frac{1}{|A(u)||A(v)|}$.

To complete the proof, we need to bound the sizes of $A(v)$. As we have already mentioned, for $v \in L, |A(v)| \leq C$. We bound the size of $A(v)$ for vertices v in R using the below lemma.

▶ **Lemma 23.** *Suppose $f : [k-1]^n \to q$ is a polymorphism of RAINBOW$(k, k-1, q)$. Let $A_1, A_2, \ldots, A_t$ be mutually disjoint subsets of $[n]$ such that each of them is a C-fixing set of f. Then, $t < k$.*

Proof. First note that all the A_is should fix f to the same value in $[q]$ since otherwise, the vector $\mathbf{u} \in [k-1]^n$ that has all the fixing sets in A_is is forced to be equal to multiple colors in $[q]$ at the same time. Let all the A_is be C-fixing with respect to value $b \in [q]$ i.e. for each $i \in [t]$, there exists an assignment to A_i such that if the value of $\mathbf{x}$ in A_i is equal to the assignment, then the value of $f(\mathbf{x})$ is equal to b irrespective of values of coordinates outside A_i. If $t \geq k$, we can find $\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_k$ such that all $[k-1]$ occur in every coordinate, and $\mathbf{y}_i$ has the fixing assignment of A_i. This implies that $f(\mathbf{y}_i) = b$ for all i. However, note that $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\}$ is an edge of $\mathtt{RH}_n(k, k-1)$, and thus if f is a polymorphism of RAINBOW$(k, k-1, q)$, all the $[q]$ elements should occur in $\{f(\mathbf{y}_1), f(\mathbf{y}_2), \ldots, f(\mathbf{y}_k)\}$. This is a contradiction since for all i, $f(\mathbf{y}_i) = b$. ◀

From the lemma, we can infer that the cardinality of $A(v)$ for $v \in R$ is at most kC. Combining this with the above, we can deduce that there is an assignment that satisfies $\frac{1}{kC^2}$ fraction of constraints, which is a constant fraction of constraints, independent of n.

5 Conclusion

In this paper, we have proved that given a k-uniform hypergraph that is promised to be $(k-1)$-rainbow colorable, it is NP hard to rainbow color it with $\lceil \frac{k-2}{2} \rceil$ colors. As a corollary, we can deduce that for $k \leq 6$, it is NP hard to 2-color a k-uniform hypergraph that is promised to be $(k-1)$-rainbow colorable. An immediate question is whether RAINBOW$(7, 6, 2)$ is NP hard. It would be interesting to get an efficient algorithm though we believe it is unlikely. In Section 3.3, we have provided a polymorphism of RAINBOW$(7, 6, 2)$ that is not C-fixing. The polymorphisms for this PCSP also have other symmetries (in the form of identities) discussed in [12].

However, it should be noted the polymorphism that we have given in Section 3.3 is very far from symmetric, it seems that we should decode to the unique special coordinate. What we are missing here is a characterization of lack of symmetries that works well with Label Cover to give NP-hardness. We believe that resolving the hardness of this particular PCSP can shed light on identifying criteria for lack of symmetries that imply hardness, beyond C-fixing. Another direction to explore is whether we can further strengthen the completeness in our result. More concretely, given a k-rainbow colorable k-uniform hypergraph, can we efficiently rainbow color it with 3 colors?

References

1. Andris Ambainis, Krišjānis Prūsis, and Jevgēnijs Vihrovs. Sensitivity Versus Certificate Complexity of Boolean Functions. In *Proceedings of the 11th International Computer Science Symposium on Computer Science — Theory and Applications - Volume 9691*, CSR 2016, pages 16–28, 2016. doi:10.1007/978-3-319-34171-2_2.
2. Per Austrin, Amey Bhangale, and Aditya Potukuchi. Improved Inapproximability of Rainbow Coloring. *CoRR*, abs/1810.02784, 2018.
3. Per Austrin, Amey Bhangale, and Aditya Potukuchi. Simplified inpproximability of hypergraph coloring via t-agreeing families. *CoRR*, abs/1904.01163, 2019. arXiv:1904.01163.
4. Per Austrin, Venkatesan Guruswami, and Johan Håstad. $(2+\epsilon)$-Sat Is NP-hard. *SIAM J. Comput.*, 46(5):1554–1573, 2017. doi:10.1137/15M1006507.
5. Nikhil Bansal and Subhash Khot. Inapproximability of Hypergraph Vertex Cover and Applications to Scheduling Problems. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, pages 250–261, 2010. doi:10.1007/978-3-642-14165-2_22.
6. Amey Bhangale. NP-hardness of coloring 2-colorable hypergraph with poly-logarithmically many colors. In *45th International Colloquium on Automata, Languages, and Programming*, pages 15:1–15:11, 2018.
7. Vijay V. S. P. Bhattiprolu, Venkatesan Guruswami, and Euiwoong Lee. Approximate Hypergraph Coloring under Low-discrepancy and Related Promises. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, pages 152–174, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.152.
8. Joshua Brakensiek and Venkatesan Guruswami. New Hardness Results for Graph and Hypergraph Colorings. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 14:1–14:27, 2016. doi:10.4230/LIPIcs.CCC.2016.14.
9. Joshua Brakensiek and Venkatesan Guruswami. The Quest for Strong Inapproximability Results with Perfect Completeness. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, pages 4:1–4:20, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.4.
10. Joshua Brakensiek and Venkatesan Guruswami. Promise Constraint Satisfaction: Structure Theory and a Symmetric Boolean Dichotomy. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1782–1801, 2018. doi:10.1137/1.9781611975031.117.
11. Andrei A. Bulatov. A Dichotomy Theorem for Nonuniform CSPs. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.
12. Jakub Bulín, Andrei A. Krokhin, and Jakub Oprsal. Algebraic approach to promise constraint satisfaction. *CoRR*, abs/1811.00970, 2018. STOC 2019, to appear. arXiv:1811.00970.
13. Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional Hardness for Approximate Coloring. *SIAM J. Comput.*, 39(3):843–873, 2009.
14. Irit Dinur, Oded Regev, and Clifford D. Smyth. The Hardness of 3-Uniform Hypergraph Coloring. *Combinatorica*, 25(5):519–535, 2005. doi:10.1007/s00493-005-0032-4.
15. Venkatesan Guruswami, Johan Håstad, and Madhu Sudan. Hardness of Approximate Hypergraph Coloring. *SIAM J. Comput.*, 31(6):1663–1686, 2002. doi:10.1137/S0097539700377165.
16. Venkatesan Guruswami and Euiwoong Lee. Strong Inapproximability Results on Balanced Rainbow-Colorable Hypergraphs. *Combinatorica*, 38(3):547–599, 2018. doi:10.1007/s00493-016-3383-0.
17. Venkatesan Guruswami and Rishi Saket. Hardness of Rainbow Coloring Hypergraphs. In *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*, pages 33:33–33:15, 2017. doi:10.4230/LIPIcs.FSTTCS.2017.33.

18 Venkatesan Guruswami and Sai Sandeep. Rainbow coloring hardness via low sensitivity polymorphisms. *Electronic Colloquium on Computational Complexity (ECCC)*, 2019. URL: `https://eccc.weizmann.ac.il/report/2019/094/`.

19 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. `doi:10.1145/502090.502098`.

20 Jonas Holmerin. Vertex cover on 4-regular hyper-graphs is hard to approximate within 2-epsilon. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 544–552, 2002.

21 Hao Huang. Induced subgraphs of hypercubes and a proof of the Sensitivity Conjecture. *arXiv preprint*, 2019. `arXiv:1907.00847`.

22 Subhash Khot. Hardness results for coloring 3-colorable 3-uniform hypergraphs. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 23–32. IEEE, 2002.

23 Subhash Khot and Rishi Saket. Hardness of Finding Independent Sets in 2-Colorable and Almost 2-Colorable Hypergraphs. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1607–1625, 2014.

24 Andrei A. Krokhin and Jakub Oprsal. The complexity of 3-colouring H-colourable graphs. *arXiv preprint*, 2019. `arXiv:1904.03214`.

25 Miklós Maróti and Ralph McKenzie. Existence theorems for weakly symmetric operations. *Algebra universalis*, 59(3):463–489, December 2008. `doi:10.1007/s00012-008-2122-9`.

26 Colin McDiarmid. A Random Recolouring Method for Graphs and Hypergraphs. *Combinatorics, Probability and Computing*, 2(3):363–365, 1993. `doi:10.1017/S0963548300000730`.

27 Elchanan Mossel. Gaussian Bounds for Noise Correlation of Functions. *Geometric and Functional Analysis*, 19:1713–1756, 2010.

28 Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. *Annals of Mathematics*, 171:295–341, 2010.

29 N. Nisan. CREW PRAMs and decision trees. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 327–335. ACM, 1989. `doi:10.1145/73007.73038`.

30 Sushant Sachdeva and Rishi Saket. Optimal Inapproximability for Scheduling Problems via Structural Hardness for Hypergraph Vertex Cover. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 219–229, 2013. `doi:10.1109/CCC.2013.30`.

31 Rishi Saket. Hardness of Finding Independent Sets in 2-Colorable Hypergraphs and of Satisfiable CSPs. In *Proceedings of the 29th IEEE Conference on Computational Complexity*, pages 78–89, 2014.

32 Hans-Ulrich Simon. A tight $\Omega(\text{loglog n})$-bound on the time for parallel RAMs to compute nondegenerated boolean functions. In *Foundations of Computation Theory*, pages 439–444. Springer Berlin Heidelberg, 1983.

33 Cenny Wenner. Circumventing d-to-1 for Approximation Resistance of Satisfiable Predicates Strictly Containing Parity of Width at Least Four. *Theory of Computing*, 9(23):703–757, 2013.

34 Marcin Wrochna and Stanislav Zivny. Improved hardness for H-colourings of G-colourable graphs. *arXiv preprint*, 2019. `arXiv:1907.00872`.

35 Dmitriy Zhuk. A Proof of CSP Dichotomy Conjecture. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 331–342, 2017. `doi:10.1109/FOCS.2017.38`.

Syntactic Separation of Subset Satisfiability Problems

Eric Allender
Rutgers University, Piscataway, NJ 08854, USA
allender@cs.rutgers.edu

Martín Farach-Colton
Rutgers University, Piscataway, NJ 08854, USA
farach@cs.rutgers.edu

Meng-Tsung Tsai
National Chiao Tung University, Hsinchu, Taiwan
mtsai@cs.nctu.edu.tw

Abstract

Variants of the *Exponential Time Hypothesis* (ETH) have been used to derive lower bounds on the time complexity for certain problems, so that the hardness results match long-standing algorithmic results. In this paper, we consider a syntactically defined class of problems, and give conditions for when problems in this class require strongly exponential time to approximate to within a factor of $(1-\varepsilon)$ for some constant $\varepsilon > 0$, assuming the *Gap Exponential Time Hypothesis* (Gap-ETH), versus when they admit a PTAS. Our class includes a rich set of problems from additive combinatorics, computational geometry, and graph theory. Our hardness results also match the best known algorithmic results for these problems.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Syntactic Class, Exponential Time Hypothesis, APX, PTAS

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.16

Category APPROX

Funding *Eric Allender*: This research was supported in part by NSF grant CCF 1514164.
Martín Farach-Colton: This research was supported in part by NSF grants CCF 1637458, CNS 1408782, IIS 1541613, and NIH grant 1U01CA198952-01.
Meng-Tsung Tsai: This research was supported in part by the Ministry of Science and Technology of Taiwan under contract MOST grant 107-2218-E-009-026-MY3.

1 Introduction

Variants of the *Exponential Time Hypothesis* (ETH) [30, 31] have been used to derive lower bounds that match long-standing upper bounds for several important problems. In particular, the *Strong Exponential Time Hypothesis* (SETH) has been used to study the fine-grained complexity of problems in **P** [46, 47, 1, 14, 8], and the *Gap Exponential Time Hypothesis* (Gap-ETH) [21, 40] was used to study inapproximability [17, 22]. In this paper, we consider a syntactically-defined class of problems, defined below, and give conditions for when problems in this class require strongly exponential time to approximate to within a factor of $(1-\varepsilon)$ for some constant $\varepsilon > 0$, assuming Gap-ETH, versus when they admit a PTAS. Our hardness results also match the best known algorithmic results for these problems. Our class includes a rich set of problems from additive combinatorics, computational geometry, and graph theory.


© Eric Allender, Martín Farach-Colton, and Meng-Tsung Tsai;
licensed under Creative Commons License CC-BY



Let $L = \{\ell_1(\mathbf{x}), \ell_2(\mathbf{x}), \ldots, \ell_k(\mathbf{x})\}$ be a finite set of homogeneous linear functions in $\mathbb{Z}[\mathbf{x}]$ on the same set of variables $\mathbf{x} = (x_1, x_2, \ldots, x_r)$. We define a function $\ell(\mathbf{x})$ to be TRUE at $\mathbf{a}$ if $\ell(\mathbf{a}) \neq 0$. Otherwise, it is FALSE at $\mathbf{a}$. For any set S and integer r, let

$$\mathcal{D}(S, r) := \{(x_1, x_2, \ldots, x_r) \in S^r : x_i \neq x_j \text{ if } i \neq j \text{ for all } i, j \in [1, r]\},$$

that is, the set of permutations over all subsets of S of size r.

SUBSET-CSAT($\mathbf{L}$). Define $L^*(\mathbf{x}) = \bigwedge_{\ell \in L} \ell(\mathbf{x})$. Given a set S of n integers, find a largest $T \subseteq S$ so that for each r-tuple $\mathbf{a} = (a_1, a_2, \ldots, a_r) \in \mathcal{D}(T, r)$, L^* is TRUE at $\mathbf{a}$.[1]

SUBSET-DSAT($\mathbf{L}$). Define $L^+(\mathbf{x}) = \bigvee_{\ell \in L} \ell(\mathbf{x})$. Given a set S of n integers, find a largest $T \subseteq S$ so that for each r-tuple $\mathbf{a} = (a_1, a_2, \ldots, a_r) \in \mathcal{D}(T, r)$, L^+ is TRUE at $\mathbf{a}$.[1]

Many problems can be encoded as one of these two problem types [35, 51, 23, 54, 20, 29, 42, 24, 2, 25], some of which are known to be **APX**-hard, some of which are known to be **NP**-hard, and some of which have no known hardness result. The best known exact algorithms for each of them take strongly exponential time, i.e. $2^{\Omega(n)}$ time. Our main results are Theorem 2 and Theorem 3, below, which can be used to show that all these problems are ***strongly* APX-*hard***, where we define a problem X to be strongly **APX**-hard if there exists a ***size-preserving* PTAS** (SPTAS) reduction from MAX-3SAT to X. A SPTAS reduction is a PTAS reduction whose output has "size"[2] $O(n)$ for any input of size n.

Consequently, given Gap-ETH (Conjecture 1), X cannot be $(1-\delta)$-approximated in subexponential time for a sufficiently small constant $\delta > 0$. To simplify the reductions shown in the subsequent sections, we may restrict the instances of MAX-3SAT as was done in [22]. That is, we make use of the observation in footnote 5 of [40], so that we may assume that there is some constant Δ such that no variable of the formula appears in more than Δ clauses, and hence there are only $O(n)$ clauses, where n is the number of variables.

▶ **Conjecture 1** (Gap-ETH [21, 40]). *There exist constants $\varepsilon, c > 0$ so that no algorithm can distinguish a satisfiable 3SAT formula from those that cannot have more than $(1-\varepsilon)$-fraction of clauses being simultaneously satisfied in 2^{cn} time where n denotes the number of variables in the input instance.*

Our results are:

▶ **Theorem 2.** *Let L be a finite set of homogeneous linear functions whose coefficients are in $\mathbb{Z}$.*

- **(i)** *If L contains only functions with 1 or 2 variables, then* SUBSET-CSAT*(L) admits a* PTAS *and can be exactly solved in $2^{O(n^c)}$ time for some constant $c < 1$.*
- **(ii)** *Otherwise,* SUBSET-CSAT*(L) is strongly* **APX**-*hard.*

We observe here that it is *necessary* to limit our attention to hardness of approximation to within a *constant factor*. The problems we consider can easily be approximated to within a superconstant factor in $2^{o(n)}$ time. Thus strong APX-hardness differs from other hardness of approximation notions (which do not rely on strongly-exponential runtimes), for which it is interesting to consider larger approximation factors. We observe further that not all problems in case (i) are easy to compute exactly, nor are all problems in case (ii) hard to

[1] We assume that $|T| \geq r$ to avoid degenerate cases, which can be identified in $O(n^r)$ time.

[2] The size parameter is determined by problems: typically the number of variables in a formula or the number of nodes in a graph.

approximate to within a constant factor. An example problem for the former is finding a maximum independent set for *c-far unit-disk graphs*, an **NP**-hard problem [41, 56]. We defer the discussions to Appendix A. As for the latter, if all terms in L have positive sign, then a linear-time 1/2-approximation algorithm exists. Moreover, the constant c of case (i) depends on the coefficients of functions in L and the inapproximability constant of case (ii) depends on the number of variables of L.

We need some notions for the next result. We say an $r \times k$ matrix M is ***strongly full rank*** if $k \leq r$ and every $k \times k$ submatrix of M is full rank. Let $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}$ be vectors of the same dimensionality, and let $\mathbf{M} = (\mathbf{v_1}|\mathbf{v_2}|\ldots|\mathbf{v_k})$ be the matrix where $M_{ij} = \mathbf{v_j}[i]$. We call $\mathbf{M}$ the ***aggregation*** of $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}$. We say a vector space is in ***general position*** if it has a set of basis vectors whose aggregation is strongly full rank.

▶ **Theorem 3.** *Let L be a finite set of homogeneous linear functions whose coefficients are in $\mathbb{Z}$. For each* Subset-DSAT*(L), if the solutions to $\bigvee_{\ell \in L} \ell(\mathbf{x}) =$ False form a vector space in general position and has dimension at least 2 (hence $\mathbf{x}$ is a vector of at least 3 variables), then* Subset-DSAT*(L) is strongly* **APX***-hard.*

Applications. We show how to apply Theorem 2 and Theorem 3 to extend previous hardness results.

(1) Max-General: Given a set S of n points in $\mathbb{R}^2$, find a largest $T \subseteq S$ so that T contains no three distinct colinear points, i.e. finding a largest subset in general position. This problem is known to be **APX**-hard [25].

Here we show how to extend the **APX**-hardness result simply by encoding Max-General as a Subset-CSAT(L) problem for some L. Let $S = \{(a, a^3) : a \in Q\}$ for any set Q of integers. It is known [28] that Q has no three distinct integers that sum to 0 if and only if S has no three distinct colinear points. Therefore,

$$\text{Subset-CSAT}(L_{\text{GP}} := \{\ell(x, y, z) = x + y + z\})$$

can be reduced to Max-General by a linear-time reduction. Together with Theorem 2, one has that Max-General is strongly **APX**-hard.

Note that Subset-CSAT(L_{GP}) can be interpreted as the Max-3SUM problem, and Max-General is a typical example of a Max-3SUM-hard problem. More examples can be found in Section 3.

(2) Max-GolombRuler: Given a set S of n integers in $\mathbb{Z}$, find a largest $T \subseteq S$ so that T has $|T|^2$ distinct pairwise sums. This problem is known to be **NP**-hard to approximate to within an additive constant $c > 0$ [42].

We show how to improve the above inapproximability by encoding Max-GolombRuler as a Subset-CSAT(L) problem for some L. Observe that S has fewer than $|S|^2$ distinct pairwise sums if and only if either there exist four distinct numbers $a, b, c, d \in S$ so that $a + b = c + d$, or there exist three distinct numbers $a, b, c \in S$ so that $a + b = 2c$. To remove the fewest elements from S so that neither of the two cases hold is the same as solving

$$\text{Subset-CSAT}(L_{\text{GR}} := \{\ell_1(x, y, z, w) = x + y - z - w, \ell_2(x, y, z, w) = x + y - 2z\}).$$

Hence, by Theorem 2, Max-GolombRuler is strongly **APX**-hard.

(3) Max-C_3-Free: Given an undirected graph, find a largest node-induced subgraph (in terms of the number of nodes) that contains no cycle of length 3, i.e. a triangle. This problem is known to be **NP**-hard [35].

We show how to extend the **NP**-hardness result by encoding Max-C_3-Free as a Subset-CSAT(L) problem for some L with a restricted input $\bar{S}$. We restrict $\bar{S}$ to be a set such that for every six distinct integers $a_1, a_2, \ldots, a_6 \in \bar{S}$, there are at most two triples summing to 0. We construct an undirected graph $G = (V, E)$ as follows. Initially, $V \leftarrow \emptyset$, $E \leftarrow \emptyset$. For each $a \in \bar{S}$, add v_a to V. For each triple $a, b, c \in \bar{S}$ summing to 0, add edges $\{v_a, v_b\}, \{v_b, v_c\}, \{v_a, v_c\}$ to E. Given this construction, G has a C_3-free node-induced subgraph of k nodes if and only if

$$\text{Subset-CSAT}(L_{\text{C3}} := \{\ell(x, y, z) = x + y + z\}) \text{ with input } \bar{S}$$

has output of size k, which is strongly **APX**-hard as shown in Corollary 15. Hence, Max-C_3-Free is strongly **APX**-hard.

(4) Max-kAP-Free for each $k \geq 3$: Given two integers n and m, decide whether there exists a subset of $S = \{1, 2, \ldots, n\}$ of size at least m so that the subset contains no k distinct integers that form a k-term arithmetic progression. The tally representation of YES-instances of this problem defines a ***sparse language***, which cannot be **NP**-complete unless $\mathbf{P} = \mathbf{NP}$ [45]. An analogous situation also arises in other problems, such as in lattice problems in statistical physics (survey in [57]) or in determining Ramsey numbers (survey in [49]). More generally, if we assume ETH, no optimization problem that has $2^{o(n/\log n)}$ feasible instances can be strongly **APX**-hard. We refer readers to Section 7 for more discussion.

The current best algorithms for Max-kAP-Free [29, 24, 2] rely on branch-and-bound and have to invoke many Max-kAP-Free subproblems, that is, with an arbitrary $S \subseteq \{1, 2, \ldots, n\}$. A hardness result for the subproblem would suggest the limit of solving Max-kAP-Free by branch-and-bound algorithms. We show that it is strongly **APX**-hard.

We encode Max-kAP-Free as

$$\text{Subset-DSAT}(L_{k\text{AP}} := \{\ell_i(x_1, x_2, \ldots, x_k) = x_i - 2x_{i+1} + x_{i+2} : i \in [1, k-2]\})$$

where $|L_{k\text{AP}}| = k - 2$ and set $\mathbf{v_1} = (1, 3, \ldots, 2k-1)$, $\mathbf{v_2} = (2, 4, \ldots, 2k)$ as two basis vectors in the solution space of $\bigvee_{\ell \in L_{k\text{AP}}} \ell(\mathbf{x})$. Because $\mathbf{M} = (\mathbf{v_1}|\mathbf{v_2})$ is strongly full rank, by Theorem 3 we are done.

Our Techniques. We outline the techniques used in the proofs of Theorem 2 and Theorem 3. Both the algorithmic and the hardness results rely on Turán's Theorem [55, 53]. As originally stated, Turán's Theorem [55] said that for every n-node undirected simple graph G, if G has no clique of $r + 1$ nodes for an integer $r \geq 2$, then G has no more than $(1 - 1/r)n^2/2$ edges. In our proofs, when we refer to "Turán's Theorem", we refer to the second formulation of Turán's Theorem [53], that is:

▶ **Theorem 4** (Turán's Theorem [55, 53]). *Every n-node m-edge undirected simple graph has an independent set of size at least $\frac{n^2}{n+2m}$.*

We now describe our approach, and the role Turán's Theorem plays in obtaining our results.

(1) Our Algorithmic Results: Let L_{2^-} be any finite set that contains only homogeneous linear functions with 1 or 2 variables, with coefficients in $\mathbb{Z}$. In Theorem 2, we claim that Subset-CSAT(L_{2^-}) admits a PTAS and can be solved exactly in $2^{O(n^c)}$ time for some constant $c < 1$.

To obtain a PTAS or an exact algorithm for Subset-CSAT(L_{2^-}), we reduce it to finding a maximum independent set for the graph class $\mathcal{G}$ that contains all subgraphs of ***c-nearest neighborhood graphs***, defined in [26, 43], for some constant c. By a generalization of Lipton and Tarjan's algorithm [36], Max Independent Set for $\mathcal{G}$ can be solved efficiently. Lipton and Tarjan show how to approximate the Max Independent Set for planar graphs by exploiting the fact that every planar graph has a node separator of size $O(n^{1/2})$ whose removal partitions the graph into two balanced disconnected subgraphs. Their algorithm can be generalized to any graph class $\mathcal{H}$ that satisfies all the following properties:

- For every graph H in $\mathcal{H}$, any subgraph of H is a graph in $\mathcal{H}$.
- Every h-node graph H in $\mathcal{H}$ has a node separator of size $O(h^c)$ for some constant $c < 1$, whose removal partitions H into two balanced disconnected subgraphs, and the separator can be found in time polynomial in h.
- Every h-node H in $\mathcal{H}$ has an independent set of size $\Omega(h)$.

In Section 2, we will see that $\mathcal{G}$ satisfies all the above properties, and we generalize Lipton and Tarjan's algorithm for any graph class that fulfills all the required properties. We remark that Lipton and Tarjan [36] use the Four Color Theorem [6, 7] to prove the last property for planar graphs. However, since $\mathcal{G}$ contains non-planar graphs, we need to replace the Four Color Theorem with Turán's Theorem to show the last property for $\mathcal{G}$.

(2) Our Hardness Results: If a finite set L of homogeneous linear functions satisfies the condition for case (2) of Theorem 2 (resp. Theorem 3), then Subset-CSAT(L) (resp. Subset-DSAT(L)) is strongly **APX**-hard.

We show the hardness results by a reduction that maps from problem instances of Max Independent Set for sparse large-girth graphs to those of Subset-CSAT(L) or Subset-DSAT(L), so that if the former problem instance has an independent set of size k, then the latter problem instance has an output set of size $f(k)$ for some function f. The existence of the hardness reduction is secured by a probabilistic proof based on the Schwartz-Zippel Lemma [48, 58] as well as some tricks that prohibit the polynomials indicating the probability of desired events from vanishing, that is, that the desired events never happened.

Since all of our claims are applied to deterministic algorithms, we show how to derandomize the probabilistic construction by noting that the construction still works even when the random variables are constant-wise independent. We then use a standard technique to derandomize algorithms that use constant-wise independent random variables [37, 38]. Then, we prove that the reduction is approximation-preserving, again by Turán's Theorem.

We complete the proof by showing that Max Independent Set is strongly **APX**-hard even for sparse large-girth graphs.

Related Work. Our strong **APX**-hardness results apply to Max-rSUM and to some similar problems, which can be viewed as replacing the sum function with more general functions. A similar generalization from rSUM problems [32, 16] to a wider class of problems has also been found useful in studies of the time complexity of rSUM-hard problems in **P**, because the sum function may be not sufficient to encode an rSUM-hard problem but a more general function may [10].

We present a class of optimization problems that are strongly **APX**-hard because of a simple syntactic criterion. In that respect, there is some similarity to prior work on the MaxOnes problem. In [34], syntactic criteria were presented for certain MaxOnes problems, that imply **APX**-hardness. Related topics were also discussed in [9, 33]. Our results are not closely related to [9, 33, 34]; the full version of our paper will compare and contrast our results in more detail.

Paper Organization. In Section 2, we show the algorithmic results. Then, in Section 3, we exhibit our main techniques by proving the strong **APX**-hardness of a simple case Max-3SUM, implying strong **APX**-hardness for a list of Max-3SUM-hard problems via previously-known approximation-preserving reductions from 3SUM-hardness. In Section 4 and Section 5, we generalize the techniques used in Section 3 to prove Theorem 2. We prove Theorem 3 in Section 6, and relate strong **APX**-hardness to the density of languages in Section 7. Then, in Appendix A, we reduce the maximum independent set problem for some intersection graphs to the 2-variate case of Theorem 2 part (ii). In Appendix B, we prove the strong **APX**-hardness of some problems, which are used as source problems for the hardness reductions used in Sections 3 to 6. Finally, we give an inapproximability constant for each intractable problem in our syntactically-defined class in Appendix C.

2 Algorithmic Results

In this section, we prove the algorithmic results stated in Theorem 2, that is, for any finite set L_{2^-} that contains only functions with 1 or 2 variables, Subset-CSAT(L_{2^-}) admits a PTAS and can be exactly solved in $2^{O(n^c)}$ time for some constant $c < 1$. Some of these problems are known to be **NP**-hard; see Appendix A. If L_{2^-} contains a homogeneous linear function with 1 variable, then it suffices to remove 0 from S. Thus, in what follows, we consider Subset-CSAT(L_2) where L_2 is a finite set of homogeneous linear functions with precisely 2 variables. Note that every $\ell(\mathbf{x}) \in L_2$ still has r input variables, but only 2 of the r variables are used.

Given a problem Subset-CSAT(L_2), we construct an undirected simple graph $G_{L_2} = (V, E)$ where $V = \{v_a : a \in S\}$ and

$$E = \{(v_a, v_b) : \ell(\mathbf{x}) = 0 \text{ when } x_i = a, x_j = b \text{ for some } i \neq j \in [1, r], \ell(\mathbf{x}) \in L_2\}.$$

Because L_2 contains only linear functions with 2 variables, finding a maximum independent set for G_{L_2} is equivalent to solving Subset-CSAT(L_2). In what follows, we show that G_{L_2} is a subgraph of some ***c*-nearest neighborhood graph** (Lemma 6), defined below, and show that Max Independent Set for the graph class that consists of subgraphs of c-nearest neighborhood graphs admits a PTAS and can be solved exactly in subexponential time (Theorem 7). We assume that the underlying point set of c-nearest neighborhood graphs (or subgraphs of c-nearest neighborhood graphs) is given. This assumption holds for our case because the c-nearest neighborhood graphs used in our proofs are induced by a point set, and their subgraphs are induced by a subset of the same point set.

▶ **Definition 5** (c-nearest neighborhood graphs [26]). *Given a set P of points in $\mathbb{R}^d$, the c-nearest neighborhood graph of P is a graph $G_P = (V, E)$ whose $V = \{v_a : a \in P\}$ and*

$$E = \{(v_a, v_b) \in V^2 : a \text{ is the } i\text{-th nearest neighbor of } b \text{ for some } i \leq c\},$$

where ties are broken arbitrarily.

▶ **Lemma 6.** *G_{L_2} is a subgraph of some c_{L_2}-nearest neighborhood graph of an n-point set P_{L_2} in $\mathbb{Z}^{d_{L_2}}$ for some constants c_{L_2}, d_{L_2}.*

Proof. We construct the n-point set P_{L_2} by projecting each $v_a \in V(G_{L_2})$ into a point in $\mathbb{Z}^{t+2}$ for some constant $t \geq 0$ as follows. Define

$$\begin{aligned} D_{L_2} &= \{d : d \text{ is prime and } d \text{ divides } c, \text{ where } c \text{ is a coefficient of some } \ell(\mathbf{x}) \in L_2\} \\ &= \{d_1, d_2, \ldots, d_t\} \text{ where } t := |D_{L_2}|. \end{aligned}$$

Since L_2 is a finite set and each $\ell(\mathbf{x}) \in L_2$ has constant coefficients, t is a constant. Given D_{L_2}, for each $v_a \in V(G_{L_2})$ we write a as the unique factorization

$$a = (d_1)^{a_1} \cdots (d_t)^{a_t} (-1)^{a_{t+1}} a_{t+2} \text{ where } a_{t+1} \in \{0,1\}, a_{t+2} > 0 \text{ and } d \nmid a_{t+2} \text{ for all } d \in D_{L_2},$$

based on which we map v_a into the point $p_a := (a_1, a_2, \ldots, a_{t+2})$ for each $v_a \in V(G_{L_2})$.

Since each $\ell(\mathbf{x}) \in L_2$ has constant coefficients with prime divisors in D_{L_2}, if $\ell(\mathbf{x}) = 0$ when we set $x_i = a$ and $x_j = b$ for some $i \neq j \in [1, r]$, then $a_{t+2} = b_{t+2}$ and $a_i - b_i = O(1)$ for each $i \in [1, t+1]$. This yields that for every $(v_a, v_b) \in E(G_{L_2})$ the Euclidean distance between their associated points p_a, p_b is a constant, i.e. $||p_a - p_b||_2$ is a constant.

Let $C = \max_{(v_a,v_b)\in E(G), i\in[1,t+2]} |a_i - b_i|$. Then, for every edge $(v_a, v_b) \in E(G_{L_2})$, p_a is the i-th nearest neighbor of p_b for some $i \leq (2C+1)^{t+2}$, and vice versa. By setting

$$c_{L_2} = (2C+1)^{t+2} \text{ and } d_{L_2} = t+2,$$

we are done. ◀

▶ **Theorem 7.** Max Independent Set *for $\mathcal{H}$ admits a* PTAS *and can be solved exactly in subexponential time, where $\mathcal{H}$ is any graph class that satisfies all the following properties.*

(a) *For every graph H in $\mathcal{H}$, any subgraph of H is a graph in $\mathcal{H}$.*

(b) *Every h-node graph H in $\mathcal{H}$ has a node separator of size $O(h^c)$ for some constant $c < 1$, whose removal partitions H into two balanced disconnected subgraphs, and the separator can be found in time polynomial in h.*

(c) *Every h-node H in $\mathcal{H}$ has an independent set of size $\alpha(H) = \Omega(h)$.*

Proof. We show this by generalizing Lipton and Tarjan's algorithm for Max Independent Set on planar graphs, whose approximate version has the following pseudocode:

Input: an h-node undirected simple graph $H \in \mathcal{H}$

1 Find a node separator C of size $O(h/s^{\varepsilon})$ whose removal partitions H into disconnected subgraphs $H_1, H_2, \ldots, H_t$, each of which has fewer than s nodes, where $s \in (1, h)$ is a function of h and ε is some constant > 0;

2 Compute a maximum independent set I_i in H_i for each $i \in [1, t]$ by exhaustive search;

Output: $I_1 \cup I_2 \cup \cdots \cup I_t$

We need to argue that such a node separator C exists, given the properties of $\mathcal{H}$. We initialize a computation tree $\mathcal{T}$ as follows. Initially, $\mathcal{T}$ has only a root node, associated with H. Then, if there exists a leaf node $a \in \mathcal{T}$ associated with a graph H_a that has more than s nodes, we find a node separator C_a to partition H_a into two balanced disconnected subgraphs H_{a_1} and H_{a_2}. Such a C_a must exist by Properties (a) and (b). Then we link a with two child nodes, a_1 and a_2, whose associated graphs are H_{a_1} and H_{a_2}. Finally, each leaf node in $\mathcal{T}$ has fewer than s nodes. We let the subgraphs associated with leaf nodes in $\mathcal{T}$ be $H_1, H_2, \ldots, H_t$, and let the union of separators found during the construction of $\mathcal{T}$ be C.

By Property (b), C can be constructed in time polynomial in h. The following shows why the size of C is $O(h/s^{\varepsilon})$ for some constant $\varepsilon > 0$. We label each node $a \in \mathcal{T}$ with a height $t(a)$, i.e. the maximum length among all a-to-descendant-leaf paths. Let s_i for $i \geq 1$ be the lower bound on $|H_a|$ for all $a \in \mathcal{T}$ with height i. Since the found separator C_a partitions graph H_a into two balanced subgraphs, both of which have a constant fraction of the nodes in H_a, one can set $s_1 = s$ and $s_i = \Delta s_{i-1}$ for some constant $\Delta > 1$. The total number of nodes in the separators associated with of all nodes in $\mathcal{T}$ with height i is thus

$$\sum_{a \in \mathcal{T}, t(a)=i} |C_a| \leq \delta \left(\sum_{a \in \mathcal{T}, t(a)=i} |H_a|^{1-\varepsilon} \right) \leq \delta \frac{h}{s_i^{\varepsilon}}$$

where δ is a constant determined in Property (b) and the last inequality holds due to Hölder's inequality. Putting it all together, we get

$$|C| = \sum_{a \in \mathcal{T}} |C_a| \leq \delta \sum_{i=1}^{\infty} \frac{h}{(\Delta^{i-1} s)^{\varepsilon}} = O\left(\frac{h}{s^{\varepsilon}}\right).$$

To devise a polynomial-time approximation algorithm, we set $s = \log h$. Thus, the exhaustive search in Step 2 can be done in polynomial time. By the maximality of I_i, we have $\alpha(H) \leq \sum_{i \in t} |I_i| + O(h/\log^{\varepsilon} h)$. Together with $\alpha(H) = \Omega(h)$ due to Property (c), $\sum_{i \in t} |I_i| = (1 - o(1))\alpha(H)$, yielding a $(1 - o(1))$-approximation algorithm.

To devise a subexponential-time exact algorithm, we set $s = h^{\delta}$ for some constant $\delta \in (0, 1)$. Thus, the separator C has size $O(h^{1-\delta\varepsilon})$. Then we try all possible independent sets I_C of C, to be included in the output independent set, in $O(h^2 2^{h^{1-\delta\varepsilon}})$ time. For each I_C, we remove the neighbor nodes of I_C in $H_1, H_2, \ldots, H_t$. Then, we exhaustively search for a maximum independent set in the rest of H_i for each $i \in [1, t]$. These exhaustive searches can be done in $O(h^3 2^{h^{\delta}})$ time. As a result, $I_C \cup I_1 \cup \cdots \cup I_t$ is a maximum independent set for some I_C, and this exact algorithm takes

$$O(h^5 2^{h^{\delta} + h^{1-\delta\varepsilon}})$$

time, which is subexponential for any constant $\delta \in (0, 1)$. ◀

It remains to show that the graph class $\mathcal{G}$ that consists of subgraphs of c-nearest neighborhood subgraphs for some constant c satisfies all the properties listed in Theorem 7. It is clear that Property (a) holds for $\mathcal{G}$. It was shown in [26] that for any h-point set P, G_P has a node separator of size $O(c^{1/d} h^{1-1/d})$ whose removal partitions G_P into two balanced disconnected subgraphs. Moreover, such a node separator can be computed deterministically in $O(ch \log c + h \log h)$ time. For any resulting subgraph H of G_P, whose nodes are associated with a point set $P' \subseteq P$, one can construct the supergraph $G_{P'}$ of H and use the node separator of $G_{P'}$ as the node separator for H. Analogously, the size of the node separator and the running time to find it match the requirement. Thus, Property (b) holds for $\mathcal{G}$. Since any h-node subgraph of c-nearest neighborhood graphs have $O(h)$ edges for any constant c, by Turán's Theorem, Property (c) holds.

3 Hardness of Max-3SUM

In this section, we prove the hardness of Subset-CSAT($L_{3S} := \{\ell(x, y, z) = x + y + z\}$) and defer a proof for the general case in Theorem 2 to Section 4. The proof of the hardness of approximating Subset-CSAT(L_{3S}) will serve as intuition for the general case. The hardness of Subset-CSAT(L_{3S}) implies the hardness of the maximization version of numerous 3SUM-hard problems whose hardness reductions satisfy the following observation.

▶ **Observation 8.** There are many rSUM-hard decision problems $\mathcal{P}$ whose hardness reductions can be directly restated as SPTAS reductions from Max-rSUM to Max-$\mathcal{P}$.

Examples [28, 11, 13] include:

- Max-General: Given $S \subset \mathbb{R}^2$, find a largest $T \subseteq S$ so that T contains no three colinear points. This is one of the applications mentioned in Section 1.
- Max-$\delta\Delta$-Free: Given $S \subset \mathbb{R}^2$, find a largest $T \subseteq S$ so that T contains no three distinct points that form a triangle with area less than δ, for any fixed constant δ.
- Max-3AP-Free: Given $S \subset \mathbb{Z}$, find a largest $T \subseteq S$ so that T contains no three distinct integers that form an arithmetic progression. A more general case Max-kAP-Free for each $k > 3$ needs the hardness results shown in Section 6. We note here that a subset containing no 4-term arithmetic progressions may have 3-term arithmetic progressions, so the hardness of Max-3AP-Free does not immediately imply the hardness of Max-kAP-Free for each $k > 3$, whose proof relies on another system Subset-DSAT(L) for some L whose $|L| = k - 2$.
- Max-3L1P: Given S, a set of lines in $\mathbb{R}^2$, find a largest $T \subseteq S$ so that T contains no three distinct lines that intersect at a point.

NP-hardness. We claim the existence of a polynomial-time many-one reduction from instances of Max Independent Set to instances of Subset-CSAT(L_{3S}). Let n-node m-edge graph $G = (V, E)$ be an instance of Max Independent Set. We need a mapping f from $V \cup E$ to a set S of $n + m$ integers so that G has an independent set of size k iff Subset-CSAT(L_{3S}) with input S has output of size $k + m$. We show that such a set S exists by the probabilistic method [5] and show how to construct S deterministically in time polynomial in n, using derandomization [37, 38].

▶ **Lemma 9.** Subset-CSAT(L_{3S}) *is* **NP***-hard.*

Proof. To implement a mapping $f : V \cup E \to S$, we will use an ***n-order superposable set*** w.r.t. the function $\ell(x, y, z) = x + y + z \in L_{3S}$, which we define as follows. For any set B of n integers $X_1, X_2, \ldots, X_n$, we define the auxiliary set A_ℓ induced by B and ℓ to be

$$\{Y_{ij} : \ell(X_i, X_j, Y_{ij}) = 0, i, j \in [1, n], i < j\}.$$

We say B is an n-order superposable set if A_ℓ contains only integers, $|B \cup A_\ell| = n + \binom{n}{2}$, and for every three distinct integers $a_1, a_2, a_3 \in B \cup A_\ell$, $\ell(a_1, a_2, a_3) = 0$ only if $\{a_1, a_2, a_3\} = \{X_i, X_j, Y_{ij}\}$ for some $i, j \in [1, n], i < j$.

Given the superposable set B, one can realize a mapping $f : V \cup E \to S$, where $f(v_i) = X_i$ for each $v_i \in V$ and $f(\{v_i, v_j\}) = Y_{ij}$ for each $\{v_i, v_j\} \in E$. The following lemma will establish that the image set S and graph G preserve the relation required in the many-one reduction.

▶ **Lemma 10.** *An n-node m-edge graph $G = (V, E)$ has an independent set of size k iff* Subset-CSAT(L_{3S}) *with input $S = f(V \cup E)$ has output of size $k + m$.*

Proof.

(⇒) For each independent set I of G, $I \cup E$ corresponds to a set $T = \{f(a) : a \in I \cup E\}$, a subset of S. Since I is an independent set, for every edge $\{v_i, v_j\}$, the two integers $f(v_i)$, $f(v_j)$ are not simultaneously contained in T. By the definition of a superposable set, T is a valid output for Subset-CSAT(L_{3S}) with input S since it does not contain all three of $f(v_i), f(v_j), f(\{v_i, v_j\})$, for each pair of $i, j \in [1, n]$, $i < j$.

(⇐) Let T be a valid output for Subset-CSAT(L_{3S}) with input S. For each edge $\{v_i, v_j\} \in E$, if both $f(v_i), f(v_j) \in T$, then $f(\{v_i, v_j\}) \notin T$ because T is a valid output. In that case, one can modify T by replacing $f(v_i)$ with $f(\{v_i, v_j\})$. Such a modification does not change the size of T but reduces the number of pairs of $f(v_i), f(v_j)$ in T whose corresponding nodes v_i, v_j are adjacent in G. One can repeat the change until no such $f(v_i), f(v_j)$ pair exists in T. Hence, G has an independent set of size at least k. ◀

Let $R_p(n)$ be a set of n integers $X_1, X_2, \ldots, X_n$ sampled uniformly at random from the universe $U = \mathbb{Z}_p$, for some prime p. In Lemma 11, we prove that, for sufficiently large p, $R_p(n)$ is a superposable set with positive probability. We choose $\mathbb{Z}_p$ to facilitate the derandomization. However, if a set is superposable under $\mathbb{Z}_p$, then it is superposable under $\mathbb{Z}$. After the construction, we use this superposable set under $\mathbb{Z}$.

▶ **Lemma 11.** *The probability that $R_p(n)$ is an n-order superposable set is $1 - O(n^6/p)$.*

Proof. We note that for any pair of different linear polynomials, assigning an integer sampled uniformly at random from a universe U to each variable in the polynomials makes the two polynomials equal in $\mathbb{Z}_p$ with probability $p_{eq} = 1/|U|$, by a simple version of the Schwartz-Zippel Lemma [48, 58]. Here $U = \mathbb{Z}_p$ and $1/|U| = 1/p$. In subsequent sections, we will replace U with another set and will rely more heavily on the Schwartz-Zippel Lemma.

To show $B = R_p(n)$ is superposable, we consider the two probabilities:

$$\Pr\left[|B \cup A_\ell| < n + \binom{n}{2}\right] \leq \sum_{X_i, X_j \in B} p_{eq} + \sum_{X_i \in B, Y_{ij} \in A_\ell} p_{eq} + \sum_{Y_{ij}, Y_{i'j'} \in A_\ell} p_{eq} = O(n^4/p)$$

and

$$\Pr\left[\ell(a_1, a_2, a_3) = 0 \text{ for some } \{a_1, a_2, a_3\} \notin \Gamma\right] \leq \sum_{a_1, a_2, a_3 \in B \cup A_\ell} p_{eq} = O(n^6/p),$$

where $\Gamma := \{\{X_i, X_j, Y_{ij}\} : i, j \in [n], i < j\}$. We are done by applying the Union bound to the two failure probabilities. ◀

Observe that a fully random assignment to the variables of the polynomials is not necessary to make the two polynomials equal with probability as small as $1/p$. Instead, since L_{3S} contains only $\ell(x, y, z) = x + y + z$, if the variables $X_1, X_2, \ldots, X_n$ are assigned 6-wise-independently, the probability p_{eq} is still $1/p$. This observation yields a polynomial-time construction of the superposable set, as follows.

▶ **Lemma 12.** *One can construct an n-order superposable set in time polynomial in n.*

Proof. Exhaustively explore the polynomial-size probability space of 6-wise independence to find the superposable set, which is known to exist [37, 38]. ◀

We complete the proof of Lemma 9 by combining Lemmas 10, 11, and 12. ◀

Strong APX-hardness. In the **NP**-hardness reduction, we have presented a mapping $f : V \cup E \to S$, so that every n-node m-edge graph G has an independent set of size k iff Subset-CSAT(L_{3S}) with input S has output of size $k+m$. In order to demonstrate the strong **APX**-hardness of Subset-CSAT(L_{3S}), it suffices to restrict the Max Independent Set problem to sparse graphs. Thus we will give an SPTAS reduction from Max Independent Set for sparse graphs, which is strongly **APX**-hard (Lemma 24), to Subset-CSAT(L_{3S}).

▶ **Lemma 13.** *There is an* SPTAS *reduction from* Max Independent Set *for sparse graphs to* Subset-CSAT(L_{3S}).

Proof. We use the same reduction as in the proof of Lemma 9, which has the property that independent sets of size k correspond to a solution of Subset-CSAT(L_{3S}) with size $\geq k+m$. But by Turán's Theorem, we have that any sparse graph has an independent set of size $\Omega(m)$. Thus any solution that approximates Subset-CSAT(L_{3S}) to within a factor of $(1-\varepsilon)$ for some constant $\varepsilon > 0$ maps to a solution that approximates Max Independent Set for sparse graphs to within a factor of $(1-O(\varepsilon))$. It is easy to verify that the size of the output of the reduction is linear in the size of the input. ◀

By Lemma 13 and Lemma 24, we get:

▶ **Theorem 14.** Subset-CSAT(L_{3S}) *is strongly* **APX***-hard.*

The above SPTAS reduction is based on the hardness of Max Independent Set for sparse graphs (Lemma 24), which specifies additional structure on the input set S for Subset-CSAT(L_{3S}). Our reduction still works if the graph class is replaced with another graph class $\mathcal{G}$, as long as every n-node graph in $\mathcal{G}$ has $O(n)$ edges and has an independent set of size $\Omega(n)$, and Max Independent Set for $\mathcal{G}$ is strongly **APX**-hard. Such a replacement is useful for proving further hardness results. For example, by Lemma 25 and Turán's Theorem, the source problem of the reduction used in the proof of Theorem 14 can be replaced with Max Independent Set for triangle-free sparse graphs. This yields the following corollary.

▶ **Corollary 15.** Subset-CSAT(L_{3S}) *with input S, in which for every 6 distinct integers, there are at most two triples summing to 0, is strongly* **APX***-hard.*

4 Hardness of Subset-CSAT(L_{S_r})

We generalize the hardness result of Subset-CSAT(L_{3S}) in Section 3 to Subset-CSAT(L_{S_r}) where $L_{S_r} := \{\ell(\mathbf{x}) = \mathbf{c}\cdot\mathbf{x}\}$, and $\ell(\mathbf{x})$ is any linear function with coefficients $\mathbf{c} \in (\mathbb{Z}\setminus\{0\})^r$, for any $r \geq 3$.

Here we extend the definition of n-order superposable set for any r-variate homogeneous linear function $\ell(\mathbf{x})$. Let $t := r-3$. For any set

$$B = \{X_i : i \in [1,n]\} \cup \{X_{ijk} : i,j \in [1,n], i<j, k \in [1,t]\}$$

of $n + t\binom{n}{2}$ integers, we define the auxiliary set A_ℓ induced by B and ℓ to be

$$A_\ell = \{Y_{ij} : \ell(X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij}) = 0, i,j \in [1,n], i<j\}.$$

Let $\Gamma = \{\mathcal{S}_{ij} := \{X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij}\} : i,j \in [1,n], i<j\}$. We say B is an n-order superposable set if A_ℓ contains only integers, $|B \cup A_\ell| = n + (t+1)\binom{n}{2}$, and for every r distinct integers $a_1, a_2, \ldots, a_r \in B \cup A_\ell$, $\ell(a_1, a_2, \ldots, a_r) = 0$ only if $\{a_1, a_2, \ldots, a_r\} \in \Gamma$.

Let $G = (V, E)$ be a problem instance of Max Independent Set for sparse graphs. Given the superposable set B, we define a mapping $f : V \cup E \to 2^{B \cup A_\ell}$, where $f(v_i) = \{X_i\}$ for each $v_i \in V$ and $f(\{v_i, v_j\}) = \{X_{ij1}, \ldots, X_{ijr}, Y_{ij}\}$ for each $\{v_i, v_j\} \in E$. As in the proof of Lemma 9, if an n-order superposable set B can be constructed in time polynomial in n, then Subset-CSAT(L_{S_r}) is **NP**-hard. Moreover, the hardness-reduction is approximation-preserving for $\ell(\mathbf{x})$ simply by replacing $(1-O(\varepsilon))$ with $(1-O(r\varepsilon))$ in the proof of Lemma 13. Hence, Lemma 16 immediately follows by constructing B in polynomial-time.

▶ **Lemma 16.** Subset-CSAT(L_{S_r}) *is strongly* **APX***-hard.*

Proof. Recall that $\ell(\mathbf{x}) = \mathbf{c} \cdot \mathbf{x} = \sum_{i=1}^{r} c_i x_i$ where $c_i \in \mathbb{Z} \setminus \{0\}$ for each $i \in [1, r]$, and $t := r - 3$. Let $m = \ell cm(c_1, c_2, \ldots, c_r)$. We construct an n-order superposable set B by sampling X_i for each $i \in [1, n]$ and X_{ijk} for $i, j \in [1, n], i < j, k \in [1, t]$ from the universe $U = \mathbb{Z}_p \cap m\mathbb{Z}$ for some prime p. We choose the universe U in this way because no matter what the sampled values of X_i's and X_{ijk}'s are, they make all Y_{ij}'s integral. Before the sampling is performed, each X_i, X_{ijk} in B can be seen as an independent random variable and each Y_{ij} in A_ℓ can be seen as some linear combination of these independent random variables.

We show that the sampled B is an n-order superposable set with positive probability by bounding the probabilities of following bad events. Let E_1 indicate the event that $|B \cup A_\ell| < n + (t+1)\binom{n}{2}$. We claim that $\Pr[E_1] = n^c/(p/m)$ for some constant $c > 0$. To see this, we note that every two distinct random variables $a_1, a_2 \in B \cup A_\ell$ are different linear combinations of the random variables in $\{X_1, X_2, \ldots, X_n\}$. Since $X_1, X_2, \ldots, X_n$ are sampled independently from U, $\Pr[a_1 = a_2] = 1/(p/m)$. Together with the Union bound, the claimed bound for $\Pr[E_1]$ holds.

Let E_2 indicate the event that $\ell(a_1, a_2, \ldots, a_r) = 0$ for some $\{a_1, \ldots, a_r\} \notin \Gamma$. We claim that for every r distinct integers in $B \cup A_\ell$, $\ell(a_1, a_2, \ldots, a_r)$ cannot be a zero function if $(a_1, a_2, \ldots, a_r) \notin \Gamma$. We express a_k for each $k \in [1, r]$ as a linear combination of the random variables in B. To make $\ell(a_1, a_2, \ldots, a_r)$ a zero function, each variable in B either does not appear or appear more than once in a_k's expressions, for all $k \in [1, r]$. This observation implies that if an X_{ijk} in B for some $i, j \in [n], i < j, k \in [1, t]$ appears in the r expressions and $\ell(a_1, a_2, \ldots, a_r)$ is a zero function, then $X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij}$ also appear in the r expressions. Hence, in this case, $\{a_1, a_2, \ldots, a_r\} \in \Gamma$.

The remaining case is that X_{ijk} does not appear in any of the r expressions. In this case, to make $\ell(a_1, a_2, \ldots, a_r)$ a zero function, the only possible case happens when $r = 3$ and $\{a_1, a_2, a_3\} = \{Y_{ij}, Y_{jk}, Y_{ik}\}$ for some $i, j, k \in [n]$, $i < j < k$. However,

$$\ell(a_1, a_2, a_3) = c_1 \left(\frac{c_1 X_i + c_2 X_j}{-c_3} \right) + c_2 \left(\frac{c_1 X_j + c_2 X_k}{-c_3} \right) + c_3 \left(\frac{c_1 X_i + c_2 X_k}{-c_3} \right)$$

cannot be a zero function because X_j's coefficient is non-zero, or

$$\ell(a_1, a_2, a_3) = c_1 \left(\frac{c_1 X_i + c_2 X_j}{-c_3} \right) + c_2 \left(\frac{c_1 X_i + c_2 X_k}{-c_3} \right) + c_3 \left(\frac{c_1 X_j + c_2 X_k}{-c_3} \right)$$

cannot be a zero function unless $(c_1, c_2, c_3) = (\Delta, -\Delta, \Delta)$ for some $\Delta \neq 0$, which can be avoided by sorting the variables in ℓ by their coefficients. The same argument works when (a_1, a_2, a_3) equals other permutations of (Y_{ij}, Y_{jk}, Y_{ik}). Hence, the claim is true. There are a polynomial number of non-zero linear functions $\ell(a_1, a_2, \ldots, a_r)$ that cannot be zeroed by the random assigned values of X_i, X_{ijk} for $i, j \in [1, n], i < j, k \in [1, t]$. Therefore the failure rate is $n^c/(p/m)$ for some constant $c > 0$.

Given the bounds on failure probability, the randomly sampled B is an n-order superposable set with positive probability by picking p polynomially large in n. After a derandomization step similar to that in Lemma 12, we have B constructed in deterministic polynomial time. ◀

5 Hardness of Subset-CSAT(L_r)

We extend the hardness result of Subset-CSAT(L_{S_r}) to

$$\text{Subset-CSAT}(L_r := \{\ell_1(\mathbf{x}), \ell_2(\mathbf{x}), \ldots, \ell_k(\mathbf{x})\}),$$

where $\ell_i(\mathbf{x}) \in \mathbb{Z}[\mathbf{x}]$ for each $i \in [1,k]$, $\mathbf{x} = (x_1, x_2, \ldots, x_r)$, and at least one $\ell_i(\mathbf{x})$ uses at least 3 of the r input variables. Showing the strong **APX**-hardness of Subset-CSAT(L_r) proves Theorem 2.

We begin by defining a ***canonical representation*** for the $\ell_i(\mathbf{x})$'s. Observe that

$$\text{Subset-CSAT}(\{\ell_1(x,y,z,w) = x + y - z, \ell_2(x,y,z,w) = y + w - x\})$$

equals Subset-CSAT($\{\ell(x,y,z,w) = x+y-z\}$), which also equals Subset-CSAT($\{\ell(x,y,z) = \delta(x+y-z)\}$) for any constant $\delta \neq 0$, because in the definition of Subset-CSAT(L), we assume that the output has size at least r. Let Coef($\ell_i(\mathbf{x})$) be the multi-set of coefficients in $\ell_i(\mathbf{x})$. We say that $\ell_i(\mathbf{x})$ and $\ell_j(\mathbf{x})$ are in the same equivalence class if Coef($\ell_i(\mathbf{x})$) = $\{\delta c : c \in \text{Coef}(\ell_j(\mathbf{x}))\}$ for some non-zero constant δ. Thus, we can remove redundant functions in L, if any, by removing $\ell_i(\mathbf{x})$ from L if $\ell_i(\mathbf{x})$ and $\ell_j(\mathbf{x})$ are from the same equivalence class, for some $j < i$. Given the succinct representation of L, let $\ell_*(\mathbf{x})$ be the $\ell_i(\mathbf{x})$ in L that has the largest number of variables. If there is a tie, then pick any of them.

▶ **Lemma 17.** *Consider an r-variate homogeneous linear function $\ell_*(\mathbf{x})$ where $r \geq 3$, and an r'-variate homogeneous linear function $\ell(\mathbf{x})$ where $r' \leq r$. Let $t := r - 3$. Then for any constant $\varepsilon > 0$ there exists a randomized algorithm that constructs with probability at least $1-\varepsilon$ an n-order superposable set $B = \{X_i : i \in [1,n]\} \cup \{X_{ijk} : i,j \in [n], i < j, k \in [1,t]\}$ and the auxiliary set $A_{\ell_*} = \{Y_{ij} : \ell(X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij}) = 0, i,j \in [1,n], i < j\}$ induced by B and ℓ_*, so that for every r distinct integers in $B \cup A_{\ell_*}$, $\ell(a_1, a_2, \ldots, a_{r'}) = 0$ only if either of the following two cases applies:*

- *$r' = r$ and $\{a_1, a_2, \ldots, a_{r'}\} = \{X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij}\}$,*
- *$r' = r = 3$ and $\{a_1, a_2, a_3\} = \{Y_{s_1s_2}, Y_{s_2s_3}, Y_{s_1s_3}\}$ for some $1 \leq s_1 < s_2 < s_3 \leq n$.*

Proof. Set each element in B to be a random variable, and therefore each element in A_{ℓ_*} is a linear combination of $r-1$ random variables and none of them in the linear combination has coefficient 0. To make $\ell(\mathbf{x})$ a zero function by setting r' distinct variables from $(B \cup A_{\ell_*})$, it is necessary that each variable in B either does not appear among any of the r' picked variables or appears in at least two of them, noting that X_i is considered to "appear" in X_i and Y_{ij} for any $j \in [1,n]$. There are two cases. If X_{ijk} for some $i < j, i,j \in [n], k \in [1,t]$ is one of the r' picked variables, then $\ell(\mathbf{x})$ is zero only if the r' picked variables are exactly $X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij}$. Otherwise, for every $i < j$, $i,j \in [n]$, $k \in [1,t]$, X_{ijk} is *not* picked as one of the r' variables. In this case, to make $\ell(\mathbf{x})$ zero it is necessary that $t = 0$ (or equivalently $r = 3$), $r' = r$, and the r' picked variables are either X_i, X_j, Y_{ij} for some $i,j \in [n]$ or $Y_{s_1s_2}, Y_{s_2s_3}, Y_{s_3s_1}$ for some $1 \leq s_1 < s_2 < s_3 \leq n$.

Therefore, if we let $B = R_p(n)$, then the probability that $\ell(a_1, a_2, \ldots, a_{r'}) = 0$ for some $a_1, a_2, \ldots, a_{r'}$ other than the two given ones (the only cases that may make $\ell(\mathbf{x})$ as a zero function) is $1/p$. By the Union bound over all possible r' distinct values from $B \cup A_{\ell_*}$ in which there are $O(n^2)$ elements, we get the success probability of our random assignment is at least $1 - n^{2r'}/p$. Picking a sufficiently large p completes the proof. ◀

We apply Lemma 17 to each $\ell(\mathbf{x})$ in the succinct representation of L, take the Union bound over the failure probabilities, by the aforementioned derandomization step, we get:

▶ **Lemma 18.** *For any set $L_r(\mathbf{x})$ of r-variate homogeneous linear functions, if the function $\ell_*(\mathbf{x})$ in $L(\mathbf{x})$ that uses the largest number of variables is r'-variate for some $r' \geq 3$, let $t := r' - 3$, there exists a deterministic polynomial-time algorithm that can construct an n-order superposable set $B = \{X_i : i \in [1,n]\} \cup \{X_{ijk} : i,j \in [n], i < j, k \in [1,t]\}$ w.r.t. ℓ_* and the auxiliary set $A_{\ell_*} = \{Y_{ij} : \ell_*(X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij})\}$ induced by B and ℓ_* so that $\bigwedge_{\ell \in L} \ell(a_1, a_2, \ldots, a_r)$ evaluates to* False *only if either of the two following cases applies:*

- $\{a_1, a_2, \ldots, a_r\} = \{X_i, X_j, X_{ij1}, \ldots, X_{ijt}, Y_{ij}\}$,
- $r = 3$ *and* $\{a_1, a_2, a_3\} = \{X_{ij}, X_{jk}, X_{ik}\}$.

Proof of Theorem 2. Given Lemma 18, one can reuse the many-one reduction mentioned previously, but restrict the input graph to be triangle-free (i.e. girth ≥ 3), so that a_1, a_2, a_3 in the second case of Lemma 18 cannot simultaneously appear in the set S, i.e. the input of the reduction target. By Lemma 25, the maximum independent set problem for sparse graphs of girth ≥ 3 is strongly **APX**-hard, implying that Subset-CSAT(L_r) is strongly **APX**-hard. ◀

6 Hardness of Subset-DSAT($L_\vee$)

In this section, we will show the strong **APX**-hardness of

$$\textsc{Subset-DSAT}(L_\vee := \{\ell_1(\mathbf{x}), \ell_2(\mathbf{x}), \ldots, \ell_k(\mathbf{x})\}),$$

where $\ell_i(\mathbf{x}) \in \mathbb{Z}[\mathbf{x}]$ for each $i \in [1,k]$, $\mathbf{x} = (x_1, x_2, \ldots, x_r)$, and the solutions to $\bigvee_{\ell \in L} \ell(\mathbf{x}) =$ False form a vector space in general position and has dimension d at least 2. That is, we prove Theorem 3.

To prove Theorem 3 for $d = r - 1$, one can use the proof of Theorem 2. For $d < r - 1$ in general, the number of dependent random variables induced by the superposable set B is no longer 1, thus requiring the solution set of to be in general position. We need to modify the definition of the superposable set w.r.t. such a $\chi_{L_\vee}(\mathbf{x})$, as described below.

Proof of Theorem 3. For any n-node, m-edge graph $G = (V, E)$ that has $m = O(n)$ and girth at least $r + 1$, we construct a set B of independent random variables and an auxiliary set $A_{\chi_{L_\vee}}$ so that

$$B = \{X_i : i \in V\} \cup \{X_{ijk} : (i,j) \in E, i < j, k \in [1, d-2]\}, \text{ and}$$

$$A_{\chi_{L_\vee}} = \{Y_{ij1}, \ldots, Y_{ij(r-d)} : \chi_{L_\vee}\left(X_i, X_j, X_{ij1}, \ldots, X_{ij(d-2)}, Y_{ij1}, \ldots, Y_{ij(r-d)}\right) = 0, \\ (i,j) \in E, i < j\},$$

where the solution space of $\chi_{L_\vee}(\mathbf{x}) = 0$ is in general position and has dimension $d \geq 2$. Hence, for every $(i,j) \in E, i < j$, $\left(Y_{ij1}, Y_{ij2}, \ldots, Y_{ij(r-d)}\right)$ is unique.

Here we define the Y_{ijk} explicitly. Let $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_d}$ be a set of basis vectors (column vectors) in $\mathbb{Z}^r$ of the solution set of $\chi_{L_\vee}(\mathbf{x}) = 0$. Let $\mathbf{A}$ be the aggregation of $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_d}$ where $\mathbf{A} = (\mathbf{v_1}|\mathbf{v_2}|\cdots|\mathbf{v_d})$. Let $\mathbf{Q}$ be the square matrix composed of the upper d rows of $\mathbf{A}$. By the definition of general position, $\mathbf{A}$ is strongly full rank, $\mathbf{Q}$ is full rank, and thus $\mathbf{z}$ is uniquely defined by

$$\mathbf{Qz} = \begin{bmatrix} X_i \\ X_j \\ X_{ij1} \\ \vdots \\ X_{ij(d-2)} \end{bmatrix}, \text{ and we set } \mathbf{Az} = \mathbf{A}\left(\mathbf{Q}^{-1}\begin{bmatrix} X_i \\ X_j \\ X_{ij1} \\ \vdots \\ X_{ij(d-2)} \end{bmatrix}\right) = \begin{bmatrix} X_i \\ X_j \\ X_{ij1} \\ \vdots \\ X_{ij(d-2)} \\ Y_{ij1} \\ \vdots \\ Y_{ij(r-d)} \end{bmatrix}.$$

Thus, each of $Y_{ij1}, \ldots, Y_{ij(r-d)}$ is a nontrivial linear combination of $X_i, X_j, X_{ij1}, \ldots, X_{ij(d-2)}$. Note that $\mathbf{AQ}^{-1}$ is also strongly full rank, yielding that any nontrivial linear combination of d variables from the set $\{X_i, X_j, X_{ij1}, \ldots, X_{ij(d-2)}, Y_{ij1}, \ldots, Y_{ij(r-d)}\}$ cannot be a zero function. We are ready to prove that B is superposable w.r.t. E, that is:

▶ **Lemma 19.** *For any distinct $a_1, a_2, \ldots, a_r \in B \cup A_{\chi_{L_\vee}}$, $\chi_{L_\vee}(a_1, a_2, \ldots, a_r)$ is a zero function only if $\{a_1, a_2, \ldots, a_r\} = \{X_i, X_j, X_{ij1}, \ldots, X_{ij(d-2)}, Y_{ij1}, \ldots, Y_{ij(r-d)}\}$ for some $(i, j) \in E, i < j$.*

Proof. If $\{a_1, a_2, \ldots, a_r\} \subseteq \{X_i : i \in [1, n]\}$, then $\chi_{L_\vee}(a_1, a_2, \ldots, a_r)$ cannot be a zero function because the X_i's are independent variables and each X_i appears at most once in any linear function $\ell_j(\mathbf{x})$ that comprises $\chi_{L_\vee}$. Thus, to zero $\chi_{L_\vee}(a_1, a_2, \ldots, a_r)$ we may assume that

$$a_p \in \mathcal{S}_{ij} := \{X_{ij1}, \ldots, X_{ij(d-2)}, Y_{ij1}, \ldots, Y_{ij(r-d)}\} \text{ for some } p \in [1, r], (i, j) \in E, i < j.$$

Say a_p appears in some homogeneous linear function $\ell_q(\mathbf{x})$ that comprises $\chi_{L_\vee}$. In order to make $\chi_{L_\vee}(a_1, \ldots, a_r)$ a zero function, one must make $\ell_q(\mathbf{x})$ a zero function. We disprove the possibility of making $\ell_q(\mathbf{x})$ zeroed as follows. If $\ell_q(\mathbf{x})$ picks $\geq d$ variables from $\mathcal{S}_{ij}$, then each of $a_1, \ldots, a_r$ can be represented by a linear combination of random variables in $\mathcal{S}_{ij}$. In other words, $\{a_1, \ldots, a_r\} \subseteq (\mathcal{S}_{ij} \cup \{X_i, X_j\})$ because $d \geq 2$. If $\ell_q(\mathbf{x})$ picks $d-1$ variables from $\mathcal{S}_{ij}$, then to make $\ell_q(\mathbf{x})$ zeroed, $\ell_q(\mathbf{x})$ needs to pick two variables a_w and a_z where a_w is from $\mathcal{S}_{ik} \cup \{X_i\}$ and a_z is from $\mathcal{S}_{j\ell} \cup \{X_j\}$. Note that $k \neq \ell$ because G has girth $r + 1 \geq d + 2 \geq 4$. This would lead to a contradiction since if we solve the system by the $d-1$ variables from $\mathcal{S}_{ij}$ as well as a_w, then a_z can be represented by linear combination of variables from $\mathcal{S}_{ij} \cup \mathcal{S}_{ik} \cup \{X_i\}$, contradicting that $a_z \in \mathcal{S}_{j\ell}$, $\ell \neq k$, and $d \geq 2$. If $\ell_q(\mathbf{x})$ picks $\leq d-2$ variables from $\mathcal{S}_{ij}$, since the rest of variables can be partitioned into subsets, each of which sum to a multiple of X_i, or a multiple of X_j, but not a linear combination of X_i and X_j due to G having girth at least $r+1$, therefore $\ell_q(\mathbf{x})$ cannot be zeroed since this effectively picks $\leq d$ variables from $\mathcal{S}_{ij} \cup \{X_i, X_j\}$. ◀

Lastly, the exact construction of the superposable set is similar to that in Theorems 2. By Lemma 19 and the Swartz-Zippel Lemma, we know that sampling $\{X_i : i \in [1, n]\} \cup \{X_{ijk} : (i, j) \in E, i < j, k \in [1, d-2]\}$ uniformly at random from $(\det(\mathbf{Q})\mathbb{Z})^{n+(d-2)m}$ yields a superposable set with positive probability. We pick every X_i and X_{ijk} as multiples of $\mathbf{det}(\mathbf{Q})$ to ensure that all dependent variables Y_{ijk}'s are in $\mathbb{Z}$. Then, after derandomization using techniques for constant-wise independence, the construction takes time polynomial in n. Setting $g = r+1$, we know that Subset-DSAT$(L_\vee)$ is strongly **APX**-hard by Lemma 25. ◀

An implication of Theorem 3 is the strong **APX**-hardness of finding the maximum-cardinality k-term AP-free subset S for any fixed $k \geq 3$, noting that S may contain elements that form an i-term arithmetic progression for $i < k$ but not $i \geq k$. This problem can be encoded as

$$\textsc{Subset-DSAT}(L_{k\text{AP}} := \{\ell_i(x_1, x_2, \ldots, x_k) = x_i - 2x_{i+1} + x_{i+2} : i \in [1, k-2]\}),$$

and the solution set of $\sum_{\ell(\mathbf{x}) \in K_{k\text{AP}}} \ell^2(\mathbf{x}) = 0$ contains the plane

$$(x_1, x_2, \ldots, x_k) = \alpha(1, 3, \ldots, 2k-1) + \beta(2, 4, \ldots, 2k) \text{ for constant } \alpha, \beta \in \mathbb{R}.$$

Therefore,

$$M = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ \vdots & \vdots \\ 2k-1 & 2k \end{bmatrix} \text{ in which every } 2 \times 2 \text{ submatrix } \begin{bmatrix} 2i-1 & 2j-1 \\ 2i & 2j \end{bmatrix}$$

is full rank. By Theorem 3, we get:

▶ **Corollary 20.** *Finding a maximum-cardinality k-term AP-free subset of a given integral set S for any fixed $k \geq 3$ is strongly* **APX***-hard.*

Sharpness of $d \geq 2$. Not every problem in the class $\textsc{Subset-DSAT}(L)$ is hard to approximate. If the solution set of $\chi(\mathbf{x}) = \sum_{\ell(\mathbf{x}) \in L} \ell^2(\mathbf{x})$ is a point[3], then it suffices to remove an integer in the set S that coincides with the coordinate of the point. If it is a line, for example $\alpha(1, 2, 4, 8)$ for $\alpha \in \mathbb{R}$, then a greedy algorithm can solve this case in P by removing the last coordinate for every tuple of 4 integers that are multiples of (1, 2, 4, 8).

7 A Sparsity Bound, Assuming ETH

Define a ***sparse language*** as one where there are $n^{O(1)}$ length-n Yes-instances. Mahaney's theorem [39, 45] states that if $\mathbf{P} \neq \mathbf{NP}$, then there is no **NP**-hard sparse language. Buhrman and Hitchcock prove a stronger (optimal) bound from a stronger hypothesis [15]: if **PH** doesn't collapse, then there is no **NP**-hard set with $2^{n^{o(1)}}$ length-n Yes-instances. If one assumes ETH, then an even stronger bound holds for strongly **APX**-hard problems.

▶ **Theorem 21.** *If X is an optimization problem such that X has $2^{o(n/\log n)}$ strings of length n, then X cannot be strongly* **APX***-hard unless* ETH *fails.*

Proof. This proof is based on the proof of Mahaney's theorem presented in [45]. Assume that X is strongly **APX**-hard, and we will present a subexponential-time algorithm to solve 3SAT. Let

$$L = \{(\varphi, a) : \varphi \text{ has a satisfying assignment } a' \text{ lexicographically smaller than } a\}.$$

L is in nondeterministic linear time, and hence by [19] there is a reduction g of L to 3SAT such that, on input (φ, a) of size n, $g(\varphi, a)$ is a 3CNF formula with $O(n \log n)$ variables, each of which appears in $O(1)$ clauses.

[3] $\mathbf{0}$ must be a solution of $\chi(\mathbf{x})$ because the $\ell_i(\mathbf{x})$'s are homogeneous.

Since X is strongly **APX**-hard, there is a function f such that, for any 3CNF formula ψ of size n, $f(\psi)$ has size $O(n)$, where f yields the SPTAS reduction from Max-3SAT to X.

Consider any satisfiable formula φ with n variables; let a_φ be its lexicographically smallest satisfying assignment. Hence, $(\varphi, a) \in L$ if and only if $a \geq a_\varphi$, lexicographically.

We now present an algorithm for finding a_φ that runs in subexponential time. (If the algorithm fails to find a satisfying assignment, then φ is not satisfiable.) We start with a search space of size 2^n. Let $C = 2^{o(n)}$ be greater than the number of strings in X of length $m = O(n \log n)$, where the output of the reduction $g(\varphi, a)$ has length m. Find C assignments $a_1, \ldots, a_C$ that are evenly spaced among the current search space, and compute $z_i = f(g(\varphi, a_i))$ for $1 \leq i \leq C$.

If there are $i < j$ such that $z_i = z_j$, then $g(\varphi, a_i)$ is in 3SAT iff $g(\varphi, a_j)$ is, and thus a_φ does not lie in the segment $(a_i, a_j]$, and thus we can reduce the size of our search space by a factor of $1/C$.

Otherwise, there are C distinct elements z_i of the form $f(g(\varphi, a_i))$, which is greater than the number of relevant elements of X that can be in the range of f. Thus at least one of the formulae $g(\varphi, a_i)$ must be unsatisfiable, since f maps it to an infeasible instance of X. But if any formula $g(\varphi, a_i)$ is unsatisfiable, it follows that $g(\varphi, a_1)$ is unsatisfiable, and hence a_φ does not lie in the segment $[0^n, a_1]$, and thus again we can reduce the size of our search space by a factor of $1/C$.

We now repeat the process with a new set of C checkpoints. As in [45], the bookkeeping that is necessary to keep track of the current search space and to compute the new checkpoints does not get too complicated, and after a small number of iterations the entire search space is of size at most C, at which point we can check directly in subexponential time if any of the remaining assignments satisfies φ.

This algorithm can thus determine if φ is satisfiable or not, which is at least as hard as solving 3SAT. ◀

References

1. Amir Abboud and Virginia Vassilevska Williams. Popular Conjectures Imply Strong Lower Bounds for Dynamic Problems. In *Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, FOCS, pages 434–443, Washington, DC, USA, 2014. IEEE Computer Society. doi:10.1109/FOCS.2014.53.
2. Tanbir Ahmed, Janusz Dybizbanski, and Hunter S. Snevily. Unique Sequences Containing No k-Term Arithmetic Progressions. *Electr. J. Comb.*, 20(4):P29, 2013.
3. M. Ajtai, P. Erdös, J. Komlós, and E. Szemerédi. On Turán's theorem for sparse graphs. *Combinatorica*, 1(4):313–317, 1981.
4. M. Ajtai, J. Komlós, and E. Szemerédi. A Dense Infinite Sidon Sequence. *European Journal of Combinatorics*, 2(1):1–11, 1981.
5. Noga Alon and Joel Spencer. *The Probabilistic Method*. John Wiley, 1992.
6. K. Appel and W. Haken. Every planar map is four colorable. Part I: Discharging. *Illinois Journal of Mathematics*, 21(3):429–490, September 1977.
7. K. Appel, W. Haken, and J. Koch. Every planar map is four colorable. Part II: Reducibility. *Illinois Journal of Mathematics*, 21(3):491–567, September 1977.
8. Arturs Backurs and Piotr Indyk. Edit Distance Cannot Be Computed in Strongly Subquadratic Time (Unless SETH is False). *SIAM J. Comput.*, 47(3):1087–1097, 2018. doi:10.1137/15M1053128.
9. Brenda S. Baker. Approximation Algorithms for NP-complete Problems on Planar Graphs. *J. ACM*, 41(1):153–180, January 1994.

10 Luis Barba, Jean Cardinal, John Iacono, Stefan Langerman, Aurélien Ooms, and Noam Solomon. Subquadratic Algorithms for Algebraic 3SUM. *Discrete & Computational Geometry*, 61(4):698–734, 2019. doi:10.1007/s00454-018-0040-y.

11 Gill Barequet and Sariel Har-Peled. Polygon Containment and Translational Min-Hausdorff-Distance Between Segment Sets are 3Sum-hard. *Int. J. Comput. Geometry Appl.*, 11(4):465–474, 2001.

12 Piotr Berman and Marek Karpinski. On Some Tighter Inapproximability Results (Extended Abstract). In *26th International Colloquium in Automata, Languages and Programming (ICALP)*, pages 200–209, 1999.

13 Prosenjit Bose and Stefan Langerman. Weighted Ham-Sandwich Cuts. In *Discrete and Computational Geometry, Japanese Conference, JCDCG, Revised Selected Papers*, pages 48–53, 2004.

14 Karl Bringmann. Why Walking the Dog Takes Time: Frechet Distance Has No Strongly Subquadratic Algorithms Unless SETH Fails. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 661–670, 2014.

15 Harry Buhrman and John M. Hitchcock. NP-hard sets are exponentially dense unless coNP $\subseteq$ NP/poly. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity, (CCC)*, pages 1–7. IEEE Computer Society, 2008. doi:10.1109/CCC.2008.21.

16 Jean Cardinal, John Iacono, and Aurélien Ooms. Solving k-SUM Using Few Linear Queries. In *24th Annual European Symposium on Algorithms, ESA*, pages 25:1–25:17, 2016.

17 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-ETH to FPT-Inapproximability: Clique, Dominating Set, and More. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 743–754, 2017.

18 Miroslav Chlebík and Janka Chlebíková. Approximation Hardness for Small Occurrence Instances of NP-hard Problems. In *5th Italian Conference on Algorithms and Complexity (CIAC)*, pages 152–164. Springer, 2003.

19 Stephen A. Cook. Short Propositional Formulas Represent Nondeterministic Computations. *Inf. Process. Lett.*, 26(5):269–270, 1988. doi:10.1016/0020-0190(88)90152-4.

20 Carlos Cotta, Iván Dotú, Antonio J. Fernández, and Pascal Van Hentenryck. A Memetic Approach to Golomb Rulers. In *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature*, PPSN, pages 252–261, Berlin, Heidelberg, 2006. Springer-Verlag.

21 Irit Dinur. Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:128, 2016.

22 Irit Dinur and Pasin Manurangsi. ETH-hardness of approximating 2-CSPs and directed Steiner network. In *9th Innovations in Theoretical Computer Science Conference, ITCS*, pages 36:1–36:20, 2018.

23 Apostolos Dollas, William T. Rankin, and David Mccracken. New Algorithms for Golomb Ruler Derivation and Proof of the 19 Mark Ruler. *IEEE Transactions on Information Theory*, 44:379–382, 1998.

24 J. Dybizbański. Sequences containing no 3-term arithmetic progressions. *Elec. J. of Comb.*, 19(2):15–19, 2012.

25 David Eppstein. *Forbidden Configurations in Discrete Geometry*. Cambridge University Press, 2018.

26 David Eppstein, Gary L. Miller, and Shang-Hua Teng. A Deterministic Linear Time Algorithm for Geometric Separators and its Applications. *Fundam. Inform.*, 22(4):309–329, 1995. doi:10.3233/FI-1995-2241.

27 S. Fajtlowicz. The independence ratio for cubic graphs. In *8th Southeastern Conf. on Combinatorics, Graph Theory, and Computing*, pages 273–277. LSU, 1977.

28 Anka Gajentaan and Mark H. Overmars. On a Class of $O(N^2)$ Problems in Computational Geometry. *Comput. Geom. Theory Appl.*, 5(3):165–185, October 1995.

29 William I. Gasarch, James Glenn, and Clyde P. Kruskal. Finding large 3-free sets I: The small n case. *J. Comput. Syst. Sci.*, 74(4):628–655, 2008.

30 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

31 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which Problems Have Strongly Exponential Complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, December 2001. doi:10.1006/jcss.2001.1774.

32 Allan Grønlund Jørgensen and Seth Pettie. Threesomes, Degenerates, and Love Triangles. *J. ACM*, 65(4):22:1–22:25, 2018. doi:10.1145/3185378.

33 Sanjeev Khanna and Rajeev Motwani. Towards a Syntactic Characterization of PTAS. In *28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 329–337. ACM, 1996.

34 Sanjeev Khanna, Madhu Sudan, Luca Trevisan, and David P. Williamson. The Approximability of Constraint Satisfaction Problems. *SIAM J. Comput.*, 30(6):1863–1920, December 2001.

35 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980.

36 Richard J. Lipton and Robert Endre Tarjan. Applications of a Planar Separator Theorem. *SIAM J. Comput.*, 9(3):615–627, 1980. doi:10.1137/0209046.

37 Michael Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM J. Comput.*, 15(4):1036–1053, 1986.

38 Michael Luby and Avi Wigderson. Pairwise Independence and Derandomization. *Found. Trends Theor. Comput. Sci.*, 1(4):237–301, August 2006.

39 Stephen R. Mahaney. Sparse Complete Sets of NP: Solution of a Conjecture of Berman and Hartmanis. *J. Comput. Syst. Sci.*, 25(2):130–143, 1982. doi:10.1016/0022-0000(82)90002-2.

40 Pasin Manurangsi and Prasad Raghavendra. A Birthday Repetition Theorem and Complexity of Approximating Dense CSPs. In *44th International Colloquium on Automata, Languages, and Programming, ICALP*, pages 78:1–78:15, 2017. doi:10.4230/LIPIcs.ICALP.2017.78.

41 Nimrod Megiddo and Kenneth J. Supowit. On the Complexity of Some Common Geometric Location Problems. *SIAM J. Comput.*, 13(1):182–196, 1984.

42 Christophe Meyer and Periklis A. Papakonstantinou. On the Complexity of Constructing Golomb Rulers. *Discrete Appl. Math.*, 157(4):738–748, February 2009.

43 Gary L. Miller, Shang-Hua Teng, William Thurston, and Stephen A. Vavasis. Separators for Sphere-packings and Nearest Neighbor Graphs. *J. ACM*, 44(1):1–29, January 1997. doi:10.1145/256292.256294.

44 Owen J. Murphy. Computing Independent Sets in Graphs with Large Girth. *Discrete Appl. Math.*, 35(2):167–170, January 1992.

45 Mitsunori Ogiwara and Osamu Watanabe. On Polynomial-Time Bounded Truth-Table Reducibility of NP Sets to Sparse Sets. *SIAM J. Comput.*, 20(3):471–483, 1991.

46 Mihai Pătraşcu and Ryan Williams. On the Possibility of Faster SAT Algorithms. In *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 1065–1075, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

47 Liam Roditty and Virginia Vassilevska Williams. Fast Approximation Algorithms for the Diameter and Radius of Sparse Graphs. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC, pages 515–524, New York, NY, USA, 2013. ACM. doi:10.1145/2488608.2488673.

48 J. T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *J. ACM*, 27(4):701–717, October 1980.

49 Pascal Schweitzer. *Problems of unknown complexity – Graph isomorphism and Ramsey theoretic numbers*. PhD thesis, Universität des Saarlandes, 2009.

50 James B. Shearer. A note on the independence number of triangle-free graphs. *Discrete Mathematics*, 46(1):83–87, 1983.

51 Stephen W. Soliday, Abdollah Homaifar, and Gary L. Lebby. Genetic Algorithm Approach to the Search for Golomb Rulers. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 528–535, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. URL: `http://dl.acm.org/citation.cfm?id=645514.658082`.

52 William Staton. Some Ramsey-Type Numbers and the Independence Ratio. *Transactions of the American Mathematical Society*, 256:353–370, 1979.

53 T. Tao and V. H. Vu. *Additive Combinatorics*. Cambridge University Press, 2009.

54 Jorge Tavares, Francisco B. Pereira, and Ernesto Costa. Understanding the role of insertion and correction in the evolution of Golomb rulers. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC*, pages 69–76, 2004. `doi:10.1109/CEC.2004.1330839`.

55 Pál Turán. Egy gráfelméleti szélsőérték feladatról. *Matematikai és Fizikai Lapok*, 48:436–452, 1941.

56 D. W. Wang and Yue-Sun Kuo. A Study on Two Geometric Location Problems. *Inf. Process. Lett.*, 28(6):281–286, 1988.

57 D. J. A. Welsh. The Computational Complexity of Some Classical Problems from Statistical Physics. In *In Disorder in Physical Systems*, pages 307–321. Clarendon Press, 1990.

58 Richard Zippel. Probabilistic Algorithms for Sparse Polynomials. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 216–226. Springer, 1979.

A Reducing Some MIS Problems to Subset-CSAT(L)

We reduce the problem of finding a maximum independent set for *c-far unit-disk graphs* to Subset-CSAT(L) for some 2-variable L. A unit disk graph G is an intersection graph of unit disks in the plane. We say a unit-disk graph is *c-far* if for each pair of disks the Euclidean distance between their centers does not fall within the interval $[0, c) \cup (2, 2 + c)$ for some constant $c > 0$. It is known that the maximum independent set problem remains **NP**-hard for c-far unit-disk graphs [41, 56], even when the locations of disks are given.

▶ **Theorem 22.** *There exists a polynomial-time many-one reduction from finding a maximum independent set for c-far unit-disk graphs to Subset-CSAT(L) for some 2-variable L.*

Proof. The reduction comes as follows. Let $D = \{d_1, d_2, \ldots, d_n\}$ be the set of the n disks and let $x(d_i)$ and $y(d_i)$ denote the x- and y-coordinate of disk d_i for each $i \in [1, n]$. We discretize the locations of disks in D so that $x(d_i)$ and $y(d_i)$ for all $i \in [1, n]$ are mapped to multiples of ε where ε is set as $c/6$. Observe that, if two disks intersect before the discretization, then their distance is in the range $[4c/6, 2 + 2c/6]$; if two disks do *not* intersect before the discretization, then their distance now falls within $[2 + 4c/6, \infty)$. If we enlarge the radius of all disks from 1 to $1 + 3c/12$, then the discretization does not alter whether two disks intersect or not. In other words, if two disks intersect, then the center of one disk is located at one of the $O(1/\varepsilon^2)$ discretized coordinates surrounding the center of the other.

Consequently, if we map each disk d_i to an integer $2^{x(d_i)/\varepsilon}3^{y(d_i)/\varepsilon}$, noting that the exponents are integers for each $i \in [1, n]$, and set

$$L_{\text{udisk}} = \{\ell(a, b) = 2^{r_1}3^{r_2}a - 2^{r_3}3^{r_4}b : \varepsilon\sqrt{(r_1 - r_3)^2 + (r_2 - r_4)^2} < 2 + c/2, r_1, r_2, r_3, r_4 \in \mathbb{N}\},$$

then it is clear that Subset-CSAT(L_{udisk}) is a restatement of finding a maximum independent set for c-far unit-disk graphs. ◀

Combining Theorem 22 and Theorem 2, we get:

▶ **Corollary 23.** *Finding a maximum independent set for c-far unit-disk graphs admits a* PTAS.

We remark here that one can have a result analogous to Corollary 23 for c-far intersection graphs whose underlying shape is a unit symmetric convex set. This result is not as general as for ordinary intersection graphs because c-farness implies that all nodes in the intersection graph have a constant degree.

B Initial Hardness Results

Our hardness proofs are based on the strong **APX**-hardness of Max Independent Set for sparse large-girth graphs, which can be shown by the following chain of reductions.

Let Max-3SAT-Δ be the subproblem of Max-3SAT so that there exists a constant Δ such that no variables of the formula appears in more than Δ clauses. Let Max-IS be the maximum independent set problem. In what follows, we will show that Max-3SAT-Δ $\leq_{\text{SPTAS}}$ Max-IS for sparse graphs $\leq_{\text{SPTAS}}$ Max-IS for sparse large-girth graphs.

▶ **Lemma 24.** Max Independent Set *for sparse graphs, i.e. with a linear number of edges, is strongly* **APX***-hard.*

Proof. Let $I_{3\text{SAT}}$ be an instance of Max-3SAT-Δ. We assume that $I_{3\text{SAT}}$ has n variables and m clauses, and each clause in $I_{3\text{SAT}}$ has exactly 3 literals. Otherwise, one can duplicate some literal in each of the 1-literal and 2-literal clauses. Given $I_{3\text{SAT}}$, we construct a graph $G = (V, E)$ as I_{MIS} as follows. For each $i \in [1, m]$, we add three nodes $v_{i_1}, v_{i_2}, v_{i_3}$ to V, link each pair of the three nodes with an edge, and label $v_{i_1}, v_{i_2}, v_{i_3}$ with the corresponding literal in the i-th clause. Then, for every pair of nodes in V, if their labels are literals which are negations of each other, then link an edge between them. Consequently, G has $3m$ nodes and at most $3m + 9\binom{\Delta}{2}n = O(m)$ edges. It can be checked that $I_{3\text{SAT}}$ can have t clauses simultaneously satisfied if and only if I_{MIS} has an independent set of size t. Moreover, the problem instances have size linear to each other. This gives a SPTAS reduction. ◀

▶ **Lemma 25.** *For every constant $c \geq 3$,* Max Independent Set *for sparse graphs of girth $\geq c$ is strongly* **APX***-hard.*

Proof. Let I_s (resp. $I_{s,g\geq c}$) be an instance of Max Independent Set for sparse graphs (resp. Max Independent Set for sparse graphs of girth $\geq c$). One can map I_s to $I_{s,g\geq c}$ by replacing each edge (v_a, v_b) with a path from v_a to v_b with $2c$ internal nodes, as shown in [44]. Hence, $I_{s,g\geq c}$ has girth $\geq 6c+3$, and I_s has an independent set t if and only if $I_{s,g\geq c}$ has an independent set of size $t + cm$. Every $(1-\varepsilon)$-approximation for $I_{s,g\geq c}$ determines that $I_{s,g\geq c}$ has an independent set of size $(1-\varepsilon)(t+cm)$, which corresponds to I_s having an independent set of size $(1-\varepsilon)t - \varepsilon cm = (1 - O(\varepsilon))t$, where the equality holds because c is a constant and $t = \Omega(n) = \Omega(m)$ by Turán's Theorem. Moreover, the problem instances have size linear to each other. This gives a SPTAS reduction. ◀

C Inapproximability Constants

Lastly, for each problem in the syntactically-defined class that does not admit a PTAS, we determine an inapproximability constant $1-\varepsilon$, so that it cannot be $(1-\varepsilon)$-approximated unless $\mathbf{P} = \mathbf{NP}$. We use the facts that Max Independent Set on 3-regular graphs cannot be approximated to within the constant $C_3 = 139/140 + \varepsilon$ for any constant $\varepsilon > 0$ [12], and Max Independent Set on 3-regular triangle-free graphs can not be approximated to within the constant $C_{3\Delta} = 1422/1432 + \varepsilon$ for any constant $\varepsilon > 0$ [18].

We first apply Lemma 13 to bound an inapproximability constant $1-\delta_r$ based on C_3 and then replace the use of Turán's Theorem in Lemma 13 with the AKS Theorem [4] and Staton's result [52] to bound the claimed inapproximability constant $1-\varepsilon_r$ based on $C_{3\Delta}$.

Since Max Independent Set on 3-regular graphs cannot be approximated to within C_3, from Lemma 13 we have following theorem:

▶ **Lemma 26.** *For every homogeneous, r-variate ($r \geq 3$), linear function $\ell(\mathbf{x})$,* Subset-CSAT$(\{\ell(\mathbf{x})\})$ *cannot be approximated to within any constant factor larger than $1-\delta_r$ in polynomial time unless $\mathbf{P} = \mathbf{NP}$, where $\delta_r = \frac{1-C_3}{7+6(r-3)}$.*

Simply replacing C_3 with $C_{3\Delta}$ cannot increase δ_r because $C_3 < C_{3\Delta}$ and such a replacement in Lemma 26 makes δ_r smaller. Instead, we replace the use of Turán's Theorem, which applies to general graphs, with the AKS Theorem (see Theorem 27), which works for triangle-free graphs. In [3, 50], the constant in the big-Omega notation in AKS Theorem is bounded above by 1/100 and 1/8, respectively. Though the size of an independent set guaranteed by the AKS theorem is asymptotically larger than that of Turán theorem, it is numerically smaller when $d = 3$.

▶ **Theorem 27** (AKS Theorem [4])**.** *Every d-regular triangle-free graph has an independent set of size $\Omega(n \log d/d)$.*

Note that the constant in the big-Omega notation is universal for every d. For a particular value of d the constant can be larger. In particular, in [52] Staton shows that every 3-regular triangle-free graph has an independent set of size $5m/21$, which is more than the $m/6$ guaranteed by Turán's theorem. The constant 5/21 is tight due to Fajtlowicz [27]. Based on this improved guarantee of the size of an independent set, we obtain the following result.

▶ **Lemma 28.** *For each homogeneous, r-variate, linear function $\ell(\mathbf{x})$,* Subset-CSAT$(\{\ell(\mathbf{x})\})$ *cannot be approximated to within any constant factor larger than $1-\varepsilon_r$ in polynomial time unless $\mathbf{P} = \mathbf{NP}$, where $\varepsilon_r = 1 - \frac{1-C_{3\Delta}}{5.2+4.2(r-3)}$.*

Lemma 28 and the proof of Theorem 2 together imply that:

▶ **Theorem 29.** *Let L be a finite set of homogeneous linear functions whose coefficients are in $\mathbb{Z}$. If L contains a homogeneous r-variate linear function $\ell(\mathbf{x})$ for some $r \geq 3$, then* Subset-CSAT(L) *cannot be approximated to within any constant factor larger than $1-\varepsilon_r$ in polynomial time unless $\mathbf{P} = \mathbf{NP}$, where $\varepsilon_r = 1 - \frac{1-C_{3\Delta}}{5.2+4.2(r-3)}$.*

To obtain the inapproximability constants for Subset-DSAT(L), we need Lemma 30.

▶ **Lemma 30.** Max Independent Set *for graphs whose maximum degree ≤ 3 and girth $\geq g$ cannot be approximated to within any constant factor larger than $1-\varepsilon_g$ in polynomial time unless $\mathbf{P} = \mathbf{NP}$, where $\varepsilon_g < \frac{1}{140(6\lceil (g-3)/6\rceil+1)}$.*

Proof. We prove this by giving a PTAS reduction from Max Independent Set for 3-regular graphs $G_{3r} = (V_{3r}, E_{3r})$ to Max Independent Set for graphs $G_{g^+} = (V_{g^+}, E_{g^+})$ of girth $\geq g$. We obtain G_{g^+} from G_{3r} by replacing each edge in E_{3r} with a path of length $2t+1$ ($t \in \mathbb{Z}$), connecting $2t$ new nodes. Hence, the smallest cycle in G_{g^+} is $3+6t$. We pick $t = \lceil (g-3)/6 \rceil$ so that G_{g^+} has no cycle of length $< g$.

It is known [44] that G_{g^+} has an independent set of size $t|E_{3r}| + k$ iff G_{3r} has an independent set of size k. Every $(1-\varepsilon)$-approximation algorithm for Max Independent Set of G_{g^+} can find an independent set of size $(1-\varepsilon)(t|E_{3r}| + k)$, which corresponds to an

independent set of size $(1-\varepsilon)k - \varepsilon t|E_{3r}| \geq (1-(6t+1)\varepsilon)k$ in G_{3r}, where the last inequality follows from the fact that $k \geq |E_{3r}|/6$ for every 3-regular graph, due to Turán's Theorem [53].

Based on [12], Max Independent Set for 3-regular graphs cannot be approximated to within $1-\varepsilon_{3r}$ for any $\varepsilon_{3r} < 1/140$. Thus, ε cannot be less than $\frac{1}{140(6t+1)} = \frac{1}{140(6\lceil(g-3)/6\rceil+1)}$. ◀

In the proof of Theorem 3, the girth g is set as $r+1$, where r denotes $|\mathbf{x}|$. Hence, we get:

▶ **Theorem 31.** *Let L be a finite set of homogeneous linear functions whose coefficients are in $\mathbb{Z}$. For each* Subset-DSAT*(L), if the solutions to $\bigvee_{\ell\in L}\ell(\mathbf{x}) =$ False form a vector space in general position and has dimension at least 2, then* Subset-DSAT*(L) cannot be approximated to within any constant factor larger than $1-\varepsilon_r$ in polynomial time unless $\mathbf{P} = \mathbf{NP}$, where $\varepsilon_r = \frac{1}{140(6\lceil(r-2)/6\rceil+1)}$.*

Malleable Scheduling Beyond Identical Machines

Dimitris Fotakis
School of Electrical and Computer Engineering, National Technical University of Athens, Greece
https://www.softlab.ntua.gr/~fotakis/
fotakis@cs.ntua.gr

Jannik Matuschke
Research Center for Operations Management, KU Leuven, Belgium
https://sites.google.com/view/jannikmatuschke/
jannik.matuschke@kuleuven.be

Orestis Papadigenopoulos
Department of Computer Science, The University of Texas at Austin, USA
http://www.cs.utexas.edu/~papadig/
papadig@cs.utexas.edu

Abstract

In malleable job scheduling, jobs can be executed simultaneously on multiple machines with the processing time depending on the number of allocated machines. Jobs are required to be executed non-preemptively and in unison, in the sense that they occupy, during their execution, the same time interval over all the machines of the allocated set. In this work, we study generalizations of malleable job scheduling inspired by standard scheduling on unrelated machines. Specifically, we introduce a general model of malleable job scheduling, where each machine has a (possibly different) speed for each job, and the processing time of a job j on a set of allocated machines S depends on the total speed of S for j. For machines with unrelated speeds, we show that the optimal makespan cannot be approximated within a factor less than $\frac{e}{e-1}$, unless $P = NP$. On the positive side, we present polynomial-time algorithms with approximation ratios $\frac{2e}{e-1}$ for machines with unrelated speeds, 3 for machines with uniform speeds, and 7/3 for restricted assignments on identical machines. Our algorithms are based on deterministic LP rounding and result in sparse schedules, in the sense that each machine shares at most one job with other machines. We also prove lower bounds on the integrality gap of $1 + \varphi$ for unrelated speeds (φ is the golden ratio) and 2 for uniform speeds and restricted assignments. To indicate the generality of our approach, we show that it also yields constant factor approximation algorithms (i) for minimizing the sum of weighted completion times; and (ii) a variant where we determine the effective speed of a set of allocated machines based on the L_p norm of their speeds.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Scheduling algorithms

Keywords and phrases malleable, jobs, moldable, machines, unrelated, uniform, parallel, speeds, approximation, scheduling

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.17

Category APPROX

Related Version A full version of the paper is available at https://arxiv.org/abs/1903.11016.

Acknowledgements Part of this work was carried out while the authors participated in the program "Real-Time Decision Making" at the Simons Institute for the Theory of Computing, Berkeley, CA.



© Dimitris Fotakis, Jannik Matuschke, and Orestis Papadigenopoulos;
licensed under Creative Commons License CC-BY



1 Introduction

Since the late 60s, various models have been proposed by researchers [7, 8] in order to capture the real-world aspects and particularities of multiprocessor task scheduling systems, i.e., large collections of identical processors able to process tasks in parallel. High performance computing, parallel architectures, and cloud services are typical applications that motivate the study of multiprocessor scheduling, both theoretical and practical. An influential model is Rayward-Smith's unit execution time and unit communication time (UET-UCT) model [22], where each parallel job is partitioned into a set of tasks of unit execution time and these tasks are subject to precedence constraints modeled by a task graph. The UET-UCT model and its generalizations have been widely studied and a large number of (approximation) algorithms and complexity results have been proposed [10, 20].

However, the UET-UCT model mostly focuses on task scheduling and sequencing, and does not account for the amount of resources allocated to each job, thus failing to capture an important aspect of real-world parallel systems. Specifically, in the UET-UCT model, the level of granularity of a job (that is, the number of smaller tasks that a job is partitioned into) is decided a priori and is given as part of the input. However, it is common ground in the field of parallel processing that the unconditional allocation of resources for the execution of a job may jeopardize the overall efficiency of a multiprocessor system. A theoretical explanation is provided by Amdahl's law [1], which suggests that the speedup of a job's execution can be estimated by the formula $\frac{1}{(1-p)+\frac{p}{s}}$, where p is the fraction of the job that can be parallelized and s is the speedup due to parallelization (i.e., s can be thought as the number of processors).

Malleable Scheduling. An interesting alternative to the UET-UCT model is that of *malleable*[1] job scheduling [5, 24]. In this setting, a set $\mathcal{J}$ of jobs is scheduled on a set $\mathcal{M}$ of parallel machine(-s), while every job can be processed by more than one machines at the same time (i.e., by partitioning the job into tasks). In order to quantify the effect of parallelization, the processing time of a job $j \in \mathcal{J}$ is determined by a function $f_j : \mathbb{N} \to \mathbb{R}_+$[2] depending on the number of allocated machines. Moreover, every job must be executed *non-preemptively* and in *unison*, i.e. having the same starting and completion time on each of the allocated machines. Thus, if a job j is assigned to a set of machines S starting at time τ, all machines in S are occupied with job j during the interval $[\tau, \tau + f_j(|S|)]$. It is commonly assumed that the processing time function of a job exhibits two useful and well-motivated properties:

- For every job $j \in \mathcal{J}$, the processing time $f_j(s)$ is *non-increasing* in the number of machines.[3]
- The total *work* of the execution of a job j on s machines, that is the product $s \cdot f_j(s)$, is *non-decreasing* in the number of machines.

The latter property, known as *monotonicity* of a malleable job, is justified by Brent's law [3]: One cannot expect superlinear speedup by increasing the level of parallelism. A great deal of theoretical results have been published on scheduling malleable jobs according to the above model (and its variants) for the objective of minimizing the *makespan*, i.e., the completion time of the last finishing job, or other standard objectives (see, e.g., [6] and the references therein).

---

[1] Malleable scheduling also appears as *moldable*, while sometimes the two terms refer to slightly different models.

[2] We denote by $\mathbb{R}_+$ (resp. $\mathbb{Z}_+$) the set of non-negative reals (resp. integers).

[3] This property holds w.l.o.g., as the system always has the choice not to use some of the allocated machines.

Although malleable job scheduling represents a valiant attempt to capture real-world aspects of massively parallel processing, the latter exhibits even more complicated characteristics. Machine heterogeneity, data locality and hardware interconnection are just a few aspects of real-life systems that make the generalization of the aforementioned model necessary. In modern multiprocessor systems, machines are not all identical and the processing time of a job not only depends on the quantity, but also on the quality of the set of allocated machines. Indeed, different physical machines may have different capabilities in terms of faster CPUs or more efficient cache hierarchies. Moreover, the above heterogeneity may be job-dependent, in the sense that a specific machine may be faster when executing a certain type of jobs than another (e.g., memory- vs arithmetic-intensive applications [21]). Finally, the execution of a job on specific combinations of machines may also yield additional benefit (e.g., machines that are local in terms of memory hierarchy).

Our Model: Malleable Scheduling on Unrelated Machines. Quite surprisingly, no results exist on scheduling malleable jobs beyond the case of identical machines, to the best of our knowledge, despite the significant theoretical and practical interest in the model. In this work, we extend the model of malleable job scheduling to capture more delicate aspects of parallel job scheduling. In this direction, while we still require our jobs to be executed non-preemptively and in unison, the processing time of a job $j \in \mathcal{J}$ becomes a set function $f_j(S)$, where $S \subseteq \mathcal{M}$ is the set of allocated machines. We require that processing times are given by a *non-increasing* function, in the set function context, while additional assumptions on the scalability of f_j are made, in order to capture the *diminishing utility* property implied by Brent's law.

These assumptions naturally lead to a generalized malleable job setting, where processing times are given by non-increasing supermodular set functions $f_j(S)$, accessed by value queries. We show that makespan minimization in this general setting is inapproximable within $\mathcal{O}(|\mathcal{J}|^{1-\varepsilon})$ factors (unless $P = NP$, see Section 4.3). The general message of the proof is that unless we make some relatively strong assumptions on processing times (in the form e.g., of a relatively smooth gradual decrease in the processing time, as more machines are allocated), malleable job scheduling (even with monotone supermodular processing times) can encode combinatorial problems as hard as graph coloring.

Thus, inspired by (standard non-malleable) scheduling models on uniformly related and unrelated machines, we introduce the notion of *speed-implementable* processing time functions. For each machine i and each job j there is a *speed* $s_{i,j} \in \mathbb{Z}_+$ that quantifies the contribution of machine i to the execution of job j, if i is included in the set allocated to j. For most of this work, we assume that the total speed of an allocated set is given by an additive function $\sigma_j(S) = \sum_{i \in S} s_{i,j}$ (but see also Section 4.1, where we discuss more general speed functions based on L_p-norms). A function is speed-implementable if we can write $f_j(S) = f_j(\sigma_j(S))$ for some function $f_j : \mathbb{R}_+ \to \mathbb{R}_+$. Again, we assume oracle access to the processing time functions.[4]

The notion of speed-implementable processing times allows us to quantify the fundamental assumptions of *monotonicity* and *diminishing utility* in a clean and natural way. More specifically, we make the following two assumptions on speed-implementable functions:

1. *Non-increasing processing time.* For every job $j \in \mathcal{J}$, the processing time $f_j(s)$ is non-increasing in the total allocated speed $s \in \mathbb{R}_+$.
2. *Non-decreasing work.* For every job $j \in \mathcal{J}$, the work $f_j(s) \cdot s$ is non-decreasing in the total allocated speed $s \in \mathbb{R}_+$.

[4] For convenience, we use the identifier f_j for both functions. Since their arguments come from disjoint domains, it is always clear from the context which one is meant.

The first assumption ensures that allocating more speed cannot increase the processing time. The second assumption is justified by Brent's law, when the increase in speed coincides with an increase in the physical number of machines, or by similar arguments for the increase of the total speed of a single physical machine (e.g., memory access, I/O bottleneck [21] etc.). We remark that speed-implementable functions with non-increasing processing times and non-decreasing work do not need to be convex, and thus, do not belong to the class of supermodular functions.

In this work, we focus on the objective of minimizing the *makespan* $C_{max} = \max_{j \in \mathcal{J}} C_j$, where C_j the completion time of job j. We refer to this setting as the problem of *scheduling malleable jobs on unrelated machines*. To further justify this term, we present a pseudopolynomial transformation of standard scheduling on unrelated machines to malleable scheduling with speed-implementable processing times (see the full version of this reading). The reduction can be rendered polynomial by standard techniques, preserving approximation factors with a loss of $1 + \varepsilon$.

1.1 Related Work

The problem of malleable job scheduling on identical machines has been studied thoroughly for more than three decades. For the case of non-monotonic jobs, i.e., jobs that do not satisfy the monotonic work condition, Du and Leung [5] show that the problem is strongly NP-hard for more than 5 machines, while in terms of approximation, Turek, Wolf and Yu [24] provided the first 2-approximation algorithm for the same version of the problem. Jansen and Porkolab [13] devised a PTAS for instances with a constant number of machines, which was later extended by Jansen and Thöle [15] to a PTAS for the case that the number of machines is polynomial in the number of jobs.

For the case of monotonic jobs, Mounié, Rapine and Trystram [19] propose a $\frac{3}{2}$-approximation algorithm, improving on the $\sqrt{3}$-approximation provided by the same authors [18]. Recently, Jansen and Land [12] gave an FPTAS for the case that $|\mathcal{M}| \geq 8|\mathcal{J}|/\varepsilon$. Together with the approximation scheme for polynomial number of machines in [12], this implies a PTAS for scheduling monotonic malleable jobs on identical machines.

Several papers also consider the problem of scheduling malleable jobs with preemption and/or under precedence constraints [2, 14, 17]. An interesting alternative approach to the general problem is that of Srinivasa, Prasanna, and Musicus [23], who consider a continuous version of malleable tasks and develop an exact algorithm based on optimal control theory under certain assumptions on the processing time functions. While the problem of malleable scheduling on identical machines is very well understood, this is not true for malleable extensions of other standard scheduling models, such as unrelated machines or the restricted assignment model. We attempt to close this gap by introducing and investigating malleable scheduling with speed-implementable processing time functions.

A scheduling model similar to malleable tasks is that of *splittable jobs*. In this regime, jobs can be split arbitrarily and the resulting parts can be distributed arbitrarily on different machines. For each pair of job j and machine i, there is a setup time s_{ij} and a processing time p_{ij}. If a fraction $x_{ij} \in (0, 1]$ of job j is to be scheduled on machine i, the load that is incurred on the machine is $s_{ij} + p_{ij}x_{ij}$. Correa et al. [4] provide an $(1 + \varphi)$-approximation algorithm for this setting (where φ is the golden ratio), which is based on an adaptation of the classic LP rounding result by Lenstra, Shmoys, and Tardos [16] for the traditional unrelated machine scheduling problem. We remark that the generalized malleable setting considered in this paper also induces a natural generalization of the splittable setting beyond setup times, when dropping the requirement that jobs need to be executed in unison. As

in [4], we provide a rounding framework based on a variant of the assignment LP from [16]. However, the fact that processing times are only given implicitly as functions in our setting makes it necessary to very carefully choose the coefficients of the assignment LP in order to ensure a constant integrality gap. Furthermore, because jobs have to be executed in unison, we employ a more sophisticated rounding scheme in order to better utilize free capacity on different machines.

1.2 Contribution and Techniques

At the conceptual level, we introduce the notion of malleable jobs with speed-implementable processing times. Hence, we generalize the standard and well-studied setting of malleable job scheduling, in a direct analogy to fundamental models in scheduling theory (e.g., scheduling on *uniformly related* and *unrelated* machines). This new and much richer model gives rise to a large family of unexplored packing problems that may be of independent interest. All omitted proofs can be found in the full version of this paper (see `https://arxiv.org/abs/1903.11016`).

From a technical viewpoint, we investigate the computational complexity and the approximability of this new setting. To the best of our understanding, standard techniques used for makespan minimization in the setting of malleable job scheduling on identical machines, such as the two-shelve approach (as used in [19, 24]) and area charging arguments, fail to yield any reasonable approximation guarantees in our more general setting. This intuition is supported by the following hardness of approximation result.

▶ **Theorem 1.** *For any $\epsilon > 0$, there is no $(\frac{e}{e-1} - \epsilon)$-approximation algorithm for the problem of scheduling malleable jobs on unrelated machines, unless $P = NP$.*

Note that the lower bound of $\frac{e}{e-1}$ is strictly larger than the currently best known approximation factor of 1.5 for malleable scheduling on identical machines.

Our positive results are based on a linear programming relaxation, denoted by $[\mathrm{LP}(C)]$ and described in Section 2. This LP resembles the assignment LP for the standard setting of non-malleable scheduling [16]. However, in order to obtain a constant integrality gap we distinguish between "small" jobs that can be processed on a single machine (within a given target makespan), and "large" jobs that have to be processed on multiple machines. For the large jobs, we carefully estimate their contribution to the load of their allocated machines. Specifically, we introduce the notion of *critical speed* and use the critical speed to define the load coefficients incurred by large jobs on machines in the LP relaxation by proportionally distributing the work volume according to machine speeds. For the rounding, we exploit the sparsity of our relaxation's extreme points (as in [16]) and generalize the approach of [4], in order to carefully distinguish between jobs assigned to a single machine and jobs shared by multiple machines.

▶ **Theorem 2.** *There exists a polynomial-time $\frac{2e}{e-1}$-approximation algorithm for the problem of scheduling malleable jobs on unrelated machines.*

An interesting corollary is that for malleable job scheduling on unrelated machines, there always exists an approximate solution where each machine shares at most one job with some other machines. We also get improved approximation guarantees for the special cases of *restricted assignment* and *uniform speeds*, respectively, by exploiting the special structure of the processing time functions.

▶ **Theorem 3.** *There exists a polynomial-time $\frac{7}{3}$-approximation algorithm for the problem of scheduling malleable jobs on restricted identical machines (i.e., $s_{i,j} \in \{0,1\}$ for all $i \in \mathcal{M}$ and $j \in \mathcal{J}$).*

▶ **Theorem 4.** *There exists a polynomial-time 3-approximation algorithm for the problem of scheduling malleable jobs on uniform machines (i.e., $s_{i,j} = s_i$ for all $i \in \mathcal{M}$ and $j \in \mathcal{J}$).*

All our approximation results imply corresponding upper bounds on the integrality gap of the linear programming relaxation [LP(C)]. Based on an adaptation of a construction in [4], we show a lower bound of $1 + \varphi \approx 2.618$ on the integrality gap of [LP(C)] for malleable job scheduling on unrelated machines, where φ is the golden ratio. For the cases of restricted assignment and uniformly related machines, respectively, we obtain an integrality gap of 2.

Moreover, we extend our model and approach in two directions. First, we consider a setting where the *effective speed* according to which a set S of allocated machines processing a job j is given by the L_p-norm $\sigma_j^{(p)}(S) = \left(\sum_{i \in S}(s_{i,j})^p\right)^{1/p}$ of the corresponding speed vector. In practical settings, we tend to prefer assignments to relatively small sets of physical machines, so as to avoid delays related to communication, memory access, and I/O (see e.g., [21]). By replacing the total speed (i.e., the L_1-norm) with the L_p-norm of the speed vector for some $p \geq 1$, we discount the contribution of additional machines (especially of smaller speeds) towards processing a job j. Thus, as p increases, we give stronger preference to *sparse* schedules, where the number of jobs shared between different machines (and the number of machines sharing a job) are kept small. Interestingly, our general approach is robust to this generalization and results in constant approximation factors for any $p \geq 1$. Asymptotically, the approximation factor is bounded by $\frac{p}{p - \ln p} + \sqrt[p]{\frac{p}{\ln p}}$ and our algorithm smoothly converges to the algorithm of [16] as p tends to infinity. For the extreme case where we use the L_∞-norm, our setting becomes identical to standard scheduling on unrelated machines and we recover the algorithm of [16], achieving an approximation ratio of 2. These results are discussed in Section 4.1.

In another direction, we combine our approach for makespan minimization with standard techniques employed for the objective of total weighted completion time, $\sum_{j \in \mathcal{J}} w_j C_j$, and obtain a constant factor approximation for minimizing the total weighted completion time for malleable job scheduling on unrelated machines. These results are discussed in Section 4.2.

Trying to generalize malleable job scheduling beyond the simple setting of identical machines, as much as possible, we believe that our setting with speed-implementable processing times lies on the frontier of the constant-factor approximability regime. We show a strong inapproximability lower bound of $\mathcal{O}(|\mathcal{J}|^{1-\varepsilon})$ for the (far more general) setting where the processing times are given by a non-increasing supermodular set functions. These results are discussed in Section 4.3. An interesting open question is to characterize the class of processing time functions for which malleable job scheduling admits constant factor (and/or logarithmic) approximation guarantees.

2 The general rounding framework

In this section, we provide a high-level description of our algorithm. We construct a polynomial-time *ρ-relaxed decision procedure* for malleable job scheduling problems. This procedure takes as input an instance of the problem as well as a target makespan C and either asserts correctly that there is no feasible schedule of makespan at most C, or returns a feasible schedule of makespan at most ρC. It is well-known that a ρ-relaxed decision procedure can

be transformed into a polynomial-time ρ-approximation algorithm [11] provided that one can compute proper lower and upper bounds to the optimal value of size polynomial in the size of the input.

Given a target makespan C, let $\gamma_j(C) := \min\{q \in \mathbb{Z}_+ \mid f_j(q) \leq C\}$ be the *critical speed* of job $j \in \mathcal{J}$. Moreover, we define for every $i \in \mathcal{M}$ the sets $J_i^+(C) := \{j \mid f(s_{i,j}) \leq C\}$ and $J_i^-(C) := J \backslash J_i^+(C)$ to be the set of jobs that can or cannot be processed by i alone within time C, respectively. Note that $\gamma_j(C)$ can be computed in polynomial-time given oracle access to f_j by performing binary search. When C is clear from the context, we use the short-hand notation γ_j, J_i^+, and J_i^- instead. The following technical fact is equivalent to the non-decreasing work property and is used throughout the proofs of this paper:

▶ **Fact 5.** *Let f be a speed-implementable processing time function satisfying the properties of our problem. Then for every speed $q \in \mathbb{R}_+$ we have that:*

1. *$f(\alpha q) \leq \frac{1}{\alpha} f(q)$ for every $\alpha \in (0,1)$, and*
2. *$f(q') \leq \frac{q}{q'} f(q)$ for every $q' \leq q$.*

The following feasibility LP is the starting point of the relaxed decision procedures we construct in this work:

$$\text{[LP(C)]:} \qquad \sum_{i \in \mathcal{M}} x_{i,j} = 1 \qquad \forall j \in \mathcal{J} \tag{1}$$

$$\sum_{j \in J_i^+} f_j(s_{i,j}) x_{i,j} + \sum_{j \in J_i^-} \frac{f_j(\gamma_j)\gamma_j}{s_{i,j}} x_{i,j} \leq C \qquad \forall i \in \mathcal{M} \tag{2}$$

$$x_{i,j} \geq 0 \quad \forall j \in \mathcal{J}, i \in \mathcal{M} \tag{3}$$

In the above LP, each variable $x_{i,j}$ can be thought as the fraction of job j that is assigned to machine i. The equality constraints (1) ensure that each job is fully assigned to a subset of machines, while constraints (2) impose an upper bound to the load of every machine. As we can prove, the above formulation is feasible for any C that is greater than the optimal makespan.

▶ **Proposition 6.** *For every $C \geq \text{OPT}$, where OPT is the makespan of an optimal schedule, [LP(C)] has a feasible solution.*

Proof. Fix a schedule of makespan OPT and let $S_j \subseteq \mathcal{M}$ be the set of machines allocated to a job j in that schedule. For every $i \in \mathcal{M}$,$j \in \mathcal{J}$ set $x_{i,j} = \frac{s_{i,j}}{\sigma_j(S_j)}$ if $i \in S_j$ and $x_{i,j} = 0$, otherwise. We show that x is a feasible solution to $[LP(C)]$. Indeed, constraints (1) are satisfied since $\sum_{i \in \mathcal{M}} x_{i,j} = \sum_{i \in S_j} \frac{s_{i,j}}{\sigma_j(S_j)} = 1$ for all $j \in \mathcal{J}$. For verifying that constraints (2) are fulfilled, let $j \in \mathcal{J}$ and $i \in S_j$. If $j \in J_i^+$ then $f_j(s_{i,j}) x_{i,j} = f_j(s_{i,j}) \frac{s_{i,j}}{\sigma_j(S_j)} \leq f_j(S_j)$, using Fact 5. If $j \in J_i^-$ then $\frac{f_j(\gamma_j)\gamma_j}{s_{i,j}} x_{i,j} = \frac{f_j(\gamma_j)\gamma_j}{\sigma_j(S_j)} \leq \frac{\sigma_j(S_j) f_j(S_j)}{\sigma_j(S_j)} \leq f_j(S_j)$, again using Fact 5 and the fact that $\sigma_j(S_j) \geq \gamma_j$. Therefore for any $i \in \mathcal{M}$ we obtain: $\sum_{j \in J_i^+} f_j(s_{i,j}) x_{i,j} + \sum_{j \in J_i^-} \frac{f_j(\gamma_j)\gamma_j}{s_{i,j}} x_{i,j} \leq \sum_{j \in J | i \in S_j} f_j(S_j) \leq OPT \leq C$. ◀

Assuming that $C \geq \text{OPT}$, let x be an extreme point solution to [LP(C)]. We create the *assignment graph* $\mathcal{G}(x)$ with nodes $V := \mathcal{J} \cup \mathcal{M}$ and edges $E := \{\{i,j\} \in \mathcal{M} \times \mathcal{J} \mid x_{i,j} > 0\}$, i.e., one edge for each machine-job pair in the support of the LP solution. Notice that $\mathcal{G}(x)$ is bipartite by definition. Furthermore, since [LP(C)] is structurally identical to the LP of unrelated machine scheduling [16], the choice of x as an extreme point guarantees the following sparsity property:

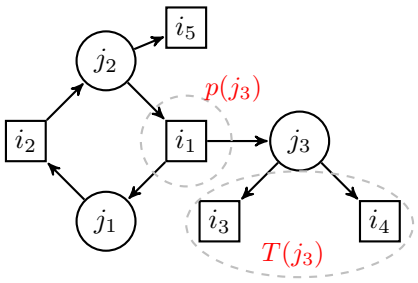


Figure 1 A properly oriented pseudotree with indegree at most 1 for each node.

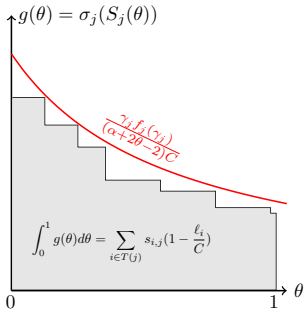


Figure 2 Volume argument for selecting a subset of the children machines in the proof of Proposition 10.

▶ **Proposition 7** ([16]). *For every extreme point solution x of [LP(C)], each connected component of $\mathcal{G}(x)$ contains at most one cycle.*

As a graph with at most one cycle is either a tree or a tree plus one edge, the connected components of $\mathcal{G}(x)$ are called *pseudotrees* and the whole graph is called a *pseudoforest*. It is not hard to see that the edges of an undirected pseudoforest can always be oriented in a way that every node has an *in-degree* of at most one. We call such a $\mathcal{G}(x)$ a *properly oriented* pseudoforest. Such an orientation can easily be obtained by first orienting the edges on the unique cycle (if it exists) consistently so as to obtain a directed cycle and, then, by orienting all remaining edges away from that cycle (see Figure 1).

Now fix a properly oriented $\mathcal{G}(x)$ with set of oriented edges $\bar{E}$. For $j \in \mathcal{J}$, we define $p(j) \in \mathcal{M}$ to be its unique *parent-machine* (if it exists) and $T(j) = \{i \in \mathcal{M} \mid (j,i) \in \bar{E}\}$ to be the set of *children-machines* of j, respectively. Notice, that for every machine i, there exists at most one $j \in \mathcal{J}$ such that $i \in T(j)$. The decision procedures we construct in this paper are based, unless otherwise stated, on the following scheme:

Algorithm: Given a target makespan C:

1. If [LP(C)] is feasible, compute an extreme point solution x of [LP(C)] and construct a properly oriented $\mathcal{G}(x)$. (Otherwise, report that $C < \mathrm{OPT}$.)
2. A *rounding scheme* assigns every job $j \in \mathcal{J}$ either only to its parent machine $p(j)$, or to the set of its children-machines $T(j)$ (see Section 3).
3. According to the rounding, every job $j \in \mathcal{J}$ that has been assigned to $T(j)$ is placed at the beginning of the schedule (these jobs are assigned to disjoint sets of machines).
4. At any point a machine i becomes idle, it processes any unscheduled job j that has been rounded to i such that $i = p(j)$.

3 Rounding schemes

In each of the following rounding schemes, we are given as an input an extreme point solution x of [LP(C)] and a properly oriented pseudoforest $\mathcal{G}(x) = (V, \bar{E})$.

3.1 A simple 4-approximation for unrelated machines

We start from the following simple rounding scheme: For each job j, assign j to its parent-machine $p(j)$ if $x_{p(j),j} \geq \frac{1}{2}$, or else, assign j to its children-machines $T(j)$. Formally, let $\mathcal{J}^{(1)} := \{j \in \mathcal{J} \mid x_{p(j),j} \geq \frac{1}{2}\}$ be the sets of jobs that are assigned to their parent-machines and $\mathcal{J}^{(2)} := \mathcal{J} \setminus \mathcal{J}^{(1)}$ the rest of the jobs. Recall that we first run the jobs

in $J^{(2)}$ and then the jobs in $J^{(1)}$ as described at the end of the previous section. For $i \in \mathcal{M}$, define $J_i^{(1)} := \{j \in \mathcal{J}^{(1)} \mid p(j) = i\}$ and $J_i^{(2)} := \{j \in \mathcal{J}^{(2)} \mid i \in T(j)\}$ as the sets of jobs in $J^{(1)}$ and $J^{(2)}$, respectively, that get assigned to i (note that $|J_i^{(2)}| \leq 1$, as each machine gets assigned at most one job as a child-machine). Furthermore, let $\ell_i := \sum_{j \in J_i^+ \cap \mathcal{J}^{(1)}} f_j(s_{i,j}) x_{i,j} + \sum_{j \in J_i^- \cap \mathcal{J}^{(1)}} f_j(\gamma_j) \frac{\gamma_j}{s_{i,j}} x_{i,j}$ be the fractional load incurred by jobs in $J^{(1)}$ on machine i in the LP solution x.

▶ **Proposition 8.** *Let $i \in \mathcal{M}$. Then $\sum_{j \in \mathcal{J}_i^{(1)}} f_j(\{i\}) \leq 2\ell_i$.*

Proof. Let $j \in J_i^{(1)}$. Since $x_{i,j} \geq \frac{1}{2}$ by definition of $\mathcal{J}^{(1)}$, we get $f_j(s_{i,j}) \leq 2 f_j(s_{i,j}) x_{i,j}$. Furthermore, if $j \in J_i^-$ then $f_j(s_{i,j}) x_{i,j} \leq f_j(\gamma_j) \frac{\gamma_j}{s_{i,j}} x_{i,j}$ by Fact 5 and the fact that $s_{i,j} < \gamma_j$. Thus, by summing up over all jobs in $J_i^{(1)}$ and then applying constraints (2), we get

$$\sum_{j \in J_i^{(1)}} f_j(\{i\}) \leq 2 \left(\sum_{j \in J_i^{(1)} \cap J_i^+} f_j(s_{i,j}) x_{i,j} + \sum_{j \in J_i^{(1)} \cap J_i^-} \frac{f_j(\gamma_j)\gamma_j}{s_{i,j}} x_{i,j} \right) \leq 2\ell_i. \qquad \blacktriangleleft$$

▶ **Proposition 9.** *Let $j \in \mathcal{J}^{(2)}$. Then $f_j(T(j)) \leq 2C$.*

Proof. If there is a machine $i \in T(j)$ with $j \in J_i^+$, then $f_j(T(j)) \leq f_j(\{i\}) \leq C$. So we can assume that $j \in J_i^-$ for all $i \in T(j)$. Hence constraints (2) imply $f_j(\gamma_j) \frac{\gamma_j}{s_{i,j}} x_{i,j} \leq C$ for all $i \in T(j)$. Summing these constraints yields $\sum_{i \in T(j)} \frac{f_j(\gamma_j)}{C} \gamma_j x_{i,j} \leq \sigma_j(T(j))$. Using the fact that $f_j(\gamma_j) \leq C$ by definition of γ_j and $\sum_{i \in T(j)} x_{i,j} > \frac{1}{2}$ because $j \in \mathcal{J}^{(2)}$, we get $\sigma_j(T(j)) \geq \frac{1}{2} \gamma_j \frac{f_j(\gamma_j)}{C}$. This implies $f_j(T(j)) \leq 2C$ by Fact 5. $\blacktriangleleft$

Clearly, the load of any machine $i \in \mathcal{M}$ in the final schedule is the sum of the load due to the execution of $\mathcal{J}^{(1)}$, plus the processing time of at most one job of $\mathcal{J}^{(2)}$. By Proposition 8 and 9, it follows that any feasible solution of [LP(C)] can be rounded in polynomial-time into a feasible schedule of makespan at most $4C$.

3.2 An improved $\frac{2e}{e-1} \approx 3.163$-approximation for unrelated machines

In the simple rounding scheme described above, it can be the case that the overall makespan improves by assigning some job $j \in \mathcal{J}^{(2)}$ only to a subset of the machines in $T(j)$. This happens because some machines in $T(j)$ may have significantly higher load from jobs of $\mathcal{J}^{(1)}$ than others, but job j will incur the same additional load to all machines it is assigned to.

We can improve the approximation guarantee of the rounding scheme by taking this effect into account and filtering out children-machines with a high load. Define $\mathcal{J}^{(1)}$ and $\mathcal{J}^{(2)}$ as before. Every job in $j \in J^{(1)}$ is assigned to its parent-machine $p(j)$, while every job $j \in \mathcal{J}^{(2)}$ is assigned to a subset of $T(j)$ as follows.

For $j \in \mathcal{J}^{(2)}$ and $\theta \in [0,1]$ define $S_j(\theta) := \{i \in T(j) \mid 1 - \frac{\ell_i}{C} \geq \theta\}$. Choose θ_j so as to minimize $2(1-\theta_j)C + f_j(\theta_j)$ (note that this minimizer can be determined by trying out at most $|T(j)|$ different values for θ_j). We then assign each job in $j \in J^{(2)}$ to the machine set $S_j(\theta_j)$.

By Proposition 8, we know that the total load of each machine $i \in \mathcal{M}$ due to the execution of jobs from $\mathcal{J}^{(1)}$ is at most $2\ell_i$. Recall that there is at most one $j \in J^{(2)}$ with $i \in T(j)$. If $i \notin S_j(\theta_j)$, then load of machine i bounded by $2\ell_i \leq 2C$. If $i \in S_j(\theta_j)$, then the load of machine i is bounded by

$$\max_{i' \in S_j(\theta_j)} \left\{ 2\ell_{i'} + f_j(S_j(\theta_j)) \right\} \leq 2(1-\theta_j)C + f_j(S_j(\theta_j)), \tag{4}$$

where the inequality comes from the fact that $1 - \frac{\ell_{i'}}{C} \geq \theta_j$ for all $i' \in S_{\theta_j}$. The following proposition gives an upper bound on the RHS of (4) as a result of our filtering technique and proves Theorem 2.

▶ **Proposition 10.** *For each $j \in \mathcal{J}^{(2)}$, there is a $\theta \in [0,1]$ with $2(1-\theta)C + f_j(S_j(\theta)) \leq \frac{2e}{e-1}C$.*

Proof. Define $\alpha := \frac{2e}{e-1}$. We show that there is a $\theta \in [0,1]$ with $\sigma_j(S_j(\theta)) \geq \frac{\gamma_j f_j(\gamma_j)}{(\alpha+2\theta-2)C}$. Then $f_j(S_j(\theta)) \leq (\alpha + 2\theta - 2)C$ by Fact 5, implying the lemma.

Define the function $g : [0,1] \to \mathbb{R}_+$ by $g(\theta) := \sigma_j(S_j(\theta))$. It is easy to see g is non-increasing integrable and that

$$\int_0^1 g(\theta)d\theta = \sum_{i \in T(j)} s_{i,j}(1 - \frac{\ell_i}{C}).$$

See Figure 2 for an illustration.

Now assume by contradiction that $g(\theta) < \frac{\gamma_j f_j(\gamma_j)}{(\alpha+2\theta-2)C}$ for all $\theta \in [0,1]$. Note that $\ell_i + \frac{\gamma_j f_j(\gamma_j)}{s_{i,j}} x_{i,j} \leq C$ for every $i \in T(j)$ by constraints (2) and the fact that $\frac{\gamma_j f_j(\gamma_j)}{s_{i,j}} \leq f_j(s_{i,j})$ for all i such that $j \in J_i^+$. Hence $\frac{f_j(\gamma_j)\gamma_j}{C} x_{i,j} \leq s_{i,j}(1 - \frac{\ell_i}{C})$ for all $i \in T(j)$. Summing over all $i \in T(j)$ and using the fact that $\sum_{i \in T(j)} x_{i,j} \geq \frac{1}{2}$ because $j \in J^{(2)}$ we get

$$\frac{f_j(\gamma_j)\gamma_j}{2C} \leq \sum_{i \in T(j)} s_{i,j}(1 - \frac{\ell_i}{C}) = \int_0^1 g(\theta)d\theta < \frac{f_j(\gamma_j)\gamma_j}{C} \int_0^1 \frac{1}{\alpha + 2\theta - 2} d\theta,$$

where the last inequality uses the assumption that $g(\theta) < \frac{\gamma_j f_j(\gamma_j)}{(\alpha+2\theta_j-2)C}$ for all $\theta \in [0,1]$. By simplifying the above inequality, we get the contradiction

$$1 < \int_{\alpha-2}^{\alpha} \frac{1}{\lambda} d\lambda = \ln(\frac{\alpha}{\alpha-2}) = 1. \qquad \blacktriangleleft$$

By the above analysis, our main result for the case of unrelated machines follows.

▶ **Theorem 2.** *There exists a polynomial-time $\frac{2e}{e-1}$-approximation algorithm for the problem of scheduling malleable jobs on unrelated machines.*

▶ Remark 11. We can slightly improve the above analysis by optimizing the threshold of assigning a job to the parent. This optimization gives a slightly better approximation guarantee of $\alpha = \inf_{\beta \in (0,1)} \left\{ \frac{e^{\frac{1}{\beta}-1}}{\beta(e^{\frac{1}{\beta}-1}-1)} \right\} \approx 3.14619$.

3.3 A (7/3)-approximation for restricted identical machines

We are able to provide an algorithm of improved approximation guarantee for the special case of restricted identical machines: Each job $j \in \mathcal{J}$ is associated with a set of machines $\mathcal{M}_j \subseteq \mathcal{M}$, such that $s_{i,j} = 1$ for $i \in \mathcal{M}_j$ and $s_{i,j} = 0$, otherwise.

Given a feasible solution to [LP(C)] and a properly oriented $\mathcal{G}(x)$, we define the sets $\mathcal{J}^{(1)} := \{j \in \mathcal{J} \mid x_{p(j),j} = 1\}$ and $\mathcal{J}^{(2)} := \mathcal{J} \setminus \mathcal{J}^{(1)}$. The rounding scheme for this special case can be described as follows: **(a)** Every job $j \in \mathcal{J}^{(1)}$ is assigned to $p(j)$ (which is the only machine in $\mathcal{G}(x)$ that is assigned to j). **(b.i)** Every job of $j \in \mathcal{J}^{(2)}$ such that $|T(j)| = 1$ or $|T(j)| \geq 3$ is assigned to the set $T(j)$ of its children-machines. **(b.ii)** For every job of $j \in \mathcal{J}^{(2)}$ such that $|T(j)| = 2$, the algorithm schedules the job to the subset $S \subseteq T(j)$ that results in the minimum makespan over $T(j)$. Notice that for $|T(j)| = 2$ there are exactly three such subsets. As usual, the jobs of $\mathcal{J}^{(2)}$ are placed at the beginning of the schedule, followed by the jobs of $\mathcal{J}^{(1)}$.

▶ **Theorem 3.** *There exists a polynomial-time $\frac{7}{3}$-approximation algorithm for the problem of scheduling malleable jobs on restricted identical machines (i.e., $s_{i,j} \in \{0,1\}$ for all $i \in \mathcal{M}$ and $j \in \mathcal{J}$).*

3.4 A 3-approximation for uniform machines

We prove an algorithm of improved approximation guarantee for the special case of uniform machines, i.e., every machine has a unique speed s_i such that $s_{i,j} = s_i$ for all $j \in \mathcal{J}$. Given a target makespan C, we say that a machine i is *j-fast* for a job $j \in \mathcal{J}$ if $j \in J_i^+$, while we say that i is *j-slow* if $j \in J_i^-$. As opposed to the previous cases, the rounding for the uniform case starts by transforming the feasible solution of [LP(C)] into another extreme point solution that satisfies a useful structural property, as described in the following proposition.

▶ **Proposition 12.** *There is an extreme point solution x of [LP(C)] that satisfies the following property: For each $j \in \mathcal{J}$ there is at most one j-slow machine $i \in \mathcal{M}$ such that $x_{i,j} > 0$ and $x_{i,j'} > 0$ for some job $j' \neq j$. Furthermore, this machine, if it exists, is the slowest machine that j is assigned to.*

Let x be an extreme point solution of [LP(C)] that satisfies the property of Proposition 12 and let $\mathcal{G}(x)$ a properly oriented pseudoforest. By the above proposition, each job j has at most three types of assignments in $\mathcal{G}(x)$: (i) j-fast machines F_j, (ii) *exclusive* j-slow machines D_j, i.e. j-slow machines that are completely assigned to j, and (iii) at most one shared j-slow machine i_j (which is the slowest machine that j is assigned to).

We now describe the rounding scheme for the special case of uniform machines. For any job $j \in \mathcal{J}$ in any order: **(a)** If $x_{p(j),j} \geq \frac{1}{2}$ then j is assigned to its parent-machine $p(j)$, otherwise **(b)** j is assigned to a subset $S \subseteq T(j)$. In the second case, the subset S is chosen according to the following rule: **(b.i)** If $F_j \neq \emptyset$, then assign j to any $i \in F_j$, else **(b.ii)** if $\sigma_j(D_j) \geq \frac{\gamma_j f_j(\gamma_j)}{3C}$, then assign j only to the machines of D_j (but not to the shared i_j). In any other case, **(b.iii)** j is assigned to $D_j \cup \{i_j\}$.

▶ **Theorem 4.** *There exists a polynomial-time 3-approximation algorithm for the problem of scheduling malleable jobs on uniform machines (i.e., $s_{i,j} = s_i$ for all $i \in \mathcal{M}$ and $j \in \mathcal{J}$).*

4 Model extensions and discussion

4.1 Sparse allocations via p-norm regularization

In the model of speed-implementable processing time functions that we study in the previous sections, each function $f_j(S)$ depends on the total additive speed, yet is oblivious to the actual number of allocated machines. However, the overhead incurred by the synchronization of physical machines naturally depends on their number and we therefore need to take into account both the total speed and the cardinality of the machine set allocated to a job. In this section, we model the impact of the number of machines through the notion of *effective speed*. In this setting, every job j is associated with a speed *regularizer* $p_j \geq 1$, while the total speed of a set $S \subseteq \mathcal{M}$ is given by: $\sigma_j^{(p_j)}(S) = \left(\sum_{i \in S} s_{i,j}^{p_j}\right)^{\frac{1}{p_j}}$. For simplicity, we assume that every job has the same speed regularizer, $p = p_j, \forall j \in \mathcal{J}$.

Clearly, the choice of p controls the effect of the cardinality of a set to the resulting speed of an allocation, given that as p increases a sparse set has higher effective speed than a non-sparse set of the same total speed. Notice that for $p = 1$ we return to the standard case of additive speeds, while for $p \to \infty$, parallelization is no longer helpful

as $\lim_{p\to\infty} \sigma_j^{(p)}(S) = \max_{i\in S}\{s_{i,j}\}$. As before, the processing time functions satisfy the standard properties of malleable scheduling, i.e., $f_j(s)$ is non-increasing while $f_j(s)\cdot s$ is decreasing. For simplicity of presentation we assume that all jobs have the same regularizer p, but we comment on the case of job-dependent regularizers at the end of this section.

Quite surprisingly, we can easily modify the algorithms of the previous section in order to capture the above generalization. Given a target makespan C, we start from a new feasibility program [LP$^{(p)}(C)$], which is given by constraints (1),(3) of [LP(C)], combined with:

$$\sum_{j\in J_i^+} f_j(s_{i,j})x_{i,j} + \sum_{j\in J_i^-} f_j(\gamma_j)\left(\frac{\gamma_j}{s_{i,j}}\right)^p x_{i,j} \leq C \qquad \forall i \in \mathcal{M} \tag{5}$$

Note that J_i^+, J_i^-, and $\gamma_j(C)$ are defined exactly as before, and that the only difference between [LP(C)] and [LP$^{(p)}(C)$] is that we replace the coefficient $\frac{\gamma_j}{s_{i,j}}$ with $\left(\frac{\gamma_j}{s_{i,j}}\right)^p$ in constraints (2) of the former. It can be shown that for every $C \geq OPT$, where OPT is the makespan of an optimal schedule, [LP$^{(p)}$(C)] has a feasible solution.

The algorithm for this case is similar to the one of the standard case (see Section 3.1), having [$LP^{(p)}(C)$] as a starting point. Moreover, the rounding scheme is a parameterized version of the simple rounding of Section 3.1, with the difference that the threshold parameter $\beta \in [0,1]$ (i.e., the parameter that controls the decision of assigning a job j to either $p(j)$ or $T(j)$) is not necessarily $\frac{1}{2}$. In short, given a pseudoforest $\mathcal{G}(x)$, the rounding scheme assigns any job j to $p(j)$ if $x_{p(j),j} \geq \beta$, or to $T(j)$, otherwise.

By similar arguments as in Propositions 8,9, it can be proved that the makespan of the produced schedule is at most $\left(\frac{1}{\beta} + \frac{1}{(1-\beta)^{1/p}}\right)C$. Therefore, the algorithm can initially compute a threshold $\beta \in [0,1]$ that minimizes the above theoretical bound. Clearly, for $p = 1$ the minimizer of the expression is $\beta = 1/2$, yielding the 4-approximation of the standard case, while for $p \to +\infty$ one can verify that $\beta \to 1$ and:

$$\lim_{p\to+\infty} \inf_{\beta\in[0,1]} \left(\frac{1}{\beta} + \frac{1}{(1-\beta)^{1/p}}\right) = 2.$$

As expected, for the limit case where $p \to +\infty$, our algorithm converges to the well-known algorithm by Lenstra et al. [16] given that our problem becomes non-malleable. By using the standard approximation $\beta = 1 - \frac{\ln p}{p}$ for $p \geq 2$, we can prove the following theorem.

▶ **Theorem 13.** *Any feasible solution of [$LP^{(p)}(C)$] for $p \geq 2$ can be rounded in polynomial-time into a feasible schedule of makespan at most* $\left(\frac{p}{p-\ln p} + \sqrt[p]{\frac{p}{\ln p}}\right)C$.

Note that an analogous approach can handle the case where jobs have different regularizers, with the approximation ratio for this scenario determined by the smallest regularizer that appears in the instance (note that the approximation factor is always at most 4).

4.2 Minimizing the $\sum_{j\in\mathcal{J}} w_j C_j$ objective

The LP-based nature of our algorithms allows the design of efficient $\mathcal{O}(1)$-approximation algorithms for the objective of minimizing the sum of weighted completion times, i.e., $\sum_{j\in\mathcal{J}} w_j C_j$, employing the standard technique of *interval-indexed formulations* [9]. In this setting, every job $j \in \mathcal{J}$ is associated with a *weight* $w_j \in \mathbb{Z}_{\geq 0}$ and the objective is to compute a feasible schedule of minimum $\sum_{j\in\mathcal{J}} w_j C_j$, where C_j the completion time of job j. In the malleable setting, the approximation guarantee of our algorithm for the $\sum_{j\in\mathcal{J}} w_j C_j$ objective depends on the approximation guarantee of the underlying makespan problem.

▶ **Theorem 14.** *There exists a $\mathcal{O}(\rho)$-approximation algorithm for the problem of malleable scheduling minimizing the $\sum_{j\in\mathcal{J}} w_j C_j$ objective, where ρ the approximation ratio of the best rounding scheme of $[LP(C)]$.*

4.3 Supermodular processing time functions

In this paper we concentrated our study on speed-implementable processing time functions. However, the general definition of malleable scheduling given in Section 1 leaves room for many other possible variants of the problem with processing times given by monotone non-increasing set functions. One natural attempt of capturing the assumption of non-decreasing workload is to assume that for each job $j \in \mathcal{J}$ the corresponding processing time function f_j is supermodular, i.e.,

$$f_j(T \cup \{i\}) - f_j(T) \geq f_j(S \cup \{i\}) - f_j(S)$$

for all $S \subseteq T \subseteq \mathcal{M}$ and $i \in \mathcal{M} \setminus T$. The interpretation of this assumption is that the decrease in processing time when adding machine i diminishes the more machines are already used for job j (note that the terms on both sides of the inequality are non-positive because f_j is non-increasing). For this setting, which we refer to as *generalized malleable scheduling with supermodular processing time functions*, we derive a strong hardness of approximation result.

▶ **Theorem 15.** *There is no $|\mathcal{J}|^{1-\varepsilon}$-approximation for generalized malleable scheduling with supermodular processing time functions, unless $P = NP$.*

References

1 G. M. Amdahl. Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities. *IEEE Solid-State Circuits Society Newsletter*, 12(3):19–20, Summer 2007.

2 J. Blazewicz, M. Y. Kovalyov, M. Machowiak, D. Trystram, and J. Weglarz. Preemptable Malleable Task Scheduling Problem. *IEEE Trans. Comput.*, 55(4):486–490, April 2006.

3 R. P. Brent. The Parallel Evaluation of General Arithmetic Expressions. *J. ACM*, 21(2):201–206, April 1974.

4 J. Correa, A. Marchetti-Spaccamela, J. Matuschke, L. Stougie, O. Svensson, V. Verdugo, and J. Verschae. Strong LP formulations for scheduling splittable jobs on unrelated machines. *Mathematical Programming*, 154(1-2):305–328, 2015.

5 J. Du and J. Leung. Complexity of Scheduling Parallel Task Systems. *SIAM Journal on Discrete Mathematics*, 2(4):473–487, 1989.

6 P. Dutot, G. Mounié, and D. Trystram. Scheduling Parallel Tasks: Approximation Algorithms. In Joseph T. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, chapter 26, pages 26–1–26–24. CRC Press, 2004.

7 M. Garey and R. Graham. Bounds for Multiprocessor Scheduling with Resource Constraints. *SIAM Journal on Computing*, 4(2):187–200, 1975.

8 R. Graham. Bounds on Multiprocessing Timing Anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.

9 L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein. Scheduling to Minimize Average Completion Time: Off-Line and On-Line Approximation Algorithms. *Math. Oper. Res.*, 22(3):513–544, August 1997.

10 C. Hanen and A. Munier. An approximation algorithm for scheduling dependent tasks on m processors with small communication delays. *Discrete Applied Mathematics*, 108(3):239–257, 2001.

11 D. S. Hochbaum and D. B. Shmoys. Using dual approximation algorithms for scheduling problems: Theoretical and practical results. In *26th Annual Symposium on Foundations of Computer Science*, FOCS '85, pages 79–89, October 1985.

12 K. Jansen and F. Land. Scheduling Monotone Moldable Jobs in Linear Time. In *2018 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2018, Vancouver, BC, Canada, May 21-25, 2018*, pages 172–181, 2018.

13 K. Jansen and L. Porkolab. Linear-Time Approximation Schemes for Scheduling Malleable Parallel Tasks. *Algorithmica*, 32(3):507–520, March 2002.

14 K. Jansen and H. Zhang. An Approximation Algorithm for Scheduling Malleable Tasks Under General Precedence Constraints. *ACM Trans. Algorithms*, 2(3):416–434, July 2006.

15 Klaus Jansen and Ralf Thöle. Approximation algorithms for scheduling parallel jobs. *SIAM Journal on Computing*, 39(8):3571–3615, 2010.

16 J. K. Lenstra, D. B. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46(1):259–271, January 1990.

17 Konstantin Makarychev and Debmalya Panigrahi. Precedence-Constrained Scheduling of Malleable Jobs with Preemption. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 823–834, 2014.

18 G. Mounié, C. Rapine, and D. Trystram. Efficient Approximation Algorithms for Scheduling Malleable Tasks. In *Proceedings of the Eleventh Annual ACM Symposium on Parallel Algorithms and Architectures, SPAA '99, Saint-Malo, France, June 27-30, 1999*, pages 23–32, 1999.

19 G. Mounié, C. Rapine, and D. Trystram. A 3/2-Approximation Algorithm for Scheduling Independent Monotonic Malleable Tasks. *SIAM J. Comput.*, 37(2):401–412, 2007.

20 C. H. Papadimitriou and M. Yannakakis. Towards an Architecture-independent Analysis of Parallel Algorithms. *SIAM J. Comput.*, 19(2):322–328, April 1990.

21 D. A. Patterson and J. L. Hennessy. *Computer Organization and Design, Fifth Edition: The Hardware/Software Interface*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, 2013.

22 V. J. Rayward-Smith. UET Scheduling with Unit Interprocessor Communication Delays. *Discrete Appl. Math.*, 18(1):55–71, November 1987.

23 G. N. Srinivasa Prasanna and B. R. Musicus. Generalised Multiprocessor Scheduling Using Optimal Control. In *Proceedings of the Third Annual ACM Symposium on Parallel Algorithms and Architectures*, SPAA '91, pages 216–228, New York, NY, USA, 1991. ACM.

24 J. Turek, J. L. Wolf, and P. S. Yu. Approximate Algorithms Scheduling Parallelizable Tasks. In *Proceedings of the Fourth Annual ACM Symposium on Parallel Algorithms and Architectures*, SPAA '92, pages 323–332, New York, NY, USA, 1992. ACM.

On the Cost of Essentially Fair Clusterings

Ioana O. Bercea
School of Electrical Engineering
Tel Aviv University, Israel
ioana@cs.umd.edu

Martin Groß
School of Business and Economics
RWTH Aachen, Germany
martin.gross@mailbox.org

Samir Khuller
Department of Computer Science
Northwestern University, Evanston, USA
samir.khuller@northwestern.edu

Aounon Kumar
Department of Computer Science
University of Maryland, College Park, USA
aounon@umd.edu

Clemens Rösner
Institute of Computer Science
University of Bonn, Germany
roesner@cs.uni-bonn.de

Daniel R. Schmidt
Institute of Computer Science
University of Cologne, Germany
schmidt@informatik.uni-koeln.de

Melanie Schmidt
Institute of Computer Science
University of Bonn, Germany
melanieschmidt@uni-bonn.de

Abstract

Clustering is a fundamental tool in data mining and machine learning. It partitions points into groups (clusters) and may be used to make decisions for each point based on its group. However, this process may harm protected (minority) classes if the clustering algorithm does not adequately represent them in desirable clusters – especially if the data is already biased.

At NIPS 2017, Chierichetti et al. [18] proposed a model for *fair clustering* requiring the representation in each cluster to (approximately) preserve the global fraction of each protected class. Restricting to two protected classes, they developed both a 4-approximation for the fair k-center problem and a $\mathcal{O}(t)$-approximation for the fair k-median problem, where t is a parameter for the fairness model. For multiple protected classes, the best known result is a 14-approximation for fair k-center [40].

We extend and improve the known results. Firstly, we give a 5-approximation for the fair k-center problem with multiple protected classes. Secondly, we propose a relaxed fairness notion under which we can give bicriteria constant-factor approximations for all of the classical clustering objectives k-center, k-supplier, k-median, k-means and facility location. The latter approximations are achieved by a framework that takes an arbitrary existing unfair (integral) solution and a fair (fractional) LP solution and combines them into an essentially fair clustering with a weakly supervised rounding scheme. In this way, a fair clustering can be established belatedly, in a situation where the centers are already fixed.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Facility location and clustering; Theory of computation → Rounding techniques; Theory of computation → Unsupervised learning and clustering

Keywords and phrases approximation, clustering, fairness, LP rounding

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.18

Category APPROX

Related Version `arXiv:1811.10319 [cs.DS]`

Acknowledgements The authors would like to thank Sorelle Friedler for useful discussions related to the topic of fairness. The first author's work was done while a Ph.D. student at the University of Maryland.



© Ioana O. Bercea, Martin Groß, Samir Khuller, Aounon Kumar, Clemens Rösner, Daniel R. Schmidt, and Melanie Schmidt;
licensed under Creative Commons License CC-BY



1 Introduction

Suppose we are to reorganize school assignments in a big city. Given a long list of children starting school next year and a short list of all available teachers, the goal is to assign the students-to-be to (public) schools such that the maximum distance to the school is small. The school capacity is given by the number of its teachers: For each teacher, s students can be admitted. This challenge is in fact an instance of the capacitated (metric) k-center problem. So using a k-center algorithm, you obtain a solution. However, by chance you notice an odd occurrence: One school has a huge excess of boys, while another has a surplus of girls. From previous assignment iterations, you remember that the schools prefer more balanced classes.

Thus a new challenge arises: Assign the children such that the ratio is (approximately) 1:1 between boys and girls, and minimize the maximum distance under this condition.[1] This can be modeled by the following combinatorial optimization problem: Given a point set, half of the points are red, the other half is blue. Compute a clustering where each cluster has an equal number of red and blue points, and minimize the maximum radius.

In this form, our example is a special case of the *fair k-center* problem, as proposed by Chierichetti et al. [18] in the context of maintaining fairness in *unsupervised* machine learning tasks. Their model is based on the concept of *disparate impact* [39] (and the p%-rule). The input points are assumed to have a binary sensitive attribute modeled by two colors, and discrimination based on this attribute is to be avoided. Since preserving exact balance in each cluster may be very costly or even be impossible[2], the idea is to ensure that at least $1/t$ of the points of each cluster are of the minority color, where t is a parameter. A cluster with this property is called *fair*, and the fairness constraint can now be added to any clustering problem, giving rise to fair k-center, fair k-median, etc. Chierichetti et al. [18] develop a 4-approximation for a special case of fair k-center and a $(t + \sqrt{3} + \epsilon)$-approximation for one case of fair k-median.

The fair clustering model as proposed by Chierichetti et al. [18] can also be used to incorporate other aspects into our school assignment example: For example, we might want to mitigate effects of gentrification or segregation. For these use cases, we need multiple colors. Then, in each cluster, the ratio between the number of points with one specific color and the total number of points shall be in some given range. If the allowed range is $[0.20, 0.25]$ for red points, we require that in each cluster, at least a fifth and at most a fourth of the points are red. This models well established notions of fairness (statistical parity, group fairness), which require that each cluster exhibits the same compositional makeup as the overall data with respect to a given attribute. One downside of this notion is that a malicious user could create an illusion of fairness by including proxy points: If we wanted to create an boy-heavy school in our above example, we could still achieve the desired parity by assigning only girls that are very unlikely to attend. Thus, instead of enforcing *equal representation* in the above sense, one could also ask for *equal opportunity* as proposed by Hardt et al. [24] for the case where we take binary decisions (i.e., $k = 2$) and have access to a labeled training set. This approach, however, raises the philosophical question if this equality of opportunity is a sufficient condition for the absence of discrimination. Rather than delving into this complex and much debated issue in this algorithmic paper, we refer to

[1] Or, incorporating the capacities, ensure that the teacher:boys:girls ratio is $1{:}\frac{s}{2}{:}\frac{s}{2}$.
[2] Imagine a point set with 49 red and 51 blue points: This cannot at all be divided into true subsets with exactly the same ratio.

the excellent surveys by Romei and Ruggieri [39] and Žliobaitė et al. [43] that systematically discuss different forms of discrimination and how they can be detected. We assume that it is the intent of the user to achieve a truly fair solution.

Finding fair clusterings turns out to be an interesting challenge from the point of view of combinatorial optimization. As other clustering problems with side constraints, it loses the property that points can be assigned locally. But while many other constraint problems at least allow polynomial algorithms that assign points to given centers optimally, we show that even this restricted problem is NP-hard in the case of fair k-center.

Chierichetti et al. [18] tackle fair clustering problems by a two-step procedure: First, they compute a micro clustering into so-called *fairlets*, which are groups of points that are fair and cannot be split further into true subsets that are also fair. Secondly, representative points of the fairlets are clustered by an approximation algorithm for the unconstrained problem. Consider the special case of a point set with 1:1 ratio of red and blue points. Then a fairlet is a pair of one red and one blue point, and a good micro clustering can be found by computing a suitable bipartite matching between the two color classes.

The problem of computing good fairlets gets increasingly difficult when considering more general variants of the problem. For multiple colors and the special case of exact ratio preservation (i.e., for all colors, the allowed range for its ratio is one specific number), the fairlet computation problem can be reduced to a capacitated clustering problem. This is used in [40] to obtain a 14 and 15-approximation for fair k-center and k-supplier with multiple colors and exact ratio preservation.

We give an extensive overview of the existing results and further the fairlet approach in order to explore its applicability for different variants of fair clustering in the Appendix of the full version [13]. Two major issues arise: Firstly, capacitated clustering is not solved for all clustering objectives; indeed, finding a constant-factor approximation for k-median is a long-standing open problem. Secondly, (even for k-center) it is unclear how fairlets even look like when we have multiple colors and want to allow ranges for the ratios. In this situation, subsets of very different size and composition may satisfy the desired ratio.

A different approach is to combine an LP relaxation of the constrained problem with a solution of the unconstrained problem. This approach is not specific for fair clustering; its general idea was for example used by Chakrabarty and Swamy [15] for the minimum latency facility location problem. Finding a reasonably good solution to the unconstrained problem is usually the easiest task with such an approach. Although finding a good formulation of the constrained problem as a linear program can be challenging, the main problem in such approaches is to combine the two solutions into a new solution whose cost can be bound using the quality of the two original solutions. We use such an approach. We start with a set of centers, i.e., a solution to the unconstrained problem. Then we build an LP to find a (fractional) fair solution, and use *weakly supervised LP rounding* to obtain the final integral fair solution. We use this method to prove the following statements.

▶ **Theorem 1.** *There exists a* 5 *and* 7*-approximation for the fair k-center and k-supplier problem which preserves ratios exactly.*

▶ **Theorem 2.** *Given any set of centers S, there exists an assignment ϕ' : which is essentially fair and incurs a cost that is linear in the cost S induces on the unconstrained problem and the cost of an optimal fractional fair clustering of P, for all objectives k-center, k-supplier, k-median, k-means, and facility location.*

▶ **Corollary 3.** *There exists an essentially fair* 3/5/3.488/4.675/62.856*-approximation for the fair k-center/k-supplier/facility location/k-median/k-means problem.*

Here, *essentially fair* refers to our notion of bicriteria approximation: A cluster C is *essentially fair* if there exists a fractional fair cluster C', such that for each color h the number of color h points in C differ by *at most* 1 from the mass of color h points in C'. So this is a small additive fairness violation. After the publication of our results on arXiv (Nov 2018), we have learned that in independent research, Bera et al. [12] find algorithms in a similar model as our essentially fair clustering model and achieve results similar to Corollary 3, for which they provide an almost identical analysis in their arXiv paper (Jan 2019). Theorem 1 is not affected.

We prove Theorem 2 and Corollary 3 in Section 2. Here the unconstrained starting solution can be any solution and we say our algorithm is a *black-box* approximation. We use the given integral solution to guide our rounding of a fractional solution to an LP that incorporates fairness. The proof of Theorem 1 can be found in Section 3. It is more involved as we cannot use a black-box approach, and instead need to find a suitable set of centers (a suitable integral solution) and have to adjust the weakly supervised rounding procedure.

Our results have two advantages. Firstly, we get results for a wide range of clustering problems, and these results improve previous results. For example, we get a 5-approximation for the fair k-center problem with exact ratio preservation, where the best known guarantee was 14. All our bicriteria results work for multiple colors and approximate ratio preservation, a case for which no previous algorithm was known. As for the quality of the guarantees, compare the 4.675-approximation for essentially fair k-median clusterings with the best previously known $\Theta(t)$-approximation, which is only applicable to the case of two colors. Notice that a similar result can *not* be achieved by using bicriteria approximation algorithms for capacitated clustering. The reduction from capacitated clustering only works when the capacities are not violated.

Secondly, the black-box approach has the advantage that fairness can be established belatedly, in a situation where the centers are already given. [21, 44]. Consider our school example and notice that the location of the schools cannot be chosen. Our result says that if we are alright with essentially fair clusterings, we get a clustering which is not much more expensive than a fair clustering where the centers were chosen with the fairness constraint at hand.

Related work

Using k centers to cluster points while minimizing a certain objective function has a long history in terms of results and applications. For the k-center problem in general metric spaces, the 2-approximations developed by Gonzalez [22] and Hochbaum and Shmoys [25] were shown to be tight by Hsu and Nemhauser [26]. The k-supplier problem can be 3-approximated [25], which is also tight. Facility location can be 1.488-approximated [35], which is very close to the known APX-hardness of 1.463 for the problem [23]. For k-median, a recent breakthrough has led to a 2.675-approximation [38, 14], while the best hardness result lies below two [27]. The gap between best upper and lower bound is even larger for k-means, where a 6.357-approximation is the best known [4], and the newest hardness result is marginally above 1 [8, 32].

The k-center problem allows for constant-factor approximations for many useful constraints such as capacity constraints [11, 19, 28], lower bounds on the size of each cluster [3, 6] or allowing for outliers [16, 20]. This is also true for facility location and capacities [2, 7, 10], uniform lower bounds [5, 42], and outliers [16]. Much less is known for k-median and k-means.

True constant-factor approximations so far exist only for the outlier constraint [17, 31]. A major problem for obtaining constant factor approximations is that the natural LP has an unbounded integrality gap, which is also true for the LP with fairness constraints. Bicriteria approximations are known that either violate the capacity constraints [34, 36, 37] or the cardinality constraint [1].

A clustering problem where the points have a color was considered by Li, Yi and Zhang [33]. They provided a 2-approximation for a constraint called *diversity*, which allows at most one point per color in each cluster.

The fairness constraint has been introduced by Chierichetti et al. [18]. They show a 4-approximation for the fair k-center problem with two color classes, where one color class contains t-times as many points as the other, for some integer t. Rösner and Schmidt gave a 14-approximation algorithm for k-center in the extended case with arbitrary many color classes. For the fair k-median problem with two color classes, where one color class contains t-times as many points as the other, for some integer t, Chierichetti et al. [18] also give a $\Theta(t)$-approximation. Backurs et al. [9] give an $O(d \cdot \log(n))$-approximation for a more general version of the fair k-median problem with two color classes, where a problem instance consists of n points in $\mathbb{R}^d$. For k-means the only known approximation algorithm only works for two color classes, which each contain exactly half of the points. Schmidt et al. [41] give a 32.875-approximation for this case. In parallel to our research, Bera et al. [12] have also extended the fairness model to multiple colors and approximate fairness preservation. Their model additionally allows for an overlap of the protected classes. They achieve results similar to Corollary 3.

Recent work of Kleindessner et al. [30] considers the fairness constraints in the context of spectral clustering. Fair data summarization was considered by Kleindessner et al. [29] who imposed the fairness constraint on the cluster centers alone. Specifically, they solve k-center instances with the added constraint that the chosen centers must satisfy an input distribution on the colors (i.e. out of the chosen centers, k_i must belong to color class i, where k_i is given as part of the input). While this formulation is useful for data summarization (when only the centers are reported), it is not guaranteed to lead to fair clusters overall. They propose a 5-approximation algorithm for the case of two color classes. When there are m color classes, they obtain a $(3 \cdot 2^m - 1)$-approximation.

Preliminaries

Points and locations

We are given a set of n points P and a set of potential locations L. We allow L to be infinite (when $L = \mathbb{R}^d$). The task is to open a subset $S \subseteq L$ of the locations and to assign each point in P to an open location via a mapping $\phi : P \to S$. We refer to the set of all points assigned to a location $i \in S$ by $P(i) := \phi^{-1}(i)$. The assignment incurs a cost governed by a semi-metric $d : (P \cup L) \times (P \cup L) \to \mathbb{R}_{\geq 0}$ that fulfills a *β-relaxed triangle inequality*

$$d(x, z) \leq \beta(d(x, y) + d(y, z)) \quad \text{for all } x, y, z \in P \cup L \tag{1}$$

for some $\beta \geq 1$. Additionally, we may have opening costs $f_i \geq 0$ for every potential location $i \in L$ or a maximum number of centers $k \in \mathbb{N}$.

Colors and fairness

We are also given a set of *colors* $Col := \{col_1, \ldots, col_g\}$, and a coloring $col : P \to Col$ that assigns a color to each point $j \in P$. For any set of points $P' \subseteq P$ and any color $col_h \in Col$ we define $col_h(P') = \{j \in P' \mid col(j) = col_h\}$ to be the set of points colored with col_h in P'. We call $r_h(P') := \frac{|col_h(P')|}{|P'|}$ the *ratio* of col_h in P'. If an implicit assignment ϕ is clear from the context, we write $col_h(i)$ to denote the set of all points of a color $col_h \in Col$ assigned to an $i \in S$, i.e., $col_h(i) = col_h(P(i))$.

A set of points $P' \subseteq P$ is *exactly fair* if P' has the same ratio for every color as P, i.e., for each $col_h \in Col$ we have $r_h(P') = r_h(P)$. We say that P' is (ℓ, u)-*fair* or just *fair* for some $\ell = (\ell_1, \ldots, \ell_g)$ and $u = (u_1, \ldots, u_g)$, if we have $r_h(P') \in [\ell_h, u_h]$ for every color $col_h \in Col$.

In our fair clustering problems, we want to preserve the ratios of colors found in P in our clusters. We distinguish two cases: *exact* preservation of ratios, and *relaxed* preservation of ratios. For the exact preservation of ratios, we ask that all clusters are exactly fair, i.e., $P(i)$ is fair for all $i \in S$.

For the relaxed preservation of ratios, we are given the lower and upper bounds $\ell = (\ell_1 = p_1^1/q_1^1, \ldots, \ell_g = p_1^g/q_1^g)$ and $u = (u_1 = p_2^1/q_2^1, \ldots, u_g = p_2^g/q_2^g)$ on the ratio of colors in each cluster and ask that all clusters are (ℓ, u)-*fair*. The exact case is a special case of the relaxed case where we set $\ell_h = u_h = r_h(P)$ for every color $col_h \in Col$.

Essentially fair clusterings are defined below (see Definition 6).

Objectives

We consider fair versions of several classical clustering problems. An instance is given by $I := (P, L, col, d, f, k, \ell, u)$, and our goal is to choose a solution (S, ϕ) according to one of the following objectives.

- ***k*-center** and ***k*-supplier**: minimize the maximum distance between a point and its assigned location: $\min \max_{j \in P} d(j, \phi(j))$. In these problems, we have $f \equiv 0$ and d is a metric. Furthermore, in k-center, $L = P$, whereas in k-supplier , $L \neq P$ is some finite set.
- ***k*-median**: minimize $\sum_{j \in P} d(j, \phi(j))$, d is a metric, $f \equiv 0$ and $L \subseteq P$.
- ***k*-means**: minimize $\sum_{j \in P} d(j, \phi(j))$, where $P \subseteq \mathbb{R}^m$ for some $m \in \mathbb{N}$, $L = \mathbb{R}^m$ and $d(x, y) = ||y - x||^2$ is a semi-metric for $\beta = 2$ and $f \equiv 0$.
- **facility location**: minimize $\sum_{j \in P} d(j, \phi(j)) + \sum_{i \in S} f_i$, where $k = n$, d is a metric and L is a finite set.

The fair assignment problem

For all the objectives above, we call the subproblem of computing a cost-minimal fair assignment of points to given centers the *fair assignment problem*. We show the following theorem in Section A.

▶ **Theorem 4.** *Finding an α-approximation for the fair assignment problem for k-center for $\alpha < 3$ is NP-hard.*

(I)LP formulations for fair clustering problems

Let $I = (P, L, col, d, f, k, \ell, u)$ be a problem instance for a fair clustering problem. We introduce a binary variable $y_i \in \{0, 1\}$ for all $i \in L$ that decides if i is opened, i.e. $y_i = 1 \Leftrightarrow i \in S$. Similarly, we introduce binary variables $x_{ij} \in \{0, 1\}$ for all $i \in L, j \in P$ with $x_{ij} = 1$ if j is assigned to i, i.e. $\phi(j) = i$. All ILP formulations have the inequalities

(2) $\sum_{i \in L} x_{ij} = 1 \ \forall j \in P$ saying that every point j is assigned to a center, the inequalities (3) $x_{ij} \leq y_i \ \forall i \in L, j \in P$ ensuring that if we assign j to i, then i must be open, and the integrality constraints (4) $y_i, x_{ij} \in \{0, 1\} \ \forall i \in L, j \in P$. We may restrict the number of open centers to k with (5) $\sum_{i \in L} y_i \leq k$. For k-center and k-supplier, the objective is commonly encoded in the constraints of the problem, and the (I)LP has no objective function. The idea is to guess the optimum value τ. Since there is only a polynomial number of choices for τ, this is easily done. Given τ, we construct a *threshold graph* $G_\tau = (P \cup L, E_\tau)$ on the points and locations, where a connection between $i \in L$ and $j \in P$ is added iff i and j are close, i.e., $\{i, j\} \in E_\tau \Leftrightarrow d(i, j) \leq \tau$. Then, we ensure that points are not assigned to centers outside their range:

$$x_{ij} = 0 \qquad \text{for all } i \in L, j \in P, \{i, j\} \notin E_\tau \tag{6}$$

For the remaining clustering problems, we pick the adequate objective function from the following three (let $d_{ij} := d(i, j)$):

$$\min \sum_{i \in L, j \in P} x_{ij} d_{ij} \tag{7}$$

$$\min \sum_{i \in L, j \in P} x_{ij} d_{ij}^2 \tag{8}$$

$$\min \sum_{i \in L, j \in P} x_{ij} d_{ij} + \sum_{i \in L} y_i f_i \tag{9}$$

We now have all necessary constraints and objectives. For k-center and k-supplier, we use inequalities (2)-(6), no objective, and define the optimum to be the smallest τ for which the ILP has a solution. We get k-median and k-means by combining inequalities (2)-(5) with (7) and (8), respectively, and we get facility location by combining (2)-(4) with the objective (9). LP relaxations arise from all ILP formulations by replacing (4) by $y_i, x_{ij} \in [0, 1]$ for all $i \in L, j \in P$. To create the fair variants of the ILP formulations, we add fairness constraints modeling the upper and lower bound on the balances.

$$\ell_h \sum_{j \in P} x_{ij} \leq \sum_{col(p_j) = col_h} x_{ij} \leq u_h \sum_{j \in P} x_{ij} \qquad \text{for all } i \in L, h \in Col \tag{10}$$

Although very similar to the canonical clustering LPs, the resulting LPs become much harder to round even for k-center with two colors. We show the following in Section B.

▶ **Lemma 5.** *There is a choice of non-trivial fairness intervals such that the integrality gap of the LP-relaxation of the canonical fair clustering ILP is $\Omega(n)$ for the fair k-center/k-supplier/k-median/facility location problem. The integrality gap is $\Omega(n^2)$ for the fair k-means problem.*

Essential fairness

For a point set P', $\text{mass}_h(P') = |col_h(P')|$ is the *mass* of color col_h in P'. For a possibly fractional LP solution (x, y), we extend this notion to $\text{mass}_h(x, i) := \sum_{j \in col_h(P)} x_{ij}$. We denote the total mass assigned to i in (x, y) by $\text{mass}(x, i) = \sum_{j \in P} x_{ij}$. With this notation, we can now formalize our notion of *essential fairness*.

▶ **Definition 6** (Essential fairness)**.** *Let I be an instance of a fair clustering problem and let (x, y) be an integral, but not necessarily fair solution to I. We say that (x, y) is* essentially fair *if there exists a fractional fair solution (x', y') for I such that $\forall i \in L$:*

$$\lfloor \text{mass}_h(x', i) \rfloor \leq \text{mass}_h(x, i) \leq \lceil \text{mass}_h(x', i) \rceil \qquad \forall col_h \in Col \tag{11}$$

$$\textit{and} \quad \lfloor \text{mass}(x', i) \rfloor \leq \text{mass}(x, i) \leq \lceil \text{mass}(x', i) \rceil. \tag{12}$$

2 Essential fair clusterings via black-box approximation

For essentially fair clustering, we give a powerful framework that employs approximation algorithms for (unfair) clustering problems as a black-box and transforms their output into an essentially fair solution. In this framework, we start by computing an approximate solution for the standard variant of the clustering problem at hand. Next, we solve the LP for the fair variant of the clustering problem. Now we have an integral unfair solution, and a fractional fair solution. Our final and most important step is to combine these two solutions into an integral and essentially fair solution. It consists of two conceptual sub-steps: Firstly, we show that it is possible to find a fractional fair assignment to the centers of the integral solution that is sufficiently cheap. Secondly, we round the assignment. This last sub-step introduces the potential fairness violation of one point per color per cluster.

We show that this approach yields constant-factor approximations with fairness violation for all mentioned clustering objectives. The description will be neutral whenever the objective does not matter. Thus, descriptions like *the LP* mean the appropriate LP for the desired clustering problem. When the problem gets relevant, we will specifically discuss the distinctions. Notice that for all clustering problems defined in Section 1, P and L are finite except for k-means. However, for the k-means problem, we can assume that $L = P$ if we accept an additional factor of 2 in the approximation guarantee. Thus, we assume in the following that L and P are finite sets. Indeed, we even assume at least $L \subseteq P$ for all problems except k-supplier and facility location.

2.1 Step 1: Obtaining a fair solution with integral y

In the first step, we assume that we are given two solutions. Let (x^{LP}, y^{LP}) be an optimal solution to the LP. This solution has the property that the assignments to all centers are fair, however, the centers may be fractionally open and the points may be fractionally assigned to several centers. Let c^{LP} be the objective value of this solution. For k-supplier and k-center, it is the smallest τ for which the LP is feasible, for the other objectives, it is the value of the LP. We denote the cost of the best *integral* solution to the LP by c^*. We know that $c^{LP} \leq c^*$.

Let $(\bar{x}, \bar{y})$ be any integral solution to the LP that may violate fairness, i.e., inequality (10), and let $\bar{c}$ be the objective value of this solution. We think of $(\bar{x}, \bar{y})$ as being a solution of an α-approximation algorithm for the standard (unfair) clustering problem for some constant α. Since the unconstrained version can only have a lower optimum cost, we then have $\bar{c} \leq \alpha \cdot c^*$.

Our goal is now to combine (x^{LP}, y^{LP}) and $(\bar{x}, \bar{y})$ into a third solution, $(\hat{x}, \hat{y})$, such that the cost of $(\hat{x}, \hat{y})$ is bounded by $O(c^{LP} + \bar{c}) \subseteq O(c^*)$. Furthermore, the entries of $\hat{y}$ shall be integral. The entries of $\hat{x}$ may still be fractional after step 1.

Let S be the set of centers that are open in $(\bar{x}, \bar{y})$. For all $j \in P$, we use $\bar{\phi}(j)$ to denote the center in S closest to j, i.e., $\bar{\phi}(j) = \arg\min_{i \in S} d(j, i)$ (ties broken arbitrarily). Notice that the objective value of using S with assignment $\bar{\phi}$ for all points in P is at most $\bar{c}$, since assigning to the closest center is always optimal for the standard clustering problems without fairness constraint.

Depending on the objective, L is a subset of P or not, i.e., $\bar{\phi}$ is not necessarily defined for all locations in L. We then extend $\bar{\phi}$ in the following way. Let $i \in L \setminus P$ be any center, and let j^* be the closest point to it in P. Then we set $\bar{\phi}(i) := \bar{\phi}(j^*)$, i.e., i is assigned to the center in S which is closest to the point in P which is closest to i. Finally, let $\bar{C}(i) = \bar{\phi}^{-1}(i)$ be the set of all points and centers assigned to i by $\bar{\phi}$. We show the following lemma.

▶ **Lemma 7.** *Let (x^{LP}, y^{LP}) and $(\bar{x}, \bar{y})$ be two solutions to the LP, where $(\bar{x}, \bar{y})$ may violate inequality (10), but is integral. Then the solution defined by $\hat{y} := \bar{y}$ and*

$$\hat{x}_{ij} := \sum_{i' \in \bar{C}(i)} x^{LP}_{i'j} \quad \text{for all } i \in S, j \in P, \qquad \hat{x}_{ij} := 0 \quad \text{for all } i \notin S, j \in P.$$

satisfies inequality (10), $\hat{y}$ is integral, and the cost $\hat{c}$ of $(\hat{x}, \hat{y})$ is bounded by $c^{LP} + \bar{c}$ for k-center, by $2 \cdot c^{LP} + \bar{c}$ for k-supplier, k-median, and facility location, and by $12 \cdot c^{LP} + 8 \cdot \bar{c}$ for k-means.

Proof. Recall that for k-center and k-supplier, speaking of the cost of an LP solution is a bit sloppy; we mean that $(\hat{x}, \hat{y})$ is a feasible solution in the LP with threshold $\hat{c}$.

The definition of $(\hat{x}, \hat{y})$ means the following. For every (fractional) assignment from a point j to a center i', we look at the cluster with center $i = \bar{\phi}(i')$ to which i' is assigned to by $\bar{\phi}$. We then transfer this assignment to i. So from the perspective of i, we collect all fractional assignments to centers in $\bar{C}(i)$ and consolidate them at i. Notice that the (fractional) number of points assigned to i after this process may be less than one since $(\bar{x}, \bar{y})$ may include centers that are very close together.

Since that $\hat{y}$ is simply $\bar{y}$ it is integral as well and has the same number of centers, thus $\hat{y}$ also satisfies (5) if the problem uses it. Next, we observe that $(\hat{x}, \hat{y})$ satisfies fairness, i.e., respects (10). This is true because (x^{LP}, y^{LP}) satisfies them, and because we move *all* assignment from a center i' to the same center $\bar{\phi}(i')$. This transferring operation preserves the fairness. Inequality (3) is true because we only move assignments to centers that are fully open in $(\bar{x}, \bar{y})$, i.e., the inequality cannot be violated as long as (2) is true (which it is for (x^{LP}, y^{LP}) since it is a feasible LP solution). Equality (2) is true for $(\hat{x}, \hat{y})$ since all assignment of j is moved to some fully open center. Thus $(\hat{x}, \hat{y})$ is a feasible solution for the LP. It remains to show that $\hat{c}$ is small enough, which depends on the objective.

***k*-median and *k*-means.** We start by showing this for k-median (where the distances are a metric, i.e., $\beta = 1$ in the β-triangle inequality (1)) and k-means (where the distances are a semi-metric with $\beta = 2$). We observe that here, the cost of $(\hat{x}, \hat{y})$ is

$$\hat{c} = \sum_{j \in P} \sum_{i \in L} \hat{x}_{ij} d(i,j) = \sum_{j \in P} \sum_{i \in L} \sum_{i' \in \bar{C}(i)} x^{LP}_{i'j} d(i,j).$$

Now fix $i \in L$, $i' \in \bar{C}(i)$ and $j \in P$ arbitrarily. By the β-relaxed triangle inequality, $d(i,j) \leq \beta \cdot d(i',j) + \beta \cdot d(i',i)$. Furthermore, we know that $i' \in \bar{C}(i)$, i.e., $\bar{\phi}(i') = i$ and $d(i',i) \leq d(i', \bar{\phi}(j))$. We can use this to relate $d(i',i)$ to the cost that j pays in $(\bar{x}, \bar{y})$:

$$d(i',i) \leq d(i', \bar{\phi}(j)) \leq \beta \cdot d(j,i') + \beta \cdot d(j, \bar{\phi}(j)).$$

Adding this up yields

$$\begin{aligned}
&\sum_{j \in P} \sum_{i \in L} \sum_{i' \in \bar{C}(i)} x^{LP}_{i'j} d(i,j) \\
\leq &\sum_{j \in P} \sum_{i \in L} \sum_{i' \in \bar{C}(i)} (\beta + \beta^2) x^{LP}_{i'j} d(i',j) + \sum_{j \in P} \sum_{i \in L} \sum_{i' \in \bar{C}(i)} \beta^2 \cdot x^{LP}_{i'j} d(j, \bar{\phi}(j)) \\
= &(\beta + \beta^2) \cdot c^{LP} + \beta^2 \cdot \bar{c}.
\end{aligned}$$

For $\beta = 1$ (k-median), this is $2c^{LP} + \bar{c}$, for $\beta = 2$ (k-means), we get $12c^{LP} + 8\bar{c}$

Facility location. For facility location, we have to include the facility opening costs. We

open the facilities that are open in $(\bar{x}, \bar{y})$, which incurs a cost of $\sum_{i \in L} \bar{y}_i f_i$. The distance costs are the same as for k-median, so we get a total cost of

$$\sum_{j \in P} \sum_{i \in L} \sum_{i' \in \bar{C}(i)} 2x^{LP}_{i'j} d(i', j) + \sum_{j \in P} \sum_{i \in L} \sum_{i' \in \bar{C}(i)} x^{LP}_{i'j} d(j, \bar{\phi}(j)) + \sum_{i \in L} \bar{y}_i f_i \leq 2c^{LP} + \bar{c}.$$

***k*-center and *k*-supplier.** For the k-center and k-supplier proof, we again fix $i \in L$, $i' \in \bar{C}(i)$ and $j \in P$ arbitrarily and use that $d(i, j) \leq d(i, i') + d(i', j)$. Now for k-center, we know that $d(i, i') \leq \bar{c}$ since $i' \in \bar{C}(i)$, and we know that $d(i', j) \leq c^{LP}$ for all j where x^{LP}_{ij} is strictly positive. Thus, if $\hat{x}_{ij}$ is strictly positive, then $d(i, j) \leq \bar{c} + c^{LP}$. For k-supplier, we have no guarantee that $d(i, i') \leq \bar{c}$ since i' is not necessarily an input point. Instead, $i' \in \bar{C}(i)$ means that the point j' in P which is closest to i' is assigned to i by $\bar{x}$. Since j' is the closest to i' in P, we have $d(i', j') \leq d(i', j)$. Furthermore, since $j' \in \bar{C}(i)$, $d(i, j') \leq \bar{c}$. Thus, we get for k-supplier that

$$d(i, j) \leq d(i, i') + d(i', j) \leq d(i, j') + d(i', j') + d(i', j) \leq \bar{c} + 2 \cdot c^{LP}. \quad \blacktriangleleft$$

2.2 Step 2: Rounding the x-variables

For rounding the x-variables, we need to distinguish between two cases of objectives. Let $j \in P$ be a point that is fractionally assigned to some centers $L_j \subseteq L$.

First, we have objectives where we can transfer mass from an assignment of j to $i' \in L_j$ to an assignment of j to $i'' \in L_j$ without modifying the objective. We say that such objectives are *reassignable* (in the sense that we can reassign j to centers in L_j without changing the cost). k-center and k-supplier have this property.

Second, we have objectives where the assignment cost is separable, i.e., where the distances influence the cost via a term of the form $\sum_{i \in L, j \in P} c_{ij} \cdot x_{ij}$ for some $c_{ij} \in \mathbb{R}_{\geq 0}$. We call such objectives *separable*. Facility location, k-median and k-means fall into the this category.

▶ **Lemma 8.** *Let (x, y) be an α-approximate fractional solution for a fair clustering problem with the property that all $y_i, i \in L$ are integral. Then we can obtain an α-approximative integral solution (x', y') with an additive fairness violation of at most one in time $O(\text{poly}(|S| + |P|))$, with $S := \{i \in L \mid y_i \geq 1\}$ being the set of locations that are opened in (x, y).*

Proof. We create our rounded α-approximate integral solution (x', y') by min-cost flow computations. We begin by constructing a min-cost flow instance which depends on our starting solution (x, y) as well as on the objective of the problem we are studying.

We define a min-cost flow instance $(G = (V, A), c, b)$ (also see Figure 1) with unit capacities and costs c on the edges as well as balances b on the nodes. We begin by defining a graph $G^h = (V^h, A^h)$ for every color $h \in Col$ with

$$V^h := V^h_S \cup V^h_P, \quad V^h_S := \left\{v^h_i \mid i \in S\right\}, \quad V^h_P := \left\{v^h_j \mid j \in col_h(P)\right\},$$
$$A^h := \left\{(v^h_j, v^h_i) \mid i \in S, j \in col_h(P) : x_{ij} > 0\right\},$$

as well as costs c^h by $c^h_a := c_{ij}$ for $a = (v^h_j, v^h_i) \in A^h, i \in S, j \in col_h(P)$ and balances b^h by $b^h_v := 1$ if $v \in V^h_P$ and $b^h_v := -\lfloor \text{mass}_h(x, i) \rfloor$ if $v = v^h_i \in V^h_S$. We use the graphs G_h to define $G = (V, A)$ by

$$V := \{t\} \cup V_S \cup \bigcup_{h \in Col} V^h, \quad V_S := \{v_i \mid i \in S\}$$
$$A := \bigcup_{h \in Col} A^h \cup \left\{(v^h_i, v_i) \mid i \in S, h \in Col : \text{mass}_h(x, i) - \lfloor \text{mass}_h(x, i) \rfloor > 0\right\}$$
$$\cup \left\{(v_i, t) \mid i \in S : \text{mass}(x, i) - \lfloor \text{mass}(x, i) \rfloor > 0\right\},$$

together with costs c of $c_a := c^h_a$ for $a \in A^h$ and 0 otherwise, and balances b of $b_v := b^h_v$ if $v \in V^h$ for some $h \in Col$, $b_v := -B_i$ if $v = v_i \in V_S$ and $b_t := -B$ with $B_i = \lfloor \text{mass}(x,i) \rfloor - \sum_{h \in Col} \lfloor \text{mass}_h(x,i) \rfloor$ and $B := |P| - \sum_{i \in S} \lfloor \text{mass}(x,i) \rfloor$.

Separable objectives – k-median and k-means

We observe that:

1. B and B_i are integers for all $i \in S$, and so are all capacities, costs and balances. Consequently, there are integral optimal solutions for the min-cost flow instance (G, c, b),
2. (x, y) induces a feasible solution for (G, c, b), by defining a flow x in G as follows:

$$x_a := \begin{cases} x_{ij} & \text{if } a = (v^h_j, v^h_i) \in A^h, j \in P, i \in S, \\ \text{mass}_h(x,i) - \lfloor \text{mass}_h(x,i) \rfloor & \text{if } a = (v^h_i, v_i) \in A, h \in Col, i \in S, \\ \text{mass}(x,i) - \lfloor \text{mass}(x,i) \rfloor & \text{if } a = (v_i, t) \in A, i \in S. \end{cases}$$

   Since (x, y) is a fractional solution, x satisfies capacity and non-negativity constraints because $x_{ij} \in [0,1]$ for all $i \in L, j \in P$ and $\text{mass}_h(x,i) - \lfloor \text{mass}_h(x,i) \rfloor, \text{mass}(x,i) - \lfloor \text{mass}(x,i) \rfloor \in [0,1]$ for all $i \in S$ and $col_h \in Col$ as well. We have flow conservation since the fractional solution needs to assign all points, and the flow of the edges (v^h_i, v_i) and (v_i, t) as well as the demand of v_i and t are chosen in such a way that we have flow conservation for all the other nodes as well.
3. Integral solutions x to the min-cost flow instance (G, c, b) induce an integral solution $(\bar{x}, y)$ to the original clustering problem by setting $\bar{x}_{ij} := x_a$ for $a = (v^h_j, v^h_i) \in A^h$ if $j \in col_h(P), i \in S$. Since the flow x is integral, this gives us an integral assignment of all points to centers which have been opened, since y was already integral before this step. This incurs the additive fairness violation of at most one, since every $i \in S$ is guaranteed by our balances to have at least $\lfloor \text{mass}_h(x,i) \rfloor$ points of color $h \in Col$ and at least $\lfloor \text{mass}(x,i) \rfloor$ points in total assigned to it. Since there is at most one outgoing arc of unit capacity (v^h_i, v_i) and (v_i, t) for an $i \in S$ if $\text{mass}_h(x,i) - \lfloor \text{mass}_h(x,i) \rfloor > 0$, we have at most $\lceil \text{mass}_h(x,i) \rceil$ points of color col_h and $\lceil \text{mass}(x,i) \rceil$ total points assigned to i.

Together, this yields that computing a min-cost flow $\hat{x}$ for (G, c, b) followed by applying the third observation to $\hat{x}$ yields a solution $(\hat{x}, y)$ to the clustering with an additive fairness violation of at most one.

Since (x, y) was inducing the fractional solution x with $\text{cost}(x) = \text{cost}(x, y)$ to the min-cost flow instances, and $\text{cost}(x) \geq \text{cost}(\hat{x})$ by construction we have $\text{cost}(\hat{x}, y) \leq \text{cost}(x, y)$.

Reassignable objectives – k-center and k-supplier

In the case of reassignable objectives, we do not have to care about costs, as long as the reassignments happen to centers in L_j for all points $j \in P$. We essentially use the same strategy as before, but instead of a min cost flow problem we solve the transshipment problem $(G = (V, A), b)$ with unit capacities on the edges and balances b on the nodes. Notice that the three observations from the previous case apply here as well, and reassignability guarantees that the cost does not increase. ◀

Lemmas 7 and 8 then lead directly to Theorem 2, or, in more detail, to:

▶ **Theorem 9.** *Black-box approximation for fair clustering gives essentially fair solutions with a cost of $c^{LP} + \bar{c}$ for k-center, $2c^{LP} + \bar{c}$ for k-supplier, k-median and facility location, and $12c^{LP} + 8\bar{c}$ for k-means where c^{LP} is the cost of an optimal solution to the fair LP relaxation and $\bar{c}$ is the cost of the given solution.*

We know that c^{LP} is not more expensive than an optimal solution to the fair clustering problem. If we use an α-approximation to obtain the unfair clustering solution, we have that $\bar{c}$ is at most α times the cost of an optimal solution to the fair clustering problem. Currently, the best known approximation factors are 2 for k-center [22, 25], 3 for k-supplier [25], 1.488 for facility location [35], 2.675 for k-median [14, 38] and 6.357 for k-means [4], which yields Corollary 3.

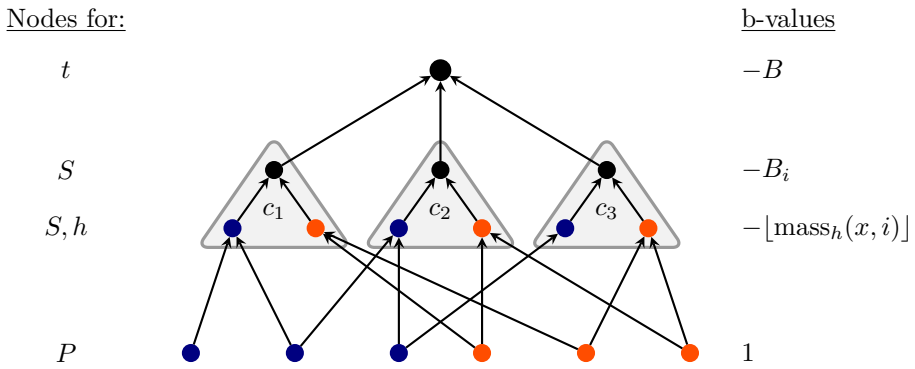


Figure 1 Example for the graph G used in the rounding of the x-variables. $B_i = \lfloor \text{mass}(x,i) \rfloor - \sum_{h \in Col} \lfloor \text{mass}_h(x,i) \rfloor$ and $B = |P| - \sum_{i \in S} \lfloor \text{mass}(x,i) \rfloor$.

3 True approximations for fair k-center and k-supplier

We now extend our weakly supervised rounding technique for k-center and k-supplier in the case of the exact fairness model. We replace the black-box algorithm with a specific approximation algorithm, and then achieve true approximations for the fair clustering problems by informed rounding of the LP solution.

3.1 5-Approximation Algorithm for k-center

In this section, we consider the fair k-center problem with exact preservation of ratios and without any additive fairness violation.

We give a 5-approximation for this variant. The algorithm begins by choosing a set of centers. In contrast to Section 2 we do not use an arbitrary algorithm for the standard k-center problem but specifically look for nodes in the threshold graph $G_\tau = (P, E_\tau)$ where $E_\tau = \{(i,j) \mid i \neq j \in P, d(i,j) \leq \tau\}$ that form a maximal independent set S in G_τ^2. Here G_τ^t denotes the graph on P that connects all pairs of nodes which are connected by a path of length at most t in G_τ and we denote the edge set of G_τ^t by E_τ^t. As we use the following procedure independent for each connected component of G_τ, we will in the description and the following proofs of the procedure assume that G_τ is a connected graph. The procedure uses the approach by Khuller and Sussmann [28] (procedure AssignMonarchs) to find S which ensures the following property: There exists a tree T spanning all the nodes in S and two adjacent nodes in T are exactly distance 3 apart in G_τ. The procedure begins by choosing an arbitrary vertex $r \in P$, called *root*, into S and marking every node within distance 2 of r (including itself). Until all the nodes in P are marked, it chooses an unmarked node u that is adjacent to a marked node v and marks all nodes in the distance two neighborhood of u. Observe that u is exactly at distance 3 from a node $u' \in S$ chosen earlier that caused v to get marked. Thus the run of the procedure implicitly defines the tree T over the nodes of

S. In case G_τ is not a connected graph this procedure is run on each connected component and the set S has the following property: There exists a forest F such that F reduced to a connected component of G_τ is a tree T spanning all the nodes of S inside of that connected component and two adjacent nodes in T are exactly distance 3 apart in G_τ.

In the next phase, we make use of some structure that feasible solutions with exact preservation of the ratios must have.

▶ **Observation 10.** *Let $m \in \mathbb{N}$ be the smallest integer such that for each color $h \in Col$ we have $r_h(P) = \frac{q_h}{m}$ for some $q_h \in \mathbb{N}$. Then for each cluster $P(i)$ in a fair clustering $\mathcal{C}$ of P with exact preservation of ratios, there exists a positive integer $i' \in \mathbb{N}_{\geq 1}$ such that $P(i)$ contains exactly $i' \cdot q_h$ points with color h for each color $h \in Col$ and $i' \cdot m$ total points. Thus every cluster must have at least q_h points of color h for each color $h \in Col$.*

We use Observation 10 and the fixed set of centers S to obtain the following adjusted LP for the fractional fair k-center problem.

$$\sum_{i \in S} x_{ij} = 1, \qquad \forall j \in P \tag{13}$$

$$\sum_{j \in col_h(P)} x_{ij} = r_h(P) \sum_{j \in P} x_{ij} \qquad \forall i \in S \tag{14}$$

$$\sum_{\substack{j \in col_h(P) \\ (i,j) \in E_\tau^2}} x_{ij} \geq q_h \qquad \forall i \in S, \forall h \in Col \tag{15}$$

$$x_{ij} = 0 \qquad \forall i \in S, j \in P \text{ with } (i,j) \notin E_\tau^3 \tag{16}$$

$$0 \leq x_{ij} \leq 1 \qquad \forall i \in S, j \in P \tag{17}$$

Here inequality (15) ensures that each cluster contains at least q_h points of color h. Let S_{opt} be the set of centers in the optimal solution and let $\phi_{opt} : P \to S_{opt}$ be the optimal fair assignment. For the correct guess τ, every center $i \in S$ has a distinct center in S_{opt} which is at most distance one away from i in G_τ. Therefore, there exists q_h points of each color h within distance two of i. This ensures that inequality (15) is satisfiable for the right guess τ. And since, every center in S_{opt} is within distance two of some $i \in S$, there exists a fair assignment of points in P to centers in S within distance three. Thus the above LP is feasible for the right τ.

Now for the final phase, the algorithm rounds a fractional solution for the above assignment LP to an integral solution of cost at most 5τ in a procedure motivated by the LP rounding approach used by Cygan et al. in [19] for the capacitated k-center problem. Let $\beta(i)$ denote the children of node $i \in S$ in the tree T. Starting from the leaf nodes we recursively define quantities $\Gamma(i)$ and $\delta(i)$, $\forall i \in S$ as follows:

$$\Gamma(i) = \left\lfloor \frac{\sum_{j \in col_1(P)} x_{ij} + \sum_{i' \in \beta(i)} \delta(i')}{q_1} \right\rfloor q_1$$

$$\delta(i) = \sum_{j \in col_1(P)} x_{ij} + \sum_{i' \in \beta(i)} \delta(i') - \Gamma(i)$$

For a leaf node i in the tree T we have $\beta(i) = \emptyset$, then $\Gamma(i)$ denotes the amount of color 1 points assigned to i rounded down to the nearest multiple of q_1, while $\delta(i)$ denotes the remaining amount. The idea is to reassign the remainder to the parent of i. Then for a non leaf i' $\Gamma(i')$ denotes the amount of color 1 points assigned to i' plus the remainder that all children of i' want to reassign to i' rounded down to the nearest multiple of q_1, while

$\delta(i')$ again denotes the remainder. Since by definition of q_1 the total number of points in $col_1(P)$ must be an integer multiple of q_1, $\Gamma(r)$ also denotes the the amount of color 1 points assigned to r plus the remainder that all children of r want to reassign to r and $\delta(r) = 0$.

Also note that $\Gamma(i)$ is always a positive integer multiple of q_1 for any i, and $\delta(i)$ is always non-negative and less than q_1.

One can think of the x_{ij} variables as encoding flow from a vertex j to a node $i \in S$. We call it a color h flow if j has color h. We will re-route these flows (maintaining the ratio constraints) such that $\forall i \in S$, $j \in col_1(P)$ x_{ij} is equal to $\Gamma(i)$ which is an integral multiple of q_1.

▶ **Lemma 11.** *There exists an integral assignment of all vertices with color* 1 *to centers in* S *in* G^5_τ *that assigns* $\Gamma(i)$ *vertices with color* 1 *to each center* $i \in S$*.*

Proof. Construct the following flow network: Take sets $col_1(P)$ and S to form a bipartite graph with an edge of capacity one between a vertex $j \in col_1(P)$ and a center $i \in S$ if and only if $(i, j) \in E^5_\tau$. Connect a source s with unit capacity edges to all vertices in $col_1(P)$ and each center $i \in S$ with capacity $\Gamma(i)$ to a sink t. We now show a feasible fractional flow of value $|col_1(P)|$ in this network. For each leaf node i in T which is not the root, assign $\Gamma(i)$ amount of color 1 flow from the total incoming color 1 flow $\sum_{j \in col_1(P)} x_{ij}$ from vertices that are at most distance three away from i in G_τ and propagate the remaining $\delta(i)$ amount of color 1 flow, coming from distance two vertices, upwards to be assigned to the parent of node i. This is always possible because by definition $\delta(i) < q_1$ and constraint (15) ensures that every center has at least q_1 amount of color 1 flow coming from distance two vertices. For every non-leaf node i, assign $\Gamma(i)$ amount of incoming color 1 flow from distance five vertices (including the color 1 flows propagated upwards by its children) and propagate $\delta(i)$ amount of color 1 flow from distance two vertices (possible due to constraint (15)). Thus every center has $\Gamma(i)$ amount of color 1 flow passing through it and it is easy to verify that the value of the total flow in the network is $|col_1(P)|$. Since the network only has integral capacities, there exists an integral max-flow of value $|col_1(P)|$. ◀

▶ **Lemma 12.** *For any reassignment of a color* 1 *flow, there exists a reassignment of color* h*-flow between the same centers for all* $h \in Col \setminus \{1\}$*, such that the resulting fractional assignment of the vertices satisfies the fairness constraints at each center.*

Proof. Say f_1 amount of color 1 flow is reassigned from center i_1 to another center i_2. Reassign $f_h = r_h \cdot f_1 / r_1$ amount of color h flow from i_1 to i_2 for each color $h \in Col \setminus \{1\}$. This is possible as constraint (14) implies that the amount of color h points assigned to i_1 must be equal to $\frac{r_h}{r_1}$ times the amount of color 1 points assigned to i_1 and f_1 must be less than the amount of color 1 points assigned to i_1. It is easy to verify that the ratios at i_1 and i_2 remain unchanged as by construction the ratio of the reassigned flows is equal to the original ratio. ◀

From Lemmas 11 and 12 we can say that there is a fair fractional assignment within distance 5τ such that all the color 1 assignments are integral and every center i has $\Gamma(i)$ color 1 vertices assigned to it. Since this assignment is fair the total incoming color h flow at each center must be $\Gamma(i)\frac{q_h}{q_1}$ which are integers for every center $i \in S$ and every color $h \in Col$.

▶ **Lemma 13.** *There exists an integral fair assignment in* G^5_τ*.*

Proof. Construct a flow network for color h vertices similar to the one in lemma 11: Take sets $col_h(P)$ and S to form a bipartite graph with an edge of capacity one between a vertex $j \in col_h(P)$ and a center $i \in S$ if and only if $(i, j) \in E^5_\tau$. Connect a source s with unit

capacity edges to all vertices in $col_h(P)$ and each center $i \in S$ with capacity $\Gamma(i)\frac{r_h}{r_1}$ to a sink t. The above fractional assignment in G_τ^5 gives a flow for the above network. Since the network only consists of integral demands and capacities, there is an integral max-flow which gives the assignment for the color h vertices. ◀

▶ **Theorem 14.** *There exists a 5-approximation for the fair k-center problem with exact preservation of ratios.*

Proof. Follows from Lemmas 11, 12 and 13 ◀

3.2 7-approximation for k-suppliers

We adapt the algorithm in Section 3.1 to work for the k-suppliers model to give a 7-approximation for the variant with exact preservation of ratios. In the k-suppliers model, we are not allowed to open centers anywhere in P. Instead, we are provided a set L of potential locations to open centers. The procedure closely resembles the k-center algorithm: construct a bipartite threshold graph $G_\tau = (P \cup L, E_\tau)$ where $E_\tau = \{(i,j) \mid i \in L, j \in P, d(i,j) \leq \tau\}$. Choose a *root* vertex $r \in P$ into S and mark all vertices in P that are within distance two. Until all vertices in P are marked, choose an unmarked vertex $u \in P$ that is distance two away from a marked vertex and mark all vertices in the distance two neighborhood of u. Note that, since G_τ is bipartite, no two vertices in P are adjacent. The vertex u is exactly at distance four from a vertex $u' \in S$ chosen earlier. This process of selecting vertices in S defines a tree T over them with the property that adjacent vertices in T are exactly at distance four of each other in G_τ. Since we apply the procedure separately for each of the connected components of the threshold graph, we may safely assume that G_τ is connected.

Let us now temporarily open one center at each vertex in S and make the following observations for the k-suppliers case:

1. Observation 10 still holds.
2. The corresponding LP is the same as the k-center LP, except it has E_τ^4 in place of E_τ^3 in constraint (16). This ensures the feasibility of the LP since every location in L is at most distance three away from some vertex in S. (Note that in case G_τ is not connected, it can happen that some locations in L are not connected to any point and therefore more than distance three away from some vertex in S, but since they are not connected to any point we can safely ignore them, as they cannot be part of the optimal solution.)
3. Lemma 11 with G_τ^6 instead of G_τ^5 holds. The extra distance of one is introduced because the distance between a child vertex and its parent vertex in T is four instead of three.
4. Lemma 12 holds as it is and Lemma 13 holds when G_τ^5 is replaced with G_τ^6.

Thus we have a distance six fair assignment to centers in S. However, this is not a valid solution for k-suppliers as $S \subseteq P$ and we are allowed to open centers only in L. So, we move each of these temporary centers to a neighboring location in L to obtain a distance seven assignment.

References

1 Karen Aardal, Pieter L. van den Berg, Dion Gijswijt, and Shanfei Li. Approximation algorithms for hard capacitated k-facility location problems. *European Journal of Operational Research*, 242:358–368, 2015. doi:10.1016/j.ejor.2014.10.011.

2 Ankit Aggarwal, Anand Louis, Manisha Bansal, Naveen Garg, Neelima Gupta, Shubham Gupta, and Surabhi Jain. A 3-approximation algorithm for the facility location problem with uniform capacities. *Mathematical Programming*, 141:527–547, 2013. doi:10.1007/s10107-012-0565-4.

3 Gagan Aggarwal, Rina Panigrahy, Tomás Feder, Dilys Thomas, Krishnaram Kenthapadi, Samir Khuller, and An Zhu. Achieving anonymity via clustering. *ACM Transactions on Algorithms*, 6:49:1–49:19, 2010. doi:10.1145/1798596.1798602.

4 Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better Guarantees for k-Means and Euclidean k-Median by Primal-Dual Algorithms. In Chris Umans, editor, *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS 2017)*, pages 61–72. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.15.

5 Sara Ahmadian and Chaitanya Swamy. Improved Approximation Guarantees for Lower-Bounded Facility Location. In Thomas Erlebach and Giuseppe Persiano, editors, *10th International Workshop on Approximation and Online Algorithms (WAOA 2012)*, volume 7846 of *Lecture Notes in Computer Science (LNCS)*, pages 257–271. Springer Berlin Heidelberg, 2012. doi:10.1007/978-3-642-38016-7_21.

6 Sara Ahmadian and Chaitanya Swamy. Approximation Algorithms for Clustering Problems with Lower Bounds and Outliers. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 69:1–69:15. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.69.

7 Hyung-Chan An, Mohit Singh, and Ola Svensson. LP-based algorithms for capacitated facility location. *SIAM Journal on Computing*, 46:272–306, 2017. doi:10.1137/151002320.

8 Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The Hardness of Approximation of Euclidean k-Means. In Lars Arge and János Pach, editors, *31st International Symposium on Computational Geometry (SoCG 2015)*, volume 34 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 754–767. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.SOCG.2015.754.

9 Arturs Backurs, Piotr Indyk, Krzysztof Onak, Baruch Schieber, Ali Vakilian, and Tal Wagner. Scalable Fair Clustering. *CoRR*, abs/1902.03519, 2019. arXiv:1902.03519.

10 Manisha Bansal, Naveen Garg, and Neelima Gupta. A 5-Approximation for Capacitated Facility Location. In Leah Epstein and Paolo Ferragina, editors, *Algorithms – ESA 2012*, volume 7501 of *Lecture Notes in Computer Science (LNCS)*, pages 133–144. Springer Berlin Heidelberg, 2012. doi:10.1007/978-3-642-33090-2_13.

11 Judit Bar-Ilan, Guy Kortsarz, and David Peleg. How to Allocate Network Centers. *Journal of Algorithms*, 15:385–415, 1993. doi:10.1006/jagm.1993.1047.

12 Suman K. Bera, Deeparnab Chakrabarty, and Maryam Negahbani. Fair Algorithms for Clustering. *CoRR*, abs/1901.02393, 2019. arXiv:1901.02393.

13 Ioana O. Bercea, Martin Groß, Samir Khuller, Aounon Kumar, Clemens Rösner, Daniel R. Schmidt, and Melanie Schmidt. On the cost of essentially fair clusterings. *CoRR*, abs/1811.10319, 2018. arXiv:1811.10319.

14 Jarosław Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An Improved Approximation for k-Median and Positive Correlation in Budgeted Optimization. *ACM Transactions on Algorithms*, 13:23:1–23:31, 2017. doi:10.1145/2981561.

15 Deeparnab Chakrabarty and Chaitanya Swamy. Facility location with client latencies: LP-based techniques for minimum-latency problems. *Math. Oper. Res.*, 41(3):865–883, 2016. doi:10.1287/moor.2015.0758.

16 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In S. Rao Kosaraju, editor, *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms (SODA 2001)*, pages 642–651, 2001. URL: http://dl.acm.org/citation.cfm?id=365411.365555.

17 Ke Chen. A constant factor approximation algorithm for k-median clustering with outliers. In Shang-Hua Teng, editor, *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2008)*, pages 826–835. SIAM, 2008. URL: http://dl.acm.org/citation.cfm?id=1347082.1347173.

18 Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair Clustering Through Fairlets. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017 (NIPS 2017)*, pages 5036–5044, 2017. URL: `http://papers.nips.cc/paper/7088-fair-clustering-through-fairlets`.

19 Marek Cygan, MohammadTaghi Hajiaghayi, and Samir Khuller. LP rounding for k-centers with non-uniform hard capacities. In Venkatesan Guruswami, editor, *53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2012)*, pages 273–282. IEEE Computer Society, 2012. `doi:10.1109/FOCS.2012.63`.

20 Marek Cygan and Tomasz Kociumaka. Constant Factor Approximation for Capacitated k-Center with Outliers. In Ernst W. Mayr and Natacha Portier, editors, *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 251–262. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014. `doi:10.4230/LIPIcs.STACS.2014.251`.

21 Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012 (ITCS 2012)*, pages 214–226. ACM, 2012. `doi:10.1145/2090236.2090255`.

22 Teofilo F. Gonzalez. Clustering to Minimize the Maximum Intercluster Distance. *Theoretical Computer Science*, 38:293–306, 1985. `doi:10.1016/0304-3975(85)90224-5`.

23 Sudipto Guha and Samir Khuller. Greedy Strikes Back: Improved Facility Location Algorithms. *Journal of Algorithms*, 31:228–248, 1999. `doi:10.1006/jagm.1998.0993`.

24 Moritz Hardt, Eric Price, and Nati Srebro. Equality of Opportunity in Supervised Learning. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, pages 3315–3323, 2016.

25 Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *Journal of the ACM*, 33(3):533–550, 1986. `doi:10.1145/5925.5933`.

26 Wen-Lian Hsu and George L. Nemhauser. Easy and hard bottleneck location problems. *Discrete Applied Mathematics*, 1:209–215, 1979. `doi:10.1016/0166-218X(79)90044-1`.

27 Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing (STOC 2002)*, pages 731–740. ACM, 2002. `doi:10.1145/509907.510012`.

28 Samir Khuller and Yoram J. Sussmann. The Capacitated K-Center Problem. *SIAM Journal on Discrete Mathematics*, 13:403–418, 2000. `doi:10.1137/S0895480197329776`.

29 Matthäus Kleindessner, Pranjal Awasthi, and Jamie Morgenstern. Fair k-Center Clustering for Data Summarization. *CoRR*, abs/1901.08628, 2019. `arXiv:1901.08628`.

30 Matthäus Kleindessner, Samira Samadi, Pranjal Awasthi, and Jamie Morgenstern. Guarantees for Spectral Clustering with Fairness Constraints. *CoRR*, abs/1901.08668, 2019. `arXiv:1901.08668`.

31 Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2018)*, pages 646–659. ACM, 2018. `doi:10.1145/3188745.3188882`.

32 Euiwoong Lee, Melanie Schmidt, and John Wright. Improved and simplified inapproximability for k-means. *Information Processing Letters*, 120:40–43, 2017. `doi:10.1016/j.ipl.2016.11.009`.

33 Jian Li, Ke Yi, and Qin Zhang. Clustering with Diversity. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming (ICALP 2010)*, volume 6198 of *Lecture Notes in Computer Science (LNCS)*, pages 188–200. Springer Berlin Heidelberg, 2010. `doi:10.1007/978-3-642-14165-2_17`.

34 Shanfei Li. An Improved Approximation Algorithm for the Hard Uniform Capacitated k-median Problem. In Klaus Jansen, José D. P. Rolim, Nikhil R. Devanur, and Cristopher Moore, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*, volume 28 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 325–338. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.325.

35 Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Information and Computation*, 222:45–58, 2013. doi:10.1016/j.ic.2012.01.007.

36 Shi Li. Approximating capacitated k-median with $(1 + \epsilon)k$ open facilities. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2016)*, pages 786–796. SIAM, 2016. doi:10.1137/1.9781611974331.ch56.

37 Shi Li. On Uniform Capacitated k-Median Beyond the Natural LP Relaxation. *ACM Transactions on Algorithms*, 13:22:1–22:18, 2017. doi:10.1145/2983633.

38 Shi Li and Ola Svensson. Approximating k-Median via Pseudo-Approximation. *SIAM Journal on Computing*, 45:530–547, 2016. doi:10.1137/130938645.

39 Andrea Romei and Salvatore Ruggieri. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review*, 29:582–638, 2014. doi:10.1017/S0269888913000039.

40 Clemens Rösner and Melanie Schmidt. Privacy Preserving Clustering with Constraints. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 96:1–96:14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.96.

41 Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair Coresets and Streaming Algorithms for Fair k-Means Clustering. *CoRR*, abs/1812.10854, 2018. arXiv:1812.10854.

42 Zoya Svitkina. Lower-bounded facility location. *ACM Transaction on Algorithms*, 6:69:1–69:16, 2010. doi:10.1145/1824777.1824789.

43 Indrė Žliobaitė, Faisal Kamiran, and Toon Calders. Handling Conditional Discrimination. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*, ICDM '11, pages 992–1001. IEEE Computer Society, 2011. doi:10.1109/ICDM.2011.72.

44 Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning Fair Representations. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, volume 28 of *Proceedings of Machine Learning Research*, pages 325–333. PMLR, 2013. URL: http://proceedings.mlr.press/v28/zemel13.html.

A NP-hardness of the fair assignment problem for k-center

In this section, we reduce the Exact Cover by 3-sets to the fair assignment problem for k-center. The input to the Exact Cover by 3-sets problem is a ground set $\mathcal{U}$ of elements and a family $\mathcal{F}$ of subsets such that each set has exactly three elements from $\mathcal{U}$. The objective is to find a set cover such that each element is included in exactly one set. For example, let $\mathcal{U} = \{a, b, c, d, e, f\}, \mathcal{F} = \{A = \{a, b, c\}, B = \{b, c, d\}, C = \{d, e, f\}\}$ be an instance. The set $\{A, C\}$ is an exact cover. We call the problem of computing a cost-minimal fair assignment of points to given centers the *fair assignment problem*. It exists once for every objective listed above. Even for k-center, the fair assignment problem is NP-hard. This can be shown by a reduction from Exact Cover by 3-sets, a variant of set cover. The input is a ground set $\mathcal{U}$ of elements and a family $\mathcal{F}$ of subsets such that each set has exactly three elements from $\mathcal{U}$. The objective is to find a set cover such that each element is included in exactly one set. For example, let $\mathcal{U} = \{a, b, c, d, e, f\}, \mathcal{F} = \{A = \{a, b, c\}, B = \{b, c, d\}, C = \{d, e, f\}\}$ be an instance. The set $\{A, C\}$ is an exact cover.

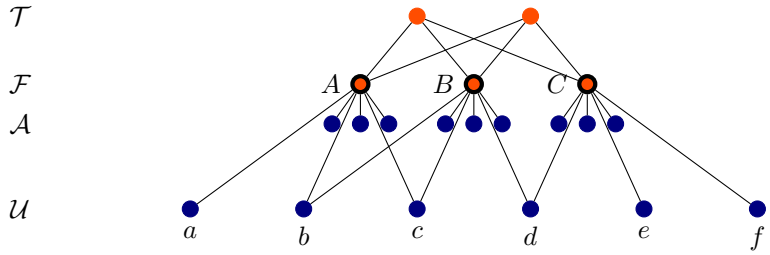


Figure 2 Example for the reduction from Exact Cover with 3-sets to the fair assignment problem for k-center, with $\mathcal{U} = \{a, b, c, d, e, f\}$ and $\mathcal{F} = \{A = \{a, b, c\}, B = \{b, c, d\}, C = \{d, e, f\}\}$.

For an instance $\mathcal{U}, \mathcal{F}$ of the exact cover problem, we construct an unweighted graph, which then translates to an input for the fair assignment problem for k-center by assigning distance 1 to each edge and using the resulting graph metric. The vertices consist of $\mathcal{U}$, $\mathcal{F}$ and two sets defined below, $\mathcal{A}$ and $\mathcal{F}$. We start by adding an edge between all $e \in \mathcal{U}$ and any $A \in \mathcal{F}$ iff $e \in A$. We assign color red to the vertices from $\mathcal{F}$ and blue to those from $\mathcal{U}$. Then we construct a set $\mathcal{A}$ which contains three auxiliary blue vertices for each vertex in $\mathcal{F}$. These are exclusively connected to their corresponding vertex in $\mathcal{F}$. Then we construct a set $\mathcal{T}$ of $|\mathcal{U}|/3$ red vertices.[3] and connect each vertex in $\mathcal{T}$ to every vertex in $\mathcal{F}$. Finally, we open a center at each vertex in $\mathcal{F}$. The construction is shown in Figure 2. Observe that the distance between an element $e \in \mathcal{U}$ and an open center at $A \in \mathcal{F}$ in this construction is 1 iff $e \in A$, and otherwise, it is 3: If $e \notin A$, then there is no edge between e and A, and since there are no direct connections between the centers, the minimum distance between e and another open center is 3.

▶ **Lemma 15.** *If there exists an exact cover, there exists a fair assignment of cost* 1 *where the red:blue ratio is 1:3 for each cluster.*

Proof. Assign each red vertex $A \in \mathcal{F}$ and the three auxiliary blue vertices connected to it to the center at A. If A is in the exact cover, assign the three blue vertices representing its elements and one red vertex from $\mathcal{T}$ to the center at A. It is straightforward to verify that this assignment is fair and assigns every vertex to some center to which it is connected via a direct edge. ◀

▶ **Lemma 16.** *If there exists a fair assignment where red:blue = 1:3 for all clusters of cost less than* 3*, there exists an exact cover.*

Proof. For $A \in \mathcal{F}$, the red vertex at A and the three auxiliary blue vertices attached to it must be assigned to the center at A as this is the only center within distance less than 3. Also, no center can have more than two red vertices assigned to it because there are only six blue vertices in distance less than 3 of any center. Therefore, each red vertex in $\mathcal{T}$ must be assigned to a distinct center and each such center A will have exactly three blue vertices from $\mathcal{U}$ assigned to it which correspond to the elements in the set that A represents. Thus, the sets corresponding to the centers that have two red vertices assigned to them form an exact cover for $\mathcal{U}$. ◀

[3] Note that if $|\mathcal{U}|$ is not a multiple of three, it cannot have an exact cover, so we can assume that $|\mathcal{U}|$ is a multiple of three.

B Integrality gap of the canonical clustering LP

We show that any integral fair solution needs large clusters to implement awkward ratios of the input points. This allows us to derive a non-constant integrality gap for the canonical clustering LP.

▶ **Lemma 17.** *Let P be a point set with r red and $r-1$ blue points and let $k \geq 1$. If the ratio of red points $r_{red}(C_i)$ is at most $\frac{r-k+1}{2r-2k+1}$ for each cluster C_i, then any fair solution can have at most k clusters.*

Proof. Consider a solution with $k' > k$ clusters. Since we have more red points there must be at least one cluster C_i that contains more red points than blue points. The ratio of red points $r_{red}(C_i)$ of this cluster is minimized if the solution contains $k'-1$ clusters with one blue and one red point, and one cluster with the remaining $r-k'$ blue and $r-k'+1$ red points. However,

$$\frac{r-k'+1}{2r-2k'+1} > \frac{r-k+1}{2r-2k+1}$$

Since the highest ratio of red points in any other solution can only be higher, the claim follows. ◀

We remark that Lemma 17 is not true for essentially fair solutions.

The canonical fair clustering ILP consists of (2)–(6) and (10). In the k-median/facility location case and in the k-means case, let write OPT_F for the optimum value of its LP relaxation and and let us call the value of an optimum integral solution OPT_I. We then define the integrality gap of the ILP as $\mathrm{OPT}_I/\mathrm{OPT}_F$. In the k-center case, the ILP does not have an objective function, but we can define its integrality gap in the following sense: If τ_I, τ_F is the smallest τ such that the LP-relaxation has a feasible *integral* or *fractional* solution, respectively, then we define the integrality gap as τ_I/τ_F.

▶ **Lemma 5.** *There is a choice of non-trivial fairness intervals such that the integrality gap of the LP-relaxation of the canonical fair clustering ILP is $\Omega(n)$ for the fair k-center/k-supplier/k-median/facility location problem. The integrality gap is $\Omega(n^2)$ for the fair k-means problem.*

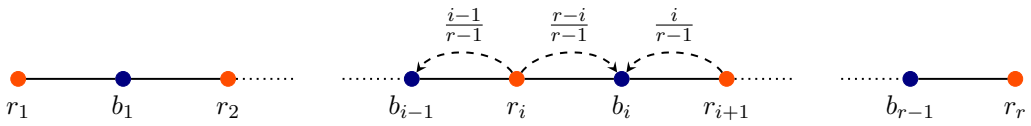


Figure 3 Integrality gap example.

Proof. Consider the input points P lying on a line, as shown in Figure 3. Specifically, we have r red points $\{r_1, r_2, \ldots, r_r\}$ that alternate with $r-1$ blue points $\{b_1, b_2, \ldots, b_{r-1}\}$. The distance between consecutive points is 1.

We require that the ratio of the red points of each cluster is between 0 and $(r-1)/(2r-3)$ and set $k = r-1$. The input ratio $r/(2r-1)$ of the red points lies in the interior of this interval as

$$\frac{r}{2r-1} < \frac{r-1}{2r-3} \iff 2r^2 - 3r < 2r^2 - 3r + 1,$$

and thus our input is well-defined and the fairness relaxation is non-trivial. We then ask for a clustering of P with at most k centers that respects the fairness constraints.

Consider the following feasible solution for the LP-relaxation. The solution opens a center at each of the $r-1=k$ blue points and assigns the blue point to itself and the red points on each side in the following way: for each $1 \leq i \leq r-1$, assign r_i to b_i by a fraction of $\frac{r-i}{r-1}$ and for each $2 \leq i \leq r$ assign r_i to b_{i-1} a fraction of $\frac{i-1}{r-1}$. Each red point is fully assigned in this way. We also get that in a cluster around some fixed b_i, the total assignment coming from red points is $\frac{r}{r-1}$ and the assignment coming from blue points is 1; thus, each cluster has a ratio of red points of

$$\frac{\frac{r}{r-1}}{1+\frac{r}{r-1}} = \frac{\frac{r}{r-1}}{\frac{2r-1}{r-1}} = \frac{r}{2r-1}.$$

We therefore respect the balance requirements.

However, as $(r-1)/(2r-3) = (r-k'+1)/(2r-2k'+1)$ for $k'=2$, by Lemma 17 any integral solution satisfying the ratio requirement can at most open two centers.

- In the k-center case, the fractional solution has a radius of 1 and the integral solution has a radius of at least $\lfloor (r-1)/2 \rfloor = \Omega(n)$. The k-center problem is a special case of the k-supplier problem; thus, the integrality gap for the k-supplier problem can only be larger.
- In the k-median case, the fractional solution has a cost of $O(n)$: The blue points incur no cost and each red point r_i contributes $(r-i)/(r-1) \cdot 1 + (i-1)/(r-1) \cdot 1 = 1$ to the objective function. Since the optimum integral solution can have at most two centers, it has to contain one cluster spanning at least $\lfloor r/2 \rfloor$ consecutive points. This incurs a cost of at least $2 \cdot \sum_{j=1}^{\lfloor r/4 \rfloor - 1} j = \Omega(n^2)$.
- In the facility location case, we observe that we can open at most two facilities in a fair integral solution. Hence, the analysis for the k-median case carries over (even if we set all opening costs to zero).
- In the k-means case, each red point r_i incurs a cost of $(r-i)/(r-1) \cdot 1^2 + (i-1)/(r-1) \cdot 1^2 = 1$ in the fractional solution; the blue points again incur no cost as they are chosen as centers. However, the integral solution now has a cost of at least $2 \cdot \sum_{j=1}^{\lfloor r/4 \rfloor - 1} j^2 = \Omega(n^3)$. ◀

This integrality gap yields a lower bound on the quality guarantee of any LP-rounding approach for this ILP. Thus, Lemma 5 implies that no fair constant factor approximation can be achieved by rounding the canoncial fair clustering ILP. The counterexample in 5 breaks down in the essential fairness model.

C Facts about the k-means cost function

We use some well-known facts about the k-means function when extending our results for k-median to k-means. The first one is that squared distances satisfy a relaxed triangle inequality:

▶ **Lemma 18.** *It holds for all $x, y, z \in \mathbb{R}^d$ that*

$$||x-z||^2 \leq 2||x-z||^2 + 2||z-y||^2.$$

The next lemma is also a folklore statement which can be extremely useful. It implies that the best 1-means is always the centroid of a point set, and has further consequences, like Lemma 20 which we state below, a fact which is also commonly used in approximation algorithms for the k-means problem.

▶ **Lemma 19.** *For any $P \subset \mathbb{R}^d$, and $z \in \mathbb{R}^d$,*

$$\sum_{x \in P} ||x - z||^2 = \sum_{x \in P} ||x - \mu(P)||^2 + |P| \cdot ||\mu(P) - z||^2,$$

where $\mu(P) = \frac{1}{|P|} \sum_{x \in P} x$ is the centroid of P.

One corollary of Lemma 19 is that the optimum cost of the best discrete solution is not much more expensive than the best choice of centers from $\mathbb{R}^d$.

▶ **Lemma 20.** *Let $P \subset \mathbb{R}^d$ be a set of point in the Euclidean space, and let $S^* \subset \mathbb{R}^d$ be a set of k points that minimizes the k-means objective, i.e., it minimizes*

$$\sum_{x \in P} \min_{c \in S} ||x - c||^2$$

over all choices of $S \subset \mathbb{R}^d$ with $|S| = k$. Furthermore, let $\hat{S}$ be the set of centers that minimizes the k-means objective over all choices of $S \subset P$ with $|S| = k$, i.e., the best choice of centers from P itself. Then it holds that

$$\sum_{x \in P} \min_{c \in \hat{S}} ||x - c||^2 \leq \sum_{x \in P} \min_{c \in S^*} ||x - c||^2.$$

Thus, restricting the set of centers to the input point set increases the cost of an optimal solution by a factor of at most 2.

The Maximum Exposure Problem

Neeraj Kumar
Department of Computer Science, University of California, Santa Barbara, USA
neeraj@cs.ucsb.edu

Stavros Sintos
Duke University, Durham, NC, USA
ssintos@cs.duke.edu

Subhash Suri
Department of Computer Science, University of California, Santa Barbara, USA
suri@cs.ucsb.edu

Abstract

Given a set of points P and axis-aligned rectangles $\mathcal{R}$ in the plane, a point $p \in P$ is called *exposed* if it lies outside all rectangles in $\mathcal{R}$. In the *max-exposure problem*, given an integer parameter k, we want to delete k rectangles from $\mathcal{R}$ so as to maximize the number of exposed points. We show that the problem is NP-hard and assuming plausible complexity conjectures is also hard to approximate even when rectangles in $\mathcal{R}$ are translates of two fixed rectangles. However, if $\mathcal{R}$ only consists of translates of a single rectangle, we present a polynomial-time approximation scheme. For general rectangle range space, we present a simple $O(k)$ bicriteria approximation algorithm; that is by deleting $O(k^2)$ rectangles, we can expose at least $\Omega(1/k)$ of the optimal number of points.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases max-exposure, PTAS, densest k-subgraphs, geometric constraint removal, Network resilience

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.19

Category APPROX

Funding Work by Kumar and Suri is supported by NSF under Grant CCF-1814172. Work by Sintos is supported by NSF under grants CCF-15-13816, CCF-15-46392, and IIS-14-08846, by ARO grant W911NF-15-1-0408, and by Grant 2012/229 from the U.S.-Israel Binational Science Foundation.

1 Introduction

Let $S = (P, \mathcal{R})$ be a geometric set system, also called a *range space*, where P is a set of points and each $R \in \mathcal{R}$ is a collection of subsets of P, also called a range. We are primarily interested in range spaces defined by a set of points in two dimensions and ranges defined by axis-aligned rectangles. We say that a point $p \in P$ is *exposed* if no range in $\mathcal{R}$ contains p. The *max-exposure* problem is defined as follows: given a range space $(P, \mathcal{R})$ and an integer parameter $k \geq 1$, remove k ranges from $\mathcal{R}$ so that a maximum number of points are exposed. That is, we want to find a subfamily $\mathcal{R}^* \subseteq \mathcal{R}$ with $|\mathcal{R}^*| = k$, so that the number of exposed points in the (reduced) range space $(P, \mathcal{R} \setminus \mathcal{R}^*)$ is maximized.

The max-exposure problem arises naturally in many geometric coverage settings. For instance, if points are the location of clients, and ranges are coverage of some facilities in the plane, then exposed points are those not covered by any facility. The max-exposure problem in this case gives a worst-case bound on the number of clients that can be exposed if an adversary disables k facilities. Similarly, in distributed sensor networks, ranges correspond to *sensing zones*, points correspond to physical assets being monitored by the network, and the max-exposure problem computes the number of assets exposed when k sensors are compromised.


© Neeraj Kumar, Stavros Sintos, and Subhash Suri;
licensed under Creative Commons License CC-BY



More broadly, the max-exposure problem is related to the densest k-subgraph problem in *hypergraphs*. In the *densest k-subhypergraph* problem, we are given a hypergraph $H = (X, E)$, and we want to find a set of k vertices with a maximum number of induced hyperedges. In general hypergraphs, finding k-densest subgraphs is known to be (conditionally) hard to approximate within a factor of $n^{1-\epsilon}$, where n is the number of vertices. The max-exposure problem is equivalent to the densest k-subhypergraph problem on a *dual* hypergraph, the vertex set X corresponds to the ranges $\mathcal{R}$, and set of edges E of the dual hypergraph correspond to the set of points P. In the rest of the paper, we will use $n = |\mathcal{R}|$ for the number of ranges in $\mathcal{R}$ and $m = |P|$ to be the number of points. We show that if the range space is defined by *convex polygons*, then the max-exposure problem is just as hard as the densest k-subhypergraph problem. However, for ranges defined by *axis-aligned rectangles*, one can achieve much better approximation. In particular, we obtain the following results.

- We show that the max-exposure problem is NP-hard and assuming the *dense vs random* conjecture to be true, it is also hard to approximate better than a factor of $O(n^{1/4})$ even if the range space is defined by *only two types of rectangles* in the plane. (For range space defined by convex polygons, we show that max-exposure is equivalent to densest k-subhypergraph problem, which is hard to approximate within $O(n^{1-\epsilon})$).
- When ranges are defined by translates of a *single* rectangle, we give a polynomial-time approximation scheme (PTAS) for max-exposure. The PTAS stands in sharp contrast to the inapproximability of ranges defined by *two* types of rectangles. Moreover, as an easy consequence of this result, we obtain a constant approximation when the ratio of longest and smallest side of rectangles in $\mathcal{R}$ is bounded by a constant. However, we do not know if max-exposure with translates of a single rectangle can be solved in polynomial time or is NP-hard.
- For ranges defined by arbitrary rectangles, we present a simple greedy algorithm that achieves a bicriteria $O(k)$-approximation. No such approximation is possible for general hypergraphs. If rectangles in $\mathcal{R}$ have a bounded aspect ratio, the approximation improves to $O(\sqrt{k})$.

Related Work. Coverage and exposure problems have been widely studied in geometry and graphs. In the classical *set cover* problem, we want to select a subfamily of k sets that cover the maximum number of items (points) [14, 17]. For the set cover problem, the classical greedy algorithm achieves a factor $\log n$ approximation on the number of sets needed to cover all the items, or factor $(1 - 1/e)$ approximation on the number of items covered by using exactly k sets. Similarly, in geometry, the art gallery problems explore coverage of polygons using a minimum number of guards. Unlike coverage problems where greedy algorithms deliver reasonably good approximation, the exposure problems turn out to be much harder. Specifically, choosing k sets whose union is of *minimum size* is much harder to approximate with a conditional inapproximability of $O(n^{1-\epsilon})$ where n is the number of elements and $O(m^{1/4-\epsilon})$ where m is the number of sets [10]. This so-called *min-union* problem is essentially the densest k-subgraph problem on hypergraphs [9]. The densest k-subgraph problem for graphs has a long history [15, 3, 2, 6]. The classical coverage problems have been extensively studied for geometric set systems and significantly better approximation bounds have been achieved for them [1, 7, 20]. Several other variations such as the set multi-cover problem [8, 12] where each input point needs to be covered by more than one set have also been studied. Also closely related to max-exposure is the geometric constraint removal problem [4, 13], where given a set of ranges, the goal is to *expose* a path

between two given points by deleting at most k ranges (a path is exposed if it lies in the exterior of all ranges). Even for simple shapes such as unit disks (or unit squares) [5, 19], no PTAS is known for this problem.

The remainder of the paper is organized as follows. In Section 2, we discuss our hardness results followed by the bicriteria $O(k)$-approximation in Section 3. In Section 4, we study the case when $\mathcal{R}$ consists of translates of a fixed rectangle and describe a PTAS for it. Finally, in Section 5, we use these ideas to obtain a bicriteria $O(\sqrt{k})$-approximation when aspect ratio of rectangles in $\mathcal{R}$ is bounded by a constant.

2 Hardness of Max-Exposure

We show that max-exposure problem for geometric ranges is both NP-hard and inapproximable within a polynomial factor, under some well known hardness conjectures. In particular, we first show that the densest k-subgraph on bipartite graphs (*bipartite-DkS*) can be easily reduced to the max-exposure problem. In the *bipartite-DkS* problem, we are given a bipartite graph $G = (A, B, E)$, an integer k, and we want to compute a set of k vertices such that the induced subgraph on those k vertices has the maximum number of edges. Given an instance $G = (A, B, E)$ of bipartite-DkS, we will construct a max-exposure instance as follows.

Let $R_1 = [0, \epsilon] \times [0, n]$ be a thin vertical rectangle and $R_2 = [0, n] \times [0, \epsilon]$ be a thin horizontal rectangle. For each vertex $v_i \in A$, we create a copy R_i of R_1, and place it such that its lower-left corner is at $(i, 0)$. Similarly, for each vertex $v_j \in B$, we create a copy R_j of R_2, and place it such that its lower-left corner is at $(0, j)$. These $|A| + |B|$ rectangles create a checkerboard arrangement, with $|A| \times |B|$ cells of intersection. For each edge $(v_i, v_j) \in E$, we place a single point in the cell corresponding to intersection of R_i and R_j. It is now easy to see that G has a k-subgraph with m^* edges if and only if we can expose m^* points in this instance by removing k-rectangles: the removed rectangles are exactly the k vertices chosen in the graph, and each exposed point corresponds to the edge included in the output subgraph. (See also Figure 1.)

▶ **Lemma 1.** *The max-exposure problem is at least as hard as bipartite-DkS.*

Since bipartite-DkS is known to be NP-hard [16], we have the following.

▶ **Theorem 2.** *Max-exposure problem with axis-aligned rectangles is NP-hard.*

2.1 Hardness of Approximation

The construction in the preceding proof shows that max-exposure with rectangles is at least as hard as bipartite-DkS problem. Moreover, the geometric construction uses translates of *only two rectangles* R_1, R_2. In the following, we show that even with such a restricted range space, the problem is also hard to approximate. To that end we prove that bipartite-DkS cannot be approximated better than a factor $O(n^{1/4})$, where n is the number of vertices in this graph. More precisely, if the densest subgraph over k vertices has m^* edges, it is hard to find a subgraph over k vertices that contains at least $\Omega(m^*/n^{1/4-\epsilon})$ edges in polynomial time. This hardness of approximation is conditioned on the so-called *dense vs random* conjecture [10] being true. Roughly speaking, we are given a graph G, constants $0 < \alpha, \beta < 1$, and a parameter k, and we want to distinguish between the following two cases.

1. (Random) $G = G(n, p)$ where $p = n^{\alpha-1}$, that is, G has average degree approximately n^α.
2. (Dense) G is adversarially chosen so that the densest k-subgraph of G has average degree k^β.

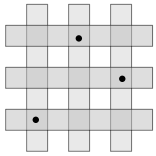


Figure 1 Reducing bipartite-DkS to max-exposure with axis-aligned rectangles.

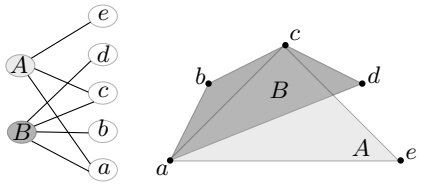


Figure 2 Reducing densest k-subhypergraph problem to max-exposure. Hypergraph vertices A, B shown as convex ranges.

The conjecture states that for all $0 < \alpha < 1$, sufficiently small $\epsilon > 0$, and for all $k \leq \sqrt{n}$, one cannot distinguish between the *dense* and *random* cases in polynomial time (w.h.p), when $\beta \leq \alpha - \epsilon$.

In order to obtain hardness guarantees using the above conjecture, one needs to find the "distinguishing ratio" r, that is the least multiplicative gap between the optimum solution for the problem on the dense and random instances. If there exists an algorithm with an approximation factor significantly smaller than r, then we would be able to use it to distinguish between the dense and random instances, thereby refuting the conjecture. We obtain the following result for densest k-subgraph problem on bipartite graphs. (See Appendix A.1 for a proof.)

▶ **Lemma 3.** *Assuming that* dense vs random *conjecture is true, the densest k-subgraph problem on bipartite graphs is hard to approximate better than a factor $O(n^{1/4})$ of optimum.*

Using the same construction as in Lemma 1, we obtain the following.

▶ **Corollary 4.** *Assuming the dense vs random conjecture, max-exposure with axis-aligned rectangles is hard to approximate better than a factor $O(n^{1/4})$ of optimum.*

Hardness of Max-exposure with Convex Polygons

If the range space $(P, \mathcal{R})$ consists of convex polygons, the max-exposure problem is *equivalent* to the densest k-subhypergraph problem for general hypergraphs. A max-exposure instance $(P, \mathcal{R})$ naturally corresponds to a hypergraph $H = (\mathcal{R}, P)$ whose vertices are the ranges and the edges correspond to points and are defined by the containment relationship. Clearly, the densest k-subhypergraph corresponds to the set of k ranges deleting which exposes maximum number of points. For the other direction, we have the following lemma. (See also Figure 2.)

▶ **Lemma 5.** *Given a hypergraph $H = (X, E)$, one can construct a max-exposure instance with convex ranges $\mathcal{R}$ and points P such that the densest k-subhypergraph of H corresponds to a solution of max-exposure.*

Proof. For each edge $e \in E$ of the hypergraph, add a point $p_e \in P$. We place all the points of P in convex position. Let $v \in X$ be a vertex and E_v be the set of hyperedges adjacent to v. Then for every $v \in X$, we add a convex polygon $R_v \in \mathcal{R}$ such that the corners of R_v is precisely the point set E_v. Note that this is possible since points of P are in convex position. It is easy to see that in order to include an edge e (expose p_e), we must include all vertices in E_v, which corresponds to removing all polygons corresponding to vertices in E_v. ◀

3 A Bicriteria $O(k)$-approximation Algorithm

In this section, we present a simple approximation algorithm for the max-exposure problem that achieves bicriteria $O(k)$-approximation for range spaces defined by arbitrary axis-aligned rectangles. Specifically, if the optimal number of points exposed is m^*, the algorithm picks a subset of k^2 rectangles such that the number of points exposed is at least m^*/ck, for some constant c. In fact, the results hold for any polygonal range with $O(1)$ complexity.

This bicriteria approximation should be contrasted with the fact that no such approximation is possible for for the densest k-subhypergraph problem: that is, one cannot compute a set of $O(k^b)$ vertices for any constant b such that the number of edges in the induced subhypergraph is at least optimal. Thus the geometric properties of the range space have a significant impact on the problem complexity. In particular, if $\mathcal{R}$ consists of rectangle ranges, we show that the following strategy picks a subset of αk ranges such that the number of points exposed is at least $\alpha m^*/ck^2$, for a parameter $1 \leq \alpha \leq k$ and constant c that will be fixed later. Choosing $\alpha = k$ gives us the claimed bound.

Our algorithm is essentially greedy. We divide the points into maximal equivalence classes, where each class is the maximal subset of points belonging to the same subset of ranges. We define $\mathcal{R}(p)$ as the set of ranges that contain a point $p \in P$, and remove all points that are contained in more than k ranges, since they can be never exposed in the optimal solution. Therefore, without loss of generality, we can assume that $|\mathcal{R}(p)| \leq k$ for all points $p \in P$.

Algorithm 1 Greedy-Bicriteria.

1. Partition P into a set $\mathcal{G}$ of groups where each group $G_i \in \mathcal{G}$ is an equivalence class of points that are contained in the same set of ranges. That is, for any $p \in G_i$, $p' \in G_j$, we have $\mathcal{R}(p) = \mathcal{R}(p')$ if $i = j$ and $\mathcal{R}(p) \neq \mathcal{R}(p')$, otherwise.
2. Sort the groups in $\mathcal{G}$ by decreasing order of their size $|G_i|$ and select the first α groups. Return $m' = \sum_{1 \leq i \leq \alpha} |G_i|$ as the number of points exposed.

Observe that every point $p \in G_i$ is contained in the same set of ranges $\mathcal{R}_i = \mathcal{R}(p)$ and $|\mathcal{R}_i| \leq k$. Therefore, the total number of ranges that we remove is at most αk. It remains to show that the number of points exposed m' is at least $\alpha m^*/ck^2$.

▶ **Lemma 6.** *Let m' be the number of points exposed by the algorithm* Greedy-Bicriteria*, and let m^* be the optimal number of exposed points, Then, $m' \geq \alpha m^*/ck^2$.*

Proof. Consider the optimal set $\mathcal{R}^*$ of k ranges that are deleted, and let P^* be the set of exposed points. We partition the set of points P^* into groups $\mathcal{G}^*$ as before, such that each group $G_i^* \in \mathcal{G}^*$ is identified by the range set $\mathcal{R}_i^* = \mathcal{R}(p)$, for any $p \in G_i^*$. Since $P^* \subseteq P$, we must have that $\mathcal{G}^* \subseteq \mathcal{G}$. This holds because for every group $G_i^* \in \mathcal{G}^*$ there must be a group $G_i \in \mathcal{G}$ such that $\mathcal{R}_i^* = \mathcal{R}_i$. Moreover since P^* is the maximum set of points that can be exposed, we must have that $G_i^* = G_i$. Finally, we note that the number of groups $|\mathcal{G}^*|$ is bounded by the number of cells in the arrangement of ranges in $\mathcal{R}^*$ which is at most ck^2 for some fixed constant c, for all $O(1)$-complexity ranges.

If the groups in $\mathcal{G}$ are arranged by decreasing order of their sizes, we have that

$$m^* = \sum_{1 \leq i \leq |\mathcal{G}^*|} |G_i^*| \leq \sum_{1 \leq i \leq |\mathcal{G}^*|} |G_i| \leq \sum_{1 \leq i \leq ck^2} |G_i| \leq \frac{ck^2}{\alpha} \sum_{1 \leq i \leq \alpha} |G_i| = \frac{ck^2}{\alpha} \cdot m' \quad \blacktriangleleft$$

The parameter α can be tuned to improve the approximation guarantee with respect to one criterion (say the number of exposed points) at the cost of other. With $\alpha = k$, the algorithm exposes at least $\Omega(m^*/k)$ by removing k^2 ranges. If the range space $\mathcal{R}$ consists of pseudodisk of *bounded-ply* (no point in the plane is incident to more than a constant number ρ of pseudodisks), then the algorithm Greedy-Bicriteria achieves an $O(\rho)$ approximation. This holds because the number of cells in an arrangement of k pseudodisks with depth at most ρ is $O(\rho k)$ [11].

4 A PTAS for Unit Square Ranges

We have seen that max-exposure is hard to approximate even if the ranges are translates of two types of rectangles. We now describe an approximation scheme when the ranges are translates of a *single* rectangle. In this case, we can scale the axes so that the rectangle becomes a *unit square* without changing any point-rectangle containment. Therefore, we can assume that our ranges are all unit squares. The problem is non-trivial even for unit square ranges, and as a warmup we first solve the following special case: *all the points lie inside a unit square.* We develop a dynamic programming algorithm to solve this case exactly, and then use it to design an approximation for the general set of points.

4.1 Exact Solution in a Unit Square

We are given a max-exposure instance consisting of unit square ranges $\mathcal{R}$ and a set of points P in a unit square C. Without loss of generality, we can assume that the lower left corner of C lies at origin $(0, 0)$ and all ranges in $\mathcal{R}$ intersect C. We classify the ranges in $\mathcal{R}$ to be one of the two types: (See also Figure 3).

Type-0 : Unit square ranges that intersect $x = 0$.
Type-1 : Unit square ranges that intersect $x = 1$.

(A unit square range coincident with both $x = 0$ and $x = 1$ is assumed to be Type-0). We draw two parallel horizontal lines $\ell_0 : y = 0$ and $\ell_1 : y = 1$ coincident with bottom and top horizontal sides of C respectively. We say that a range $R \in \mathcal{R}$ is *anchored* to a line ℓ if it intersects ℓ. Note that every $R \in \mathcal{R}$ is anchored to exactly one of ℓ_0 or ℓ_1. (When R is coincident with both ℓ_0 and ℓ_1, we say that it is anchored to ℓ_0). Moreover, for the rest of our discussion, let $x = x_i$ be a vertical line and define $P_i \subseteq P$ to be the set of points that have x-coordinate at least x_i. Similarly, define $\mathcal{R}_i \subseteq \mathcal{R}$ to be the set of ranges that have at least one corner to the right of $x = x_i$. That is a range $R \in \mathcal{R}_i$ either intersects $x = x_i$ or lies completely to the right of it.

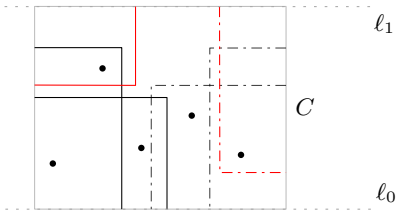


Figure 3 Max-exposure in a unit square C. Type 0 ranges are drawn with solid lines, Type 1 ranges are dash-dotted.

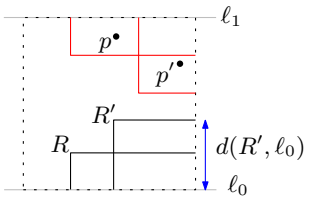


Figure 4 An example of *closer* relationship. Point p is *closer* to ℓ_1 than p'. R is closer to ℓ_0 than R'.

In order to gain some intuition, we will first consider the following two natural dynamic programming formulations for the problem.

DP-template-0. Suppose that the points in P are ordered by their increasing x-coordinates and let x_i be the x-coordinate of the ith point p_i. We define a subproblem as $S(i, k', \mathcal{R}_d)$ which represents the maximum number of points in P_i that can be exposed by removing k' ranges from the set $\mathcal{R}_i \setminus \mathcal{R}_d$. If we define $x_0 = 0$, then $S(0, k, \emptyset)$ gives the optimal number of exposed points for our problem.

Let $k_i = |\mathcal{R}(p_i) \setminus \mathcal{R}_d|$ be the number of ranges of $\mathcal{R}_i \setminus \mathcal{R}_d$ that contain p_i. Then, we can can express the subproblems at i in terms of subproblems at $i+1$ as follows.

$$S(i,\ k', \mathcal{R}_d) = \max \begin{cases} S(i+1,\ k' - k_i,\ \mathcal{R}_d \cup \mathcal{R}(p_i)) \ +\ 1 & \text{expose } p_i \\ S\,(i+1,\ k',\ \mathcal{R}_d) & \text{otherwise} \end{cases}$$

Roughly speaking, at $x = x_i$ which is the *event* corresponding to a point $p_i \in P$, we have two choices : *expose* p_i or *do not expose* p_i. If we expose p_i, we pay for deleting the ranges in $\mathcal{R}_i \setminus \mathcal{R}_d$ that contain p_i and mark them as deleted by adding to the *deleted range set* $\mathcal{R}_d$. Moreover, since we only delete ranges from $\mathcal{R}_i \setminus \mathcal{R}_d$, we can assume that $\mathcal{R}_d = \mathcal{R}_d \cap \mathcal{R}_i$ at each x_i. It is easy to see that this correctly computes the optimal number of exposed points. However, there is one complication: a priori it is not clear how to bound the number of range subset $\mathcal{R}_d$ used by this dynamic program. We later argue that the geometry of range space for Type-0 ranges allows us to use only a polynomial number of choices.

DP-template-1. An alternative approach is to consider both *point* and *begin-range* events. That is, $x = x_i$ is either incident to a point $p_i \in P$ or to the left vertical side of a range $R_i \in \mathcal{R}$. Then, we can define a subproblem by the tuple $S(i, k', P_f)$ which represents the maximum number of points in $(P_i \setminus P_f)$ that can be exposed by removing k' ranges in $\mathcal{R}_i$. If we define $x_0 = 0$, then $S(0, k, \emptyset)$ gives the optimal number of exposed points. Let $P(R_i) \subseteq P$ be the set of points contained in the range R_i, then we have the following recurrence.

$$\begin{aligned} S(i,\ k', P_f) \ &=\ \max \begin{cases} S(i+1,\ k'-1,\ P_f) & \text{delete range } R_i \\ S(i+1,\ k',\ P_f \cup P(R_i)) & \text{otherwise} \end{cases} \\ &\quad \textit{(event } x = x_i \textit{ was beginning of a range } R_i \in \mathcal{R}_i\textit{)} \\ &=\ \max \begin{cases} S(i+1,\ k', P_f) & \text{if } p_i \in P_f\text{, cannot expose } p_i \\ S(i+1,\ k', P_f) + 1 & \text{otherwise, expose } p_i \end{cases} \\ &\quad \textit{(otherwise, event } x = x_i \textit{ was a point } p_i \in P_i\textit{)} \end{aligned}$$

In the above formulation, at each *begin-range* event for some $R_i \in \mathcal{R}_i$, we have two choices: *delete* R_i or *do not delete* R_i. If R_i was deleted, we reduce the budget k' by one. Otherwise, if R_i was not deleted, we can never expose the points in $P(R_i)$, and therefore we add $P(R_i)$ to the *forbidden point set* P_f. The correctness of the dynamic program follows from the fact that for every point p_i, all the ranges containing it must begin before $x = x_i$, and we expose p_i only if those ranges were *deleted*. Finally, since we only expose points in $P_i \setminus P_f$, we can assume that $P_f = P_f \cap P_i$ at each x_i. Again, it is not obvious how many different subsets P_f are needed by the dynamic program. However, we will later show that by keeping track of polynomial number of sets P_f, we can solve max-exposure with Type-1 ranges.

We note that the Type-0 and Type-1 ranges may superficially seem symmetric but once we fix the order of computing subproblems, they become structurally different. Therefore, we would need slightly different techniques to handle each type. For the ease of exposition, we present dynamic programs for Type-0 and Type-1 ranges separately and finally combine them.

We first define the following ordering relations that will be useful. Let ℓ be a horizontal line, and let $d(p, \ell)$ denote the orthogonal distance of $p \in P$ from ℓ. If $p, p' \in P$ are two points, we say that p is *closer* to ℓ than p' if $d(p, \ell) < d(p', \ell)$. Similarly, for a range $R \in \mathcal{R}$ that is anchored to ℓ, let $d(R, \ell)$ be the vertical distance *inside the unit square* C between ℓ and the side of R parallel to ℓ. If $R, R' \in \mathcal{R}$ are two ranges, we say that R is *closer* (or equivalently R' is *farther*) from ℓ if both R, R' are anchored to ℓ and $d(R, \ell) < d(R', \ell)$. (See Figure 4.)

4.1.1 Max-exposure with Type-0 Ranges

Recall that Type-0 ranges intersect the vertical lines $x = 0$ and are anchored to either ℓ_0 or ℓ_1. We will apply the formulation discussed in *DP-template-0*. The key challenge here is to bound the number of possible deleted range sets $\mathcal{R}_d$. Towards that end, we make the following claim.

▶ **Lemma 7.** *Let q_0, q_1 be the two exposed points strictly to the left of $x = x_i$ that are closest to ℓ_0 and ℓ_1 respectively. Then our dynamic program only needs to consider the set of deleted ranges $\mathcal{R}_d = \mathcal{R}(q_0) \cup \mathcal{R}(q_1)$ at $x = x_i$ conditioned on q_0, q_1.*

Proof. Observe that since $\mathcal{R}$ consists of Type-0 ranges, every range in $\mathcal{R}_i$ must intersect the vertical line $x = x_i$. Suppose we partition $\mathcal{R}_i$ into ranges $\mathcal{R}_i^0$ that are anchored to ℓ_0 and $\mathcal{R}_i^1$ that are anchored to ℓ_1. Let $P' \subseteq P$ be the set of all *exposed points* strictly to the left of $x = x_i$. Observe that for all $p \in P'$, any range $R \in \mathcal{R}_i^0$ that contains p must also contain q_0. Therefore, we must have $\mathcal{R}_i^0 \cap \mathcal{R}(p) \subseteq \mathcal{R}_i^0 \cap \mathcal{R}(q_0)$, for all $p \in P'$. Similarly, $\mathcal{R}_i^1 \cap \mathcal{R}(p) \subseteq \mathcal{R}_i^1 \cap \mathcal{R}(q_1)$, for all $p \in P'$. Hence, $\bigcup_{p \in P'} \mathcal{R}_i \cap \mathcal{R}(p) = \mathcal{R}(q_0) \cup \mathcal{R}(q_1)$. Recall that $\mathcal{R}_d$ is precisely the set of ranges at $x = x_i$ that contain any exposed point to the left of $x = x_i$, so we have $\mathcal{R}_d = \mathcal{R}(q_0) \cup \mathcal{R}(q_1)$. ◀

Therefore, if our dynamic program remembers the exposed points q_0, q_1, then we can compute the deleted range set $\mathcal{R}_d = \mathcal{R}(q_0) \cup \mathcal{R}(q_1)$ at $x = x_i$. There are $O(m^2)$ choices for the pair q_0, q_1, so the number of possible sets $\mathcal{R}_d$ is also $O(m^2)$. We can therefore identify our subproblems by the tuple $S(i,\ k',\ q_0,\ q_1)$ which represents the maximum number of exposed points with x-coordinates x_i or higher using k' rectangles from the set $\mathcal{R}_i \setminus \mathcal{R}_d$. With $k_i = |\mathcal{R}(p_i) \setminus \mathcal{R}_d|$, we obtain the following recurrence:

$$S(i,\ k',\ q_0,\ q_1) = \max \begin{cases} S\left(i+1,\ k' - k_i,\ \mathit{closer}(p_i, q_0),\ \mathit{closer}(p_i, q_1)\right) \ +\ 1 & \text{expose } p_i \\ S\left(i+1,\ k',\ q_0,\ q_1\right) & \text{otherwise} \end{cases}$$

where the function $\mathit{closer}(p_i, q_0)$ returns whichever of p_i, q_0 is closer to ℓ_0, and $\mathit{closer}(p_i, q_1)$ returns whichever of p_i, q_1 is closer to ℓ_1. The optimal solution is given by $S(0,\ k,\ q_0^*,\ q_1^*)$, where $q_0^* = (0, 1)$ and $q_1^* = (0, 0)$ are two artificial points with $\mathcal{R}(q_0^*) = \mathcal{R}(q_1^*) = \emptyset$ (not contained in any range). The base case is defined by the vertical line $x = 1$ and is initialized with zeroes for all q_0, q_1 and $k' \geq 0$. Any subproblem with $k' < 0$ has value $-\infty$.

4.1.2 Max-exposure with Type-1 Ranges

Next we consider the case when we only have Type-1 ranges in $\mathcal{R}$. Unfortunately in this case, our previous dynamic program does not work and we need to remember a different set of parameters. More precisely, we will apply the formulation discussed in *DP-template-1*, and bound the number of possible forbidden point sets P_f.

▶ **Lemma 8.** *Let Q_0, Q_1 be two ranges that begin to the left of $x = x_i$ and were* not deleted. *Moreover, Q_0 is anchored to and is farthest from ℓ_0. Similarly Q_1 is anchored to and is farthest from ℓ_1 (Figure 5). Then the forbidden point set at $x = x_i$ is given by $P_f = P(Q_0) \cup P(Q_1)$, where $P(Q)$ is the set of points contained in range Q.*

Proof. Recall that the set $\mathcal{R}_i$ consists of ranges that have at least one corner to the right of the vertical line $x = x_i$. Since we are dealing with Type-1 ranges, every range that begins to the left of $x = x_i$ lies in $\mathcal{R}_i$. Now let $\mathcal{R}' \subseteq \mathcal{R}_i$ be the set of ranges that begin to the left of $x = x_i$ and were *not deleted*. Recall that P_i is the set of points in P that have x-coordinate x_i or higher. Now consider any range $R \in \mathcal{R}'$. Observe that if R was anchored to ℓ_0, then every point of P_i that lies in R also lies in Q_0. Otherwise, if R was anchored to ℓ_1, every point of P_i that lies in R also lies in Q_1. Therefore, we must have $\bigcup_{R \in \mathcal{R}'} (P_i \cap P(R)) = P(Q_0) \cup P(Q_1)$. Recall that P_f was precisely the set of points in P_i contained in ranges that begin to the left of $x = x_i$ and were not deleted. Therefore, we have that $P_f = P(Q_0) \cup P(Q_1)$. ◀

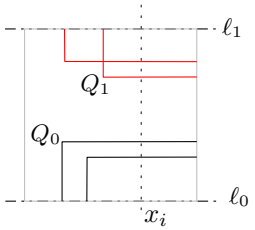


Figure 5 Undeleted ranges Q_0 and Q_1 farthest from ℓ_0 and ℓ_1 respectively.

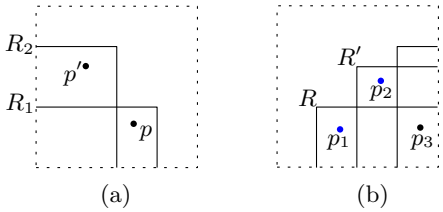


Figure 6 Remembering one of R_1, R_2 in (a) or one of p_1, p_2 in (b) is not sufficient.

Therefore, if our dynamic program remembers the ranges Q_0 and Q_1, we can compute the forbidden point set $P_f = P(Q_0) \cup P(Q_1)$ at $x = x_i$. Since there are $O(n^2)$ choices for the pair Q_0, Q_1, the number of possible sets P_f is also $O(n^2)$. We can now identify the subproblems by the tuple $S(i,\ k',\ Q_0,\ Q_1)$ which represents the maximum number of points in $P_i \setminus P_f$ that are exposed by deleting k' ranges that begin on or after $x = x_i$. This gives us the following recurrence.

$$S(i,\ k',\ Q_0,\ Q_1)\ =$$

$$\max \begin{cases} S(i+1,\ k'-1,\ Q_0,\ Q_1) & \text{delete range } R_i \\ S(i+1,\ k',\ \mathit{farther}(R_i, Q_0),\ Q_1) & \text{otherwise, } R_i \text{ is not deleted and anchored to } \ell_0 \\ S(i+1,\ k',\ Q_0,\ \mathit{farther}(R_i, Q_1)) & \text{otherwise, } R_i \text{ is not deleted and anchored to } \ell_1 \end{cases}$$

(event $x = x_i$ was beginning of a range $R_i \in \mathcal{R}$)

$$\max \begin{cases} S(i+1,\ k',\ Q_0,\ Q_1) & \text{if } p_i \in P_f, \text{ cannot expose } p_i \\ S(i+1,\ k',\ Q_0,\ Q_1) + 1 & \text{otherwise, expose } p_i \end{cases}$$

(otherwise, event $x = x_i$ was a point $p_i \in P$)

Here, *farther*(R_i, Q_0) returns whichever of R_i, Q_0 is farther from ℓ_0; and *farther*(R_i, Q_1) returns whichever of R_i, Q_1 is farther from ℓ_1. The optimal solution is given by $P(0, k, Q_0^*, Q_1^*)$, where Q_0^*, Q_1^* are two artificial ranges of zero-width : Q_0^* is anchored to ℓ_0 and is defined by corners $(0, 0)$ and $(0, 1)$; similarly, Q_1^* is anchored to ℓ_1 and is defined by corners $(0, 1)$ and $(1, 1)$.

▶ Remark 9. We note that remembering constant number of exposed points q_0, q_1 or a constant number of undeleted ranges Q_1, Q_2 by themselves cannot solve *both* Type-0 and Type-1 ranges. For instance, in Figure 6(a) with Type-0 ranges, if R_1, R_2 were both *not deleted* but we remembered one of them, then we will incorrectly expose one of p, p'. Similarly in Figure 6(b) with Type-1 ranges, if p_1, p_2 were both exposed but we only remembered one of them, we will pay for one of the ranges R, R' again when we expose p_3. However, since the previous dynamic programs for Type-0 and Type-1 ranges express subproblems at event i in terms of subproblems at event $i + 1$, we can easily combine them with minor adjustments.

4.1.3 Combining them together

In the following, we combine the dynamic programs for Type-0 and Type-1 ranges to obtain a dynamic program for max-exposure in a unit square C. We will need a couple of changes. First, the events at $x = x_i$ are now defined by either a point $p_i \in P$ or beginning of a Type-1 range R_i. Next, the *deleted range set* $\mathcal{R}_d$ at $x = x_i$ will only consist of Type-0 ranges and is defined as $\mathcal{R}_d = \mathcal{R}_{i0} \cap (\mathcal{R}(q_0) \cup \mathcal{R}(q_1))$ where $\mathcal{R}_{i0} \subseteq \mathcal{R}_i$ is the set of Type-0 ranges that intersect the vertical line $x = x_i$, The *forbidden point set* $P_f = P(Q_0) \cup P(Q_1)$ stays the same. Here q_0, q_1, Q_0, Q_1 are same as defined before. The subproblems represent the maximum number of points in $P_i \setminus P_f$ that can be exposed by deleting k' ranges from $\mathcal{R}_i \setminus \mathcal{R}_d$. If $k_i = |\mathcal{R}(p_i) \setminus \mathcal{R}_d|$, then we obtain the following combined recurrence.

$$
\begin{aligned}
&S(i,\ k',\ q_0,\ q_1, Q_0, Q_1)\ = \\
&\quad \max \begin{cases} S(i+1,\ k',\ q_0,\ q_1,\ Q_0,\ Q_1) & \text{if } p_i \in P_f, \text{ cannot expose } p_i \\ S(i+1,\ k',\ q_0,\ q_1,\ Q_0,\ Q_1) & \text{choose to not expose } p_i \\ S(i+1,\ k' - k_i,\ q_0,\ q_1,\ Q_0,\ Q_1) + 1 & \text{otherwise, expose } p_i \end{cases} \\
&\qquad (\textit{event } x = x_i \textit{ was a point } p_i \in P_i) \\
&\quad \max \begin{cases} S(i+1,\ k' - 1,\ q_0,\ q_1,\ Q_0,\ Q_1) & \text{delete Type-1 range } R_i \\ S(i+1,\ k',\ q_0,\ q_1,\ \textit{farther}(R_i, Q_0),\ Q_1) & R_i \text{ not deleted and anchored to } \ell_0 \\ S(i+1,\ k',\ q_0,\ q_1,\ Q_0,\ \textit{farther}(R_i, Q_1)) & R_i \text{ not deleted and anchored to } \ell_1 \end{cases} \\
&\qquad (\textit{event } x = x_i \textit{ was beginning of a Type-1 range } R_i \in \mathcal{R}_i)
\end{aligned}
$$

The optimal solution is given by $S(0,\ k,\ q_0^*,\ q_1^*,\ Q_0^*,\ Q_1^*)$. The correctness of the above formulation follows from the fact that when we choose to expose p_i, we are guaranteed that all Type-1 ranges in $\mathcal{R}(p_i)$ have already been deleted, and the expression k_i only charges for Type-0 ranges containing p_i. As for the running time, for each event $x = x_i$, we compute $O(kn^2m^2)$ entries and computing each entry takes constant time. Since there are $O(n + m)$ events, we obtain the following.

▶ **Lemma 10.** *Given a set P of m points in a unit square C and a set of n unit square ranges $\mathcal{R}$, we can compute their max-exposure in $O(k(n + m)n^2m^2)$ time.*

4.2 A Constant Factor Approximation

We now use the preceding algorithm to solve the max-exposure problem for general set of points and unit square ranges within a factor 4 of optimum. In particular, we compute a set of $4k$ ranges in $\mathcal{R}$ such that the number of points exposed in P by deleting them is at least the optimal number of points. Suppose we embed the ranges $\mathcal{R}$ on a uniform unit-sized grid G, and define $\mathcal{C}$ as the collection of all cells in G that contain at least one point of P. We have the following approximation algorithm.

Algorithm 2 DP-Approx.

1. Apply Lemma 10 to solve max-exposure locally in every cell $C_i \in \mathcal{C}$ for all $0 \leq k_i \leq k$. Call this a *local* solution denoted by $local(P(C_i), \mathcal{R}(C_i), k_i)$, where $P(C_i) \subseteq P$ is the set of points contained in cell C_i and $\mathcal{R}(C_i)$ is the set of ranges intersecting C_i.
2. Process cells in $\mathcal{C}$ in any order $C_1, C_2, \ldots, C_g$, and define $global(i,\ k')$ as the maximum number of points exposed in the cells C_i through C_g using k' ranges. Combine local solutions to obtain $global(i,\ k')$ as follows.

$$global(i,\ k') \ = \ \max_{0 \leq k_i \leq k'} \ global(i+1,\ k'-k_i) \ + \ local(P(C_i),\ \mathcal{R}(C_i),\ k_i)$$

3. Return $global(1, 4k)$ as the number of exposed points.

We have the following lemma. (See Section A.2 in the Appendix for a proof.)

▶ **Lemma 11.** *If $P^* \subseteq P$ is the optimal set of exposed points, then $global(1, 4k) \geq |P^*|$, that is , the algorithm* DP-Approx *achieves a 4-approximation and runs in $O(k(n+m)n^2m^2)$ time.*

4.3 Towards a PTAS

We now consider the max-exposure instance in a horizontal strip of unit width. That is, all points in P lie in a horizontal strip bounded by lines ℓ_0, ℓ_1 and $\mathcal{R}$ consists of unit square ranges. Suppose, we subdivide the strip into unit square cells $C_1, C_2, \ldots, C_r \in \mathcal{C}$ ordered from left to right. We make the following simple observation.

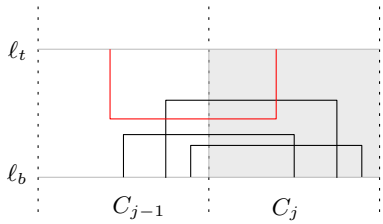


Figure 7 Max-exposure instance in a strip. $C_{j-1}, C_j \in \mathcal{C}$ are two consecutive cells.

▶ **Lemma 12.** *Let $R \in \mathcal{R}$ be a unit square range and C_{j-1} be the first cell from left which it intersects. Then the only other cell that R can intersect is C_j. Moreover, R is Type-1 with respect to C_{j-1} and Type-0 with respect to C_j. (See Figure 7.)*

Observe that the set of points exposed in cell C_j will also depend on the set of Type-0 objects of C_j that were already deleted in C_{j-1}. So we need to ensure that we do not double count the set of ranges that were already deleted in C_{j-1}. To do this, we again use a dynamic

program similar to that for max-exposure within a cell where we express the subproblems at $x = x_i$ in terms of subproblems to the right of $x = x_i$. However, there are some important differences in how we define our subproblems. First, events at a vertical line $x = x_i$ are one of three types:

1. *cell-boundary*: $x = x_i$ is coincident with left-boundary of a cell $C_j \in \mathcal{C}$,
2. *begin-range*: $x = x_i$ is coincident with left-vertical side of a range $R_i \in \mathcal{R}$
3. *point*: $x = x_i$ is incident to an input $p_i \in P$

Moreover for a given cell C_j, in addition to the points q_0, q_1, and ranges Q_0, Q_1, we will also need to remember two additional ranges : L_0 (anchored to ℓ_0) and L_1 (anchored to ℓ_1) that begin in C_{j-1}, were *not deleted* and are farthest from ℓ_0, ℓ_1 respectively. For the sake of clarity, we will use $Z_0 = (q_0, Q_0, L_0)$ to denote the triplets corresponding to ℓ_0 and $Z_1 = (q_1, Q_1, L_1)$ to denote the triplets corresponding to ℓ_1.

Suppose $x = x_i$ lies in the cell C_j. Then we show that the set of deleted ranges $\mathcal{R}_d$ consisting of Type-0 ranges in C_j, and the set of forbidden points P_f can be uniquely identified using the triples Z_0, Z_1.

- *Deleted Type-0 range-set* $\mathcal{R}_d$ Let $\mathcal{R}_{j-1}$ be the set of ranges that begin in cell C_{j-1}, and therefore are Type-1 with respect to C_{j-1}. Suppose we define $\mathcal{L}_{>0} \subseteq \mathcal{R}_{j-1}$ to be the set consisting of ranges anchored to ℓ_0 and farther from ℓ_0 than L_0. Similarly, $\mathcal{L}_{>1} \subseteq \mathcal{R}_{j-1}$ consists of ranges anchored to ℓ_1 and farther from ℓ_1 than L_1. Then, we define $\mathcal{R}_d = (\mathcal{R}(q_0) \cup \mathcal{R}(q_1) \cup \mathcal{L}_{>0} \cup \mathcal{L}_{>1})$.
- *Forbidden point-set* P_f We define $P_f = (P(L_0) \cup P(L_1) \cup P(Q_0) \cup P(Q_1))$.

Finally, we say that a range R *dominates* another range R', if both R, R' begin in the same cell C_j and $R' \cap C_j \subseteq R \cap C_j$. That is, R completely contains the part of R' that lies in cell C_j. Note that the key difference from earlier formulations is that at a begin-range event for a Type-1 range R_i in cell C_j, we choose to *ignore* R_i if it is dominated by ranges Q_0 or Q_1, because the points of R_i contained in C_j already lie in the forbidden set P_f. With $k_i = |\mathcal{R}(p_i) \setminus \mathcal{R}_d|$, we obtain the following recurrence.

$$S(i,\ k',\ Z_0,\ Z_1) \ =\ S(i+1,\ k,\ \mathcal{U}(Z_0, C_j),\ \mathcal{U}(Z_1, C_j))$$

(*event* $x = x_i$ *is left-boundary of cell* C_j)

$$\max \begin{cases} S(i+1,\ k',\ Z_1,\ Z_2) & \text{if } p_i \in P_f, \text{ cannot expose } p_i \\ S(i+1,\ k',\ Z_1,\ Z_2) & \text{otherwise, choose to not expose } p_i \\ S(i+1,\ k'-k_i,\ Z_1,\ Z_2)+1 & \text{otherwise, expose } p_i \end{cases}$$

(*otherwise, event* $x = x_i$ *was a point* p_i *in cell* C_j)

$$\max \begin{cases} S(i+1,\ k',\ Z_0,\ Z_1) & \text{if either } Q_0 \text{ or } Q_1 \text{ dominates } R_i, \textit{ ignore } R_i \\ S(i+1,\ k'-1,\ Z_0,\ Z_1) & \text{otherwise, delete Type-1 range } R_i \\ S(i+1,\ k',\ \mathcal{U}(Z_0, R_i), Z_1) & \text{otherwise if } R_i \text{ is not deleted and anchored to } \ell_0 \\ S(i+1,\ k',\ Z_0,\ \mathcal{U}(Z_1, R_i)) & \text{otherwise, } R_i \text{ is not deleted and anchored to } \ell_1 \end{cases}$$

(*otherwise, event* $x = x_i$ *was beginning of a Type-1 range* R_i *in cell* C_j.)

The function $\mathcal{U}(Z,\ E)$ used above is defined as follows. Roughly speaking, it updates the triplets $Z \in \{Z_0, Z_1\}$ based on the event E and returns an updated triplet. We have the following three cases.

- For a cell-boundary event C_j, if we have $Z_0 = (q_0, Q_0, L_0)$, the function $\mathcal{U}(Z_0, C_j) = (q_0^*,\ Q_0^*,\ Q_0)$. Similarly, $\mathcal{U}(Z_0, C_j) = (q_1^*,\ Q_1^*,\ Q_1)$. This corresponds to resetting the points q_0, q_1, rectangles Q_0, Q_1 for the current cell C_j, and remembering the rectangles L_0, L_1 from the previous cell C_{j-1}.

- For a point event p_i, we have $\mathcal{U}(Z_0, p_i) = (closer(p_i, q_0),\ Q_0,\ L_0)$ and similarly $\mathcal{U}(Z_1, p_i) = (closer(p_i, q_1),\ Q_1,\ L_1)$. Recall that the function $closer(p_i, q_0)$ returns whichever of p_i, q_0 is closer to ℓ_0, and $closer(p_i, q_1)$ returns whichever of p_i, q_1 is closer to ℓ_1.
- Finally for a begin-rectangle event R_i, we have $\mathcal{U}(Z_0, R_i) = (q_0,\ farther(R_i, Q_0),\ L_0)$ and $\mathcal{U}(Z_1, R_i) = (q_1,\ farther(R_i, Q_1),\ L_1)$. Recall that the function $farther(R_i, Q_0)$ returns whichever of R_i, Q_0 is farther from ℓ_0, and $farther(R_i, Q_1)$ returns whichever of R_i, Q_1 is farther from ℓ_1.

The optimal solution is given by $W(0,\ k,\ Z_0^{\emptyset},\ Z_1^{\emptyset})$ where $Z_0^{\emptyset} = (q_0^*,\ Q_0^*,\ Q_0^*)$ and $Z_1^{\emptyset} = (q_1^*,\ Q_1^*,\ Q_1^*)$. In order to establish the correctness of the above formulation, we make the following claim.

▶ **Lemma 13.** *Let $P^* \subseteq P$ be the optimal set of exposed points. Then, for every point $p_i \in P^*$, we count the range $R \in \mathcal{R}(p_i)$ towards the total number of deleted ranges exactly once.*

Proof. We begin by noting that R intersects at most two cells : C_{j-1} as a Type-0 range and C_j as a Type-1 range. It suffices to show that we count R towards the total number of deleted ranges in exactly one of these two cells. Alternatively, it suffices to show that we count R in cell C_j if and only if we have not already counted R in C_{j-1}. Recall that we can only count for R in C_{j-1} by deleting it at a begin-range event. Moreover, we can only count for R in C_j when a point $p_i \notin P_f$ that lies in cell C_j is exposed. Without loss of generality, assume that R is anchored to ℓ_0. The case when R is anchored to ℓ_1 is symmetric.

We first consider the easy case when R was not deleted in C_{j-1}. Observe that since R is Type-0 with respect to C_j, similar to the earlier cases, the terms $\mathcal{R}(q_0) \cup \mathcal{R}(q_1)$ in the expression for $\mathcal{R}_d$ will correctly charge for R in cell C_j.

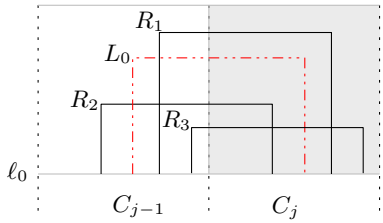


Figure 8 Three cases for the proof: $R_1 \in \mathcal{L}_{>0}$, and $R_2, R_3 \notin \mathcal{L}_{>0}$. R_2 begins before L_0 and R_3 begins after L_0.

Now, we move to the second case where we are currently in cell C_j and we have already counted R by deleting it at a begin-range event in cell C_{j-1}. In this case, we show that we will not count R again in C_j. More precisely, we show that if R contains a point p that lies in cell C_j but *is not contained* in the forbidden point set P_f, then the deleted range set $\mathcal{R}_d$ contains R, and therefore the expression $k_i = \mathcal{R}(p) \setminus \mathcal{R}_d$ will not charge for R again. We have three cases.

1. $R \in \mathcal{L}_{>0}$. This case is straightforward as $\mathcal{R}_d$ contains all ranges in $\mathcal{L}_{>0}$.
2. $R \notin \mathcal{L}_{>0}$ *and* R *begins before* L_0. This case is not possible because any point that is contained in $(R \cap C_j)$ is also contained in L_0. This holds because R and L_0 have the same width, so if R begins before L_0 in C_{j-1}, it must end before L_0 in C_j. Since every point contained in L_0 is contained in the forbidden set P_f, we must have $p \in P_f$ which is a contradiction. (See Figure 8 with $R = R_2$.)
3. $R \notin \mathcal{L}_{>0}$ *and* R *begins after* L_0. This case is also not possible because if this were true L_0 would have dominated R. Therefore, we would have *ignored* R in C_{j-1} and would not have deleted it. (See Figure 8 with $R = R_3$.) ◀

▶ **Lemma 14.** *The restricted max-exposure instance such that all points in P lie within a unit-width horizontal strip bounded by lines ℓ_0, ℓ_1 and $\mathcal{R}$ consists of unit squares can be solved in $O(k(n+m)n^4m^2)$ time, where $m = |P|$ and $n = |\mathcal{R}|$.*

Using similar ideas as Lemma 11, the above lemma readily gives a 2-approximation for max-exposure. More precisely, we can embed the input instance on to a unit-sized grid as before, but instead of solving max-exposure in a cell, we use the above algorithm to solve max-exposure locally in a row of the grid. Since each range $R \in \mathcal{R}$ can intersect at most two rows, R is split into two sub-ranges R_1, R_2 contained in at most two rows. Since these new sub-ranges in two different rows are disjoint, there exists an optimal solution with $2k$ sub-ranges. Therefore, if we have already computed the local solutions for each row i, using the algorithm *DP-Approx* we can compute *global*$(1, 2k)$ which exposes at least optimal number of points using at most $2k$ ranges.

▶ **Corollary 15.** *There exists a 2-approximation algorithm for max-exposure with unit square ranges running in $O(k(n+m)n^4m^2)$ time.*

Generalizing to h anchor lines. The dynamic program for max-exposure in a horizontal strip bounded by two anchor lines ℓ_0, ℓ_1 can be generalized to the case when we have h anchor lines $\ell_1, \ell_2, \ldots, \ell_h$. However, there is a minor technical change required. Observe that for a given anchor line ℓ_i, there can be points and anchored ranges on either side of ℓ_i. Therefore, we will need to remember the closest exposed points and the farthest undeleted ranges on both sides of ℓ_i. So for each anchor line ℓ_i, we will need the triplet $Z_i^+ = (q_i^+, Q_i^+, L_i^+)$ for points and ranges above ℓ_i and the triplet $Z_i^- = (q_i^-, Q_i^-, L_i^-)$ for points and ranges below ℓ_i. The dynamic program will now need to remember at most $4h$ ranges and $2h$ points which gives a running time of $O(k(n+m)n^{4h}m^{2h})$. If we denote a collection of h consecutive anchor lines by a *bundle* of width h, then we have the following.

▶ **Lemma 16.** *Max-exposure in a bundle of width h can be solved in $O(k(n+m)n^{4h}m^{2h})$ time.*

4.4 An $(1+\epsilon)$-Approximation Algorithm

We are now ready to describe our PTAS for the problem. Suppose the anchor lines correspond to the horizontal lines of the uniform unit-sized grid G. Since we have already solved max-exposure exactly for h consecutive rows in G, we can now apply standard shifting techniques [18] to obtain an $(1+\epsilon)$-approximation. If P^* is the optimal set of exposed points, then we show how to compute a set of $(1+\epsilon)k$ ranges deleting which will expose at least $|P^*|$ points. Note that using similar ideas, it is also possible to expose at least $(1-\epsilon)|P^*|$ points by deleting exactly k ranges (See Appendix B).

Suppose that anchor lines $\ell_1, \ell_2, \ldots, \ell_z$ are ordered by increasing y-coordinates. We define a *bundle* B_j to be a set of h consecutive anchor lines, identified by the lowest index anchor ℓ_j. We also define *bundle-set* to be a sequence of consecutive bundles, identified by the index of the lowest bundle. For instance the bundle B_1 comprises of anchor lines ℓ_1 through ℓ_h (inclusive). And the bundle-set $\mathcal{B}_1$ comprises of bundles $B_1, B_h, B_{2h}, \ldots B_{\lceil z/h \rceil}$. The lines $\ell_1, \ell_h, \ldots, \ell_{\lceil z/h \rceil}$ form the *bundle boundaries* $\partial\mathcal{B}_1$ of bundle-set $\mathcal{B}_1$.

For each bundle $B_j \in \mathcal{B}_1$, we can use the dynamic program from Lemma 16 to solve max-exposure locally. Using the exact solution for each bundle as local solution, we can use the algorithm DP-Approx (from Section 4.2) to combine them into a global solution for the

bundle-set $\mathcal{B}_1$ given by $P(\mathcal{B}_1) = \mathit{global}(1, (k + k/h))$. We repeat this for each bundle-set $\mathcal{B}_i$ for all $i \in \{1, 2, \ldots, h\}$, and return the point set $P(\mathcal{B}_i)$ that has maximum cardinality over all $i \in \{1, 2, \ldots, h\}$.

It remains to show that this achieves a good approximation. To see this, we observe that the only ranges that may be double counted are the ones that are anchored to bundle boundaries of $\partial\mathcal{B}_i$. In the following, we show that this number is a small fraction of the optimum solution. (Proof in Appendix A.3.)

▶ **Lemma 17.** *The bundle boundaries $\partial\mathcal{B}_i, \partial\mathcal{B}_j$ for any two bundle-set $\mathcal{B}_i, \mathcal{B}_j$ are disjoint, and therefore the set of ranges anchored to lines in $\partial\mathcal{B}_i$ are also disjoint. Then, there exists a bundle-set $\mathcal{B}_{\min}$ such that the number of ranges of the optimal solution anchored to lines in $\partial\mathcal{B}_{\min}$ is at most k/h.*

Choosing $\epsilon = 1/h$ gives us a set of $(1+\epsilon)k$ objects such that the number of points exposed by selecting these objects is at least the optimum number of points.

▶ **Theorem 18.** *There exists an $(1 + \epsilon)$-approximation algorithm for max-exposure with unit square ranges running in $O(k(n + m)n^{4/\epsilon}m^{2/\epsilon})$ time.*

5 Extensions and Applications

In this section, we discuss some extensions and applications of our the results from previous section. We say that the range family $\mathcal{R}$ consists of *fat rectangles* if every range $R \in \mathcal{R}$ is a rectangle of bounded *aspect ratio*. Moreover, we say that $\mathcal{R}$ consists of *similar and fat rectangles*, if ranges in $\mathcal{R}$ are rectangles and the ratio of the largest to the smallest side in $\mathcal{R}$ is constant. We show that if $\mathcal{R}$ consists of *similar and fat* rectangles, one can achieve a constant approximation. Moreover, if $\mathcal{R}$ consists of *fat rectangles* one can achieve a bicriteria $O(\sqrt{k})$-approximation.

5.1 Approximation for Similar and Fat Rectangles

Let a, b be the length of smallest and largest sides of rectangles in $\mathcal{R}$ such that $b/a = c$ is constant. Then we can modify the input instance as follows. Replace each range $R \in \mathcal{R}$ by tiling it with at most c^2 squares of sidelength a such that the area occupied by R and its replacements are the same. Now, we have a modified set of ranges $\mathcal{R}'$ consisting of squares that have the same sidelength. Consider the optimal solution with k ranges $\mathcal{R}^*$ that exposes m^* points. It is easy to see that the set $\mathcal{R}^*$ corresponds to at most c^2k ranges in the modified instance, and therefore deleting c^2k ranges from $\mathcal{R}'$ exposes at least m^* points. Therefore, we can run the polynomial-time 2-approximation algorithm (Corollary 15) to obtain a set of at most $2c^2k$ ranges that expose at least m^* points.

▶ **Theorem 19.** *Given a set of points P, a set of rectangle ranges $\mathcal{R}$ such that the ratio of largest to smallest side in $\mathcal{R}$ is bounded by a constant, then there exists a polynomial time $O(1)$-approximation algorithm for max-exposure.*

5.2 Approximation for Fat Rectangles

We now consider the case when rectangles in $\mathcal{R}$ have bounded aspect ratio. That is for all rectangles $R \in \mathcal{R}$, the ratio of its two sides is bounded by a constant c. We transform the input ranges $\mathcal{R}$ to obtain a modified set of ranges $\mathcal{R}'$ as follows. For each rectangle $R \in \mathcal{R}$, let x be the length of the smaller side of R. Then we replace R by at most $\lceil c \rceil$ *squares* each of sidelength x. If m^* is the optimal number of points exposed by deleting k ranges from $\mathcal{R}$,

then there exists a set of $O(k)$ ranges in $\mathcal{R}'$ deleting which will expose at least m^* points. Observe that the set $\mathcal{R}'$ consists of *square* ranges, of possibly different sizes. Therefore, if we can obtain an f-approximation for square ranges, we can easily obtain $O(f)$-approximation with fat rectangles.

5.2.1 A Bicriteria $O(\sqrt{k})$-approximation for Squares

We will describe an approximation algorithm for the case when the set of ranges $\mathcal{R}$ consists of axis-aligned squares. We achieve an approximation algorithm in three steps. First, we partition the point set by assigning them to one of the input squares. Next, we solve the problem exactly for a fixed square. Finally, we combine these solutions to achieve a good approximation to the optimal solution.

We define $\mathcal{A} : P \to \mathcal{R}$ to be a function that assigns a point in P to exactly one range in $\mathcal{R}$. If $\mathcal{R}(p_i)$ is the set of squares that contain p_i, then $\mathcal{A}(p_i)$ is the smallest square in $\mathcal{R}(p_i)$. This assignment scheme ensures the following property.

▶ **Lemma 20.** *Let $R \in \mathcal{R}$ be a square and let $P(R) = \mathcal{A}^{-1}(R)$ be the set of points* assigned *to it. Moreover, let $\mathcal{R}' \subseteq \mathcal{R}$ be the set of squares that intersect R and contain at least one point in $P(R)$. Then, every square $R' \in \mathcal{R}'$ must have sidelength bigger than that of R, and therefore contains at least one corner of R.*

Now suppose we fix a square R, and consider a restricted max-exposure instance with the set of its assigned points $P(R)$. Since, ranges that contain a point in $P(R)$ are all bigger then R, this case is essentially the same as points inside a unit square, and therefore Lemma 10 can be easily extended to solve it exactly. This gives us the following algorithm. Here $1 \leq \alpha \leq k$ is a parameter.

Algorithm 3 Greedy-Squares.

1. For every square $R \in \mathcal{R}$, apply Lemma 10 over the point set $P(R)$ to expose the maximum set of points $P(R, k) \subseteq P(R)$ by deleting k ranges.
2. Order squares in $\mathcal{R}$ by decreasing $|P(R, k)|$ values, and pick the set $\mathcal{S} \subseteq \mathcal{R}$ of first α squares. Return $\bigcup_{R \in \mathcal{S}} P(R, k)$ as the set of exposed points.

▶ **Lemma 21.** *Let m^* be the optimal number of points exposed using k squares, then algorithm* Greedy-Squares *computes a set of at most αk squares that expose at least $\alpha m^*/k$ points.*

For $\alpha = \sqrt{k}$, the above algorithm achieves a bicriteria $O(\sqrt{k})$-approximation. Since an f-approximation for square ranges gives an $O(f)$-approximation for fat rectangles, we obtain the following.

▶ **Theorem 22.** *Given a set of points P and a set of ranges $\mathcal{R}$ consisiting of rectangles of bounded aspect ratio, then one can obtain a bicriteria $O(\sqrt{k})$-approximation for max-exposure in polynomial time.*

6 Conclusion

In this paper, we introduced the max-exposure problem over the range space $(P, \mathcal{R})$ and presented approximation algorithms for rectangle range spaces. We showed that the problem is hard to approximate even when $\mathcal{R}$ consists of two types of rectangles, and therefore focused

on the complexity of the problem for the case when $\mathcal{R}$ consists of translates of a single rectangle. We show that in this case, the geometry of ranges can be exploited to obtain a PTAS. A natural question to consider is how does the complexity of the problem change with more general shapes. In particular, does there exist a constant approximation when $\mathcal{R}$ consists of axis-aligned squares?

References

1. P. K Agarwal and J. Pan. Near-linear algorithms for geometric hitting sets and set covers. In *Proceedings of 30th SoCG*, page 271. ACM, 2014.
2. S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. *Journal of computer and system sciences*, 58(1):193–210, 1999.
3. Y. Asahiro, K. Iwama, H. Tamaki, and T. Tokuyama. Greedily finding a dense subgraph. *Journal of Algorithms*, 34(2):203–221, 2000.
4. S. Bandyapadhyay, N. Kumar, S. Suri, and K. Varadarajan. Improved Approximation Bounds for the Minimum Constraint Removal Problem. In *Proceedings of 21st APPROX*, pages 2:1–2:19, 2018.
5. S. Bereg and D. G. Kirkpatrick. Approximating Barrier Resilience in Wireless Sensor Networks. In *Proceedings of 5th ALGOSENSORS*, pages 29–40, 2009.
6. A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph. In *Proceedings of the 42nd STOC*, pages 201–210. ACM, 2010.
7. H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.
8. C. Chekuri, K. L. Clarkson, and S. Har-Peled. On the set multi-cover problem in geometric settings. *ACM Transactions on Algorithms (TALG)*, 9(1):9, 2012.
9. E. Chlamtac, M. Dinitz, C. Konrad, G. Kortsarz, and G. Rabanca. The Densest k-Subhypergraph Problem. In *Proceedings of 19th APPROX*, pages 6:1–6:19, 2016.
10. E. Chlamtáč, M. Dinitz, and Y. Makarychev. Minimizing the union: Tight approximations for small set bipartite vertex expansion. In *Proceedings of the 28th SODA*, pages 881–899, 2017.
11. K. L. Clarkson and P. W. Shor. Application of Random Sampling in Computational Geometry, II. *Discrete & Computational Geometry*, 4:387–421, 1989.
12. M. Cygan, F. Grandoni, S. Leonardi, M. Mucha, M. Pilipczuk, and P. Sankowski. Approximation Algorithms for Union and Intersection Covering Problems. In *Proceedings of 31st FSTTCS*, page 28, 2011.
13. E. Eiben, J. Gemmell, I. Kanj, and A. Youngdahl. Improved results for minimum constraint removal. In *Proceedings of 32nd AAAI Conference on Artificial Intelligence*, 2018.
14. U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.
15. U. Feige, D. Peleg, and G. Kortsarz. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.
16. U. Feige and M. Seltser. On the Densest K-subgraph Problems. Technical report, Weizmann Institute of Science, Jerusalem, Israel, 1997.
17. R. J. Fowler, M. S. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information processing letters*, 12(3):133–137, 1981.
18. D. S. Hochbaum and W Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM (JACM)*, 32(1):130–136, 1985.
19. M. Korman, M. Löffler, R. I. Silveira, and D. Strash. On the complexity of barrier resilience for fat regions and bounded ply. *Comput. Geom.*, 72:34–51, 2018.
20. N. H. Mustafa, R. Raman, and S. Ray. Settling the APX-hardness status for geometric set cover. In *Proceedings of 55th FOCS*, pages 541–550. IEEE, 2014.

A Missing Proofs

A.1 Proof of Lemma 3

Given a graph $G' = (V', E')$ sampled from one of the dense or random instances, we first construct a bipartite graph $G = (A, B, E)$ as follows. For every vertex $v \in V'$, we add a vertex v_a to A and v_b to B. Now for every edge $e = (u, v) \in E'$, we add the pair of edges $e_1 = (u_a, v_b)$ and $e_2 = (v_a, u_b)$ to E. That is, every edge $e \in E'$ is mapped to two copies $e_1, e_2 \in E$ and we can define $par(e_1) = par(e_2) = e$. Similarly, we define $par(u_a) = par(u_b) = u$. We say that G is dense if the underlying graph G' was sampled from the dense case, otherwise we say that G is random.

Consider a set of $k^* = 2k$ vertices in G. If G came from the dense case, there must be a set of $2k$ vertices that have $2k^{\beta+1}$ edges between them. So the number of edges in dense case $m_d^* \geq 2k^{\beta+1}$. Otherwise, we are in the random case. Consider the optimal set of $2k$ vertices V^* and let E^* be the set of edges in the induced subgraph $G[V^*]$. Now consider the corresponding set of vertices $V_p = \{par(v) \mid v \in V^*\}$ of the original graph G' and the set of edges E_p in the induced subgraph $G'[V_p]$). We have that $|V_p| \leq |V^*| = 2k$ and $|E_p| \geq |E^*|/2$ because for each edge $e = (u, v) \in E^*$, we will have the edge $par(e) = (par(u), par(v)) \in E_p$. We can now bound the number of edges E_p over $2k$ vertices in the random case to be $\tilde{O}(\max(2k, 4k^2 n^{\alpha-1}))$ w.h.p, and therefore the optimum number of edges in the random case is $m_r^* = |E^*| \leq 2|E_p| = \tilde{O}(\max(k, k^2 n^{\alpha-1}))$ w.h.p.

Choosing $k = n^{1/2}$, $\alpha = \frac{1}{2}$, $\beta = \frac{1}{2} - \epsilon$, gives us $m_r^* = \tilde{O}(n^{1/2})$ w.h.p. and $m_d^* = \tilde{\Omega}(n^{\frac{3-2\epsilon}{4}})$. Suppose, we could approximate this problem within a factor $O(n^{1/4-\epsilon})$, then in the dense case, the number of edges computed by this approximation algorithm is $\tilde{\Omega}(n^{\frac{1+\epsilon}{2}})$ which is strictly more than the maximum possible edges in the random case. Therefore, we would be able to distinguish between dense and random cases, and thereby refuting the conjecture for these values of α, β and k.

A.2 Proof of Lemma 11

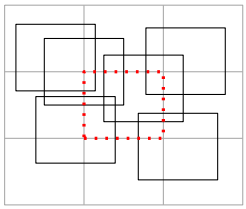


Figure 9 Embedding a max-exposure instance with unit square ranges on a unit-sized grid. Optimal solution in each grid cell can be computed exactly using Lemma 10.

Consider the optimal set of ranges $\mathcal{R}^* \subseteq \mathcal{R}$. Observe that each range $R \in \mathcal{R}^*$ intersects at most four grid cells. Let $R_i = R \cap C_i$ be the rectangular region defined by intersection of R and C_i. Clearly, there are at most four regions R_i for each $R \in \mathcal{R}^*$ and therefore $4k$ in total. At this point, the regions in cell C_i are disjoint from regions in some other cell $C_j \in \mathcal{C}$. Therefore, optimal solution exposes $|P^*|$ points over a set of cells $\mathcal{C}^*$ such that the set $\mathcal{R}^*$ has at most $4k$ disjoint components in the cells $\mathcal{C}^*$. Since we can solve the problem exactly for each cell and can combine them using the above dynamic program, we have that $global(1, 4k) \geq |P^*|$ and we achieve a 4-approximation.

For the running time, we observe that solving max-exposure locally in a cell C_i takes $O(k(n_i+m_i)n_i^2m_i^2)$ time, where n_i is the number of ranges that intersect C_i and m_i is the number of points in P that lie in C_i. Summed over all cells, we get the following bound.

$$\begin{aligned}\sum_i k(n_i+m_i)n_i^2m_i^2 \;&\leq\; k\,\sum_i(n_i+m_i)\,\sum_i n_i^2\,\sum_i m_i^2\\ &\leq k(n+m)\,(\sum_i n_i)^2\,(\sum_i m_i)^2 \;=\; O(k(n+m)n^2m^2)\end{aligned}$$

Once the local solutions are computed, the dynamic program that merges them into a global solution has $O(k|\mathcal{C}|)$ subproblems and computing each subproblem takes $O(k)$ time. Recall that every cell in $\mathcal{C}$ contains at least one point, so $|\mathcal{C}| \leq n$ and the merge step takes an additional $O(k^2n)$ time.

A.3 Proof of Lemma 17

Let $\mathcal{R}^* \subseteq \mathcal{R}$ be the optimal set of ranges, and let $\mathcal{R}_i^* \subseteq \mathcal{R}^*$ be the set of ranges anchored to lines in $\partial\mathcal{B}_i$. Since $\bigcup_{i\in\{1,\ldots h\}} \partial\mathcal{B}_i$ is the set of all anchor lines, we have

$$\begin{aligned}\bigcup_{i\in\{1,\ldots h\}} \mathcal{R}_i^* \;=\; \mathcal{R}^* \qquad &\Longrightarrow \sum_{i\in\{1,\ldots h\}} |\mathcal{R}_i^*| \;=\; k\\ \Longrightarrow \sum_{i\in\{1,\ldots h\}} |\mathcal{R}_{\min}^*| \;\leq\; k \qquad &\Longrightarrow |\mathcal{R}_{\min}^*| \;\leq\; k/h\end{aligned}$$

A.4 Proof of Lemma 21

It is easy to see that the number of squares is at most αk. To show the bound on number of points exposed, consider the optimal solution $\mathcal{R}^*$ and let the optimal set of points exposed by $\mathcal{R}^*$ to be P^*. We will now use the same assignment procedure $\mathcal{A}^* : P^* \to \mathcal{R}^*$ to assign points in P^* to a square in $\mathcal{R}^*$. That is, $\mathcal{A}^*(p_i)$ is the smallest square in $\mathcal{R}^*$ that contains p_i. We claim that $\mathcal{A}^*(p_i) = \mathcal{A}(p_i)$ for all $p_i \in P^*$ since every square that contains p_i lies in $\mathcal{R}^*$. Moreover, let $P^*(R)$ denote the set of points of P^* assigned to R.

Let m^* be the optimal number of points that are exposed, and m' be the number of points exposed by the algorithm. Now assume that the squares in $\mathcal{R}$ are ordered such that $|P(R_i,k)| \geq |P(R_j,k)|$ for all $i<j$. Then, we have the following.

$$\begin{aligned}m^* \;&=\; \left|\bigcup_{R\in\mathcal{R}^*} P^*(R)\right| \;=\; \sum_{R\in\mathcal{R}^*} |P^*(R)|\\ &\leq \sum_{1\leq i\leq k} |P(R_i,k)| \;\leq\; \frac{k}{\alpha}\sum_{1\leq i\leq \alpha} |P(R_i,k)|\\ &= \frac{k}{\alpha}m'\end{aligned}$$

B PTAS for Unit Square Ranges on Number of Exposed Points

Given a set of points P, unit square ranges $\mathcal{R}$, we will now show that the PTAS for unit square ranges can be modified so that we can compute a set of k ranges that expose at least $(1-\epsilon)$ fraction of the maximum possible number of points. For simplicity we assume that h is odd. The basic setup is the same: we have the anchor lines $\ell_1, \ell_2, \ldots, \ell_z$ that

are unit distance apart. However, there is one important change, we will only use the odd-numbered lines $\ell_1, \ell_3, \ldots, \ell_h, \ell_{h+2}, \ldots, \ell_z$ to define bundles. For instance, the bundle B_1 now consists of the anchor lines $\ell_1, \ell_3, \ldots, \ell_h$, while the bundle-set $\mathcal{B}_1$ now comprises of bundles $B_1, B_h, B_{2h}, \ldots, B_{z/h}$. Same as before, the lines $\ell_1, \ell_h, \ldots, \ell_{z/h}$ form the boundary $\partial\mathcal{B}_1$. We have the following algorithm.

Algorithm 4 PTAS-Exposed-Points.

1. Assign each point $p \in P$ to the closest line among $l_1, l_3, \ldots l_z$.
2. For each $i \in \{1, 3, \ldots, h\}$, process bundle set $\mathcal{B}_i$ as follows.
   - Let P_i be the set of points assigned to anchor lines $l_j \in \partial\mathcal{B}_i$, boundaries of $\mathcal{B}_i$.
   - Using the exact algorithm for each bundle $B \in \mathcal{B}_i$ as local solutions, we run the algorithm DP-Approx (from Section 4.2) over the point set $P \setminus P_i$ to obtain global solutions given by $global(1, k)$. Let $P(\mathcal{B}_i)$ be the set of exposed points returned by DP-Approx.
3. Return the set $P(\mathcal{B}_i)$ that has maximum cardinality over all $i \in \{1, 3, \ldots, h\}$.

Clearly, the number of ranges used by the above algorithm is k. It remains to show that the number of points m' exposed by the algorithm is also close to m^*, the optimal number of exposed points. Let $P^* \subseteq P$ be the optimal set of exposed points.

▶ **Lemma 23.** *The bundle boundaries $\partial\mathcal{B}_i, \partial\mathcal{B}_j$ for any two bundle-set $\mathcal{B}_i, \mathcal{B}_j$ are disjoint, and therefore the set of points assigned to lines in $\partial\mathcal{B}_i$ are also disjoint. Then, there exists a bundle-set $\mathcal{B}_{\min}$ such that the number of points of P^* assigned to its boundaries $\partial\mathcal{B}_{\min}$ is at most $\frac{2m^*}{h-1}$.*

Proof. let $P_i^* \subseteq P^*$ be the set of points in P^* that are assigned to lines in boundaries $\partial\mathcal{B}_i$ of some bundle $\mathcal{B}_i$. Since $\bigcup_{i\in\{1,3,\ldots,h\}} \partial\mathcal{B}_i$ is the set of all anchor lines to which we assign points, we have

$$\bigcup_{i\in\{1,3,\ldots h\}} P_i^* = P^* \implies \sum_{i\in\{1,3\ldots h\}} |P_i^*| = m^*$$

$$\implies \sum_{i\in\{1,3,\ldots h\}} |P_{\min}^*| \leq m^* \implies \left(\frac{h-1}{2}\right)|P_{\min}^*| \leq m^*$$

$$\implies |P_{\min}^*| \leq \frac{2m^*}{h-1}$$

◀

Observe that for the bundle-set $\mathcal{B}_{\min}$, we may have removed $P_{\min}$ points, but the remaining set $P \setminus P_{\min}$ consists at least $m^* - \frac{2m^*}{h-1} = (1 - \frac{2}{h-1})m^*$ points of the optimal set P^*. Moreover, observe that we have removed points that are within a unit distance on either side of anchor line $\ell_j \in \partial\mathcal{B}_{\min}$, the set of ranges deleted in each bundle are disjoint from another. Therefore, the value $P(\mathcal{B}_{\min})$ returned by the algorithm DP-Approx exposes at least $P \setminus P_{\min} = (1 - \frac{2}{h-1})m^*$ points by deleting k ranges. If we set $h = 2/\epsilon + 1$ we have the following result.

▶ **Theorem 24.** *There exists an $(1-\epsilon)$-approximation on the number of exposed points for max-exposure with unit-square ranges running in $k(nm)^{O(1/\epsilon)}$ time.*

Small Space Stream Summary for Matroid Center

Sagar Kale
EPFL, Lausanne, Switzerland
sagar.kale@epfl.ch

Abstract

In the matroid center problem, which generalizes the k-center problem, we need to pick a set of centers that is an independent set of a matroid with rank r. We study this problem in streaming, where elements of the ground set arrive in the stream. We first show that any randomized one-pass streaming algorithm that computes a better than Δ-approximation for partition-matroid center must use $\Omega(r^2)$ bits of space, where Δ is the aspect ratio of the metric and can be arbitrarily large. This shows a quadratic separation between matroid center and k-center, for which the Doubling algorithm [7] gives an 8-approximation using $O(k)$-space and one pass. To complement this, we give a one-pass algorithm for matroid center that stores at most $O(r^2 \log(1/\varepsilon)/\varepsilon)$ points (viz., stream summary) among which a $(7+\varepsilon)$-approximate solution exists, which can be found by brute force, or a $(17+\varepsilon)$-approximation can be found with an efficient algorithm. If we are allowed a second pass, we can compute a $(3+\varepsilon)$-approximation efficiently.

We also consider the problem of matroid center with z outliers and give a one-pass algorithm that outputs a set of $O((r^2+rz)\log(1/\varepsilon)/\varepsilon)$ points that contains a $(15+\varepsilon)$-approximate solution. Our techniques extend to knapsack center and knapsack center with z outliers in a straightforward way, and we get algorithms that use space linear in the size of a largest feasible set (as opposed to quadratic space for matroid center).

2012 ACM Subject Classification Theory of computation → Streaming models; Theory of computation → Facility location and clustering; Mathematics of computing → Matroids and greedoids

Keywords and phrases Streaming Algorithms, Matroids, Clustering

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.20

Category APPROX

Related Version A full version of the paper is available at `http://arxiv.org/abs/1810.06267`.

Funding This work was supported by ERC Starting Grant 335288-OptApprox.

Acknowledgements I thank Ashish Chiplunkar for his contributions, Maryam Negahbani for discussions, and anonymous reviewers for helpful comments.

1 Introduction

In the k-center problem, the input is a metric, and we need to select a set of k centers that minimizes the maximum distance between a point and its nearest center. Matroid center is a natural generalization of k-center, where, along with a metric over a set, the input also contains a matroid of rank r over the same set. We then need to choose a set of centers that is an independent set of the matroid that minimizes the maximum distance between a point and its nearest center. Then k-center is rank-k-uniform-matroid center. Examples of clustering problems where the set of centers needs to form an independent set of a partition matroid arise in content distribution networks (see Hajiaghayi et al. [16] and references therein). A partition matroid constraint can also be used to enforce fairness conditions such as having k_M centers of type M and k_W centers of type W. As another example, say the input points lie in a euclidean space, and we are required to output linearly independent centers, then


© Sagar Kale;
licensed under Creative Commons License CC-BY

this is the linear-matroid center problem. Studying a combinatorial optimization problem in the streaming model is worthwhile not only in its own right, but also because it can lead to discovery of much faster algorithms[1].

In the streaming model, the input points arrive in the stream, and we are interested in designing algorithms that use space sublinear in the input size. We study the matroid center problem in the streaming model. By a clean reduction from the INDEX problem, we first show that any randomized one-pass streaming algorithm that computes a better than Δ-approximation for matroid center must use $\Omega(r^2)$ bits of space, where Δ is the aspect ratio of the metric (ratio of the largest distance to the smallest distance between two points), which can be arbitrarily large. Since the Doubling algorithm [7] gives an 8-approximation for k-center in one pass over the stream by storing at most k points, we get a quadratic separation between matroid center and k-center. We then give a one-pass algorithm that computes a $(7+\varepsilon)$-approximation using a *stream summary* of $O(r^2 \log(1/\varepsilon)/\varepsilon)$ points. The algorithm maintains an efficiently-updatable summary, and runs a brute-force step when the end of the stream is reached. We can replace the brute-force step by an efficient algorithm to get a $(17+\varepsilon)$-approximation. Alternatively, using a second pass, we can (efficiently) compute a $(3+\varepsilon)$-approximation. Our algorithms assume only oracle accesses to the metric and to the matroid.

In k-center or matroid center, even very few rogue points can wreck up the solution, which motivates the outlier versions where we can choose up to z points that our solution will not serve. McCutchen and Khuller [28] give a one-pass $(4+\varepsilon)$-approximation algorithm for k-center with z outliers that uses space $O(kz \log(1/\varepsilon)/\varepsilon)$. Building on their ideas, we give a $(15+\varepsilon)$-approximation one-pass algorithm for matroid center with z outliers, using a brute-force search through the summary as the last step, and a $(51+\varepsilon)$-approximation algorithm if we want an efficient implementation in the last step.

To the best of our knowledge, matroid center problems have not been considered in streaming. Chen, Li, Liang, and Wang [11] give an offline 3-approximation algorithm for matroid center and a 7-approximation algorithm for the outlier version; this approximation ratio is improved to 3 by Harris et al. [19]. These algorithms are not easily adaptable to the streaming setting if we are allowed only one pass, though, our two-pass algorithm for matroid center may be thought of as running multiple copies of Chen et al.'s 3-approximation algorithm. We mention that optimization problems over matroid or related constraints have been studied before in streaming [2, 3, 10].

The Doubling algorithm [7] gives an 8-approximation for k-center. Guha [15], using his technique of "stream-strapping", improves this to $2+\varepsilon$. We use the stream-strapping technique in this paper to reduce space-usage of our algorithms as well. Known streaming algorithms for k-center problems do not extend to the matroid center problems. Indeed, the gap between the space complexities of k-center and matroid center, exhibited by our lower bound, warrants the need for new ideas.

[1] This is demonstrated by Chakrabarti and Kale [3] who give streaming algorithms for submodular maximization problems that make *only* $2|E|$ total submodular-oracle calls ($\tilde{O}(|E|)$ total time) and achieve constant-factor approximations, where E is the ground set. On the other hand earlier fastest algorithms were greedy and potentially could make $\Omega(|E|^2)$ oracle calls. Trivially, $|E|$ oracle calls are needed for any non-trivial approximation.

1.1 Techniques

At the heart of many algorithms for k-center is Gonzalez's [13] furthest point heuristic that gives a 2-approximation. It first chooses an arbitrary point and adds it to the current set C of centers. Then it chooses a point that is farthest from C and adds it to C. This is repeated until C has k centers. Let C_E be the set of centers returned by this algorithm, and let p be the point that is farthest from C_E. Then $d(p, C_E)$ is the cost of the solution, whereas the set $C_E \cup \{p\}$ of size $k+1$ acts as a certificate that an optimum solution must have cost at least $d(p, C_E)/2$. This can be easily implemented in streaming if we are given a "guess" τ of OPT, i.e., the cost of an optimum solution. When we see a new point e in the stream, we add it to C if $d(e, C) > 2\tau$. Assuming that we know the aspect ratio Δ, we can do this for $2\log_{1+\varepsilon}\Delta$ guesses of OPT to get a $(2+\varepsilon)$-approximation as follows. Let R be the distance between first two points in the stream. Then maintain the set C as described above for guesses $\tau \in \{R/\Delta, (1+\varepsilon)R/\Delta, (1+\varepsilon)^2 R/\Delta, \ldots, R\Delta\}$. The stream-strapping technique reduces the number of active guesses to $O(\log(1/\varepsilon)/\varepsilon)$.

In extending this to matroid center, the biggest challenge is deciding which point to make a center. In a solution to k-center, if we replace a point by another point that is very close to it, then the cost can change only slightly, whereas if we do the same in a solution to matroid center, the solution might just become infeasible. Therefore, if we maintain a set C as earlier, it might quickly lose its independence in the matroid. The idea is to store, for each of the at most r points $c \in C$, a maximal independent set I_c of points close to c; here, by close we mean close in terms of the guess τ. This way, we store at most $r^2 + r$ points. Storing a maximal independent set for each point in C may seem wasteful, but our lower bound shows that it is necessary. Our first algorithmic insight is to show that this idea works for a correct guess. We show that if each optimum center s is in the span of an independent set I_c for a c that is close to s, then we can recover an independent set of small cost from the *summary* $\bigcup_{c \in C_E} I_c$. And as our second insight, we show how to extend the stream-strapping approach to reduce the number of active guesses, which helps us reduce the space usage. These ideas naturally combine with those of McCutchen and Khuller [28] and help us design an algorithm for matroid center with z outliers, but it is nontrivial to prove that the combination of these ideas works.

Knapsack center

In the knapsack center problem, each point e has a non-negative weight $w(e)$, and the goal is to select a set C of centers that minimizes the maximum distance between a point and its nearest center subject to the constraint that $\sum_{c \in C} w(c) \leqslant B$, where B is the *budget*. The k-center problem is a special case with unit weights and $B = k$. In the streaming setting, our algorithms for matroid center and matroid center with outliers can be extended to get constant approximations using space proportional to the size of a largest feasible set, i.e., $\max\{|S| : \sum_{e \in S} w(e) \leqslant B\}$. As described earlier, we maintain a set C of potential centers using the guess τ, and for each potential center c, we also maintain a smallest weight point, say s_c, in its vicinity. Then, in the end, the summary $\{s_c : c \in C\}$ contains a good solution. This idea works because replacing a center by a nearby point with a smaller weight does not affect the feasibility in the knapsack setting (which could destroy independence in the matroid setting).

1.2 Related Work

The k-center problem was considered in the '60s [17, 18]. It is NP-hard to achieve a factor of better than 2 [23], and polynomial-time 2-approximation algorithms exist [13, 21]. As mentioned earlier, Chen et al. [11] give a 3-approximation algorithm for matroid center

and a 7-approximation algorithm for the outlier version, and this approximation ratio is improved to 3 by Harris et al. [19]. Motivated by applications in content distribution networks, the matroid median problem is considered as well [16, 25]. The problem of k-center with outliers was first studied by Charikar et al. [8] who gave a 3-approximation algorithm. The approximation ratio was recently improved to 2 by Chakrabarty et al. [4]. We mention the work of Lattanzi et al. [26] that considers hierarchical k-center with outliers.

For knapsack center, a 3-approximation was given by Hochbaum and Shmoys [22]. For the outlier version of knapsack center, very recently, Chakrabarty and Negahbani [5] gave the first non-trivial approximation (a 3-approximation).

Streaming

Charikar et al. [9] and Guha et al. [14] consider k-median with and without outliers in streaming. Guha [15] gives a $(2+\varepsilon)$-approximation one-pass algorithm for k-center that uses $O(k\log(1/\varepsilon)/\varepsilon)$ space, and McCutchen and Khuller [28] give a $(4+\varepsilon)$-approximation one-pass algorithm for k-center with z outliers that uses $O(kz\log(1/\varepsilon)/\varepsilon)$ space. The special cases of 1-center (or, the minimum enclosing ball problem) and 2-center in euclidean spaces have been considered [29, 24, 20] and better approximation ratios than the general k-center problem are known in streaming. Correlation clustering is studied in streaming by Ahn et al. [1]. Cohen-Addad et al. [12] give streaming algorithms for k-center in the sliding windows model, where we want to maintain a solution for only some number of the most recent points in the stream. Guha [15] also gives a space lower bound of $\Omega(n)$ for one-pass algorithms that give a better than 2 approximation for (even the special case of) 1-center by a simple reduction from INDEX, where n is the number of points.

k-center in different models

Chan et al. [6] consider k-center in the fully dynamic adversarial setting, where points can be added or deleted from the input, and the goal is to always maintain a solution by processing the input updates quickly. Malkomes et al. [27] study distributed k-center with outliers.

1.3 Organization of the Paper

We define the model and the problems in Section 2. Section 3 is on the lower bound. In Section 4, we give our important algorithmic ideas and discuss our algorithm for matroid center, and then in Section 5, we discuss the outlier version. In Appendix A, we give the improved space bounds.

2 Preliminaries

A matroid $\mathcal{M}$ is a pair $(E,\mathcal{I})$, where E is a finite set and is called the ground set of the matroid, and $\mathcal{I}$ is a collection of subsets of E that satisfies the following *axioms*:

1. $\emptyset \in \mathcal{I}$,
2. if $J \in \mathcal{I}$ and $I \subseteq J$, then $I \in \mathcal{I}$, and
3. if $I, J \in \mathcal{I}$ and $|I| < |J|$, then there exists $e \in J \setminus I$ such that $I \cup \{e\} \in \mathcal{I}$.

If a set $A \subseteq E$ is in $\mathcal{I}$, then it is called an *independent* set of the matroid $\mathcal{M}$, otherwise it is called a *dependent set*. A singleton dependent set is called a *loop*. *Rank* of a set A, denoted by $\text{rank}(A)$, is the size of a maximal independent set within A; note that rank is a well-defined function because of the third axiom, which is called the *exchange* axiom. Clearly,

for $A \subseteq B$, $\text{rank}(A) \leqslant \text{rank}(B)$. Rank of a matroid is the size of a maximal independent set within E. *Span* of a set A, denoted by $\text{span}(A)$, is the largest set that contains A and has the same rank as A (it can be shown that such a set is unique). We will also use *submodularity* of the rank function, i.e., for $A, B \subseteq E$,

$$\text{rank}(A \cup B) + \text{rank}(A \cap B) \leqslant \text{rank}(A) + \text{rank}(B)\,. \tag{1}$$

A matroid $(E, \mathcal{I})$ is a *partition* matroid if there exists a partition $\{E_1, E_2, \ldots, E_p\}$ of E and nonnegative integers $\ell_1, \ell_2, \ldots, \ell_p$, such that $\mathcal{I} = \{A \subseteq E : \forall i \in [p], |A \cap E_i| \leqslant \ell_i\}$. We say that ℓ_i is the *capacity* of part E_i. Observe that the rank of the matroid is $\sum_{i=1}^{p} \ell_i$.

A metric d over E is a (distance) function $d : E \times E \to \mathbb{R}_+$ that satisfies the following properties for all $e_1, e_2, e_3 \in E$:

1. $d(e_1, e_2) = 0$ if and only if $e_1 = e_2$,
2. $d(e_1, e_2) = d(e_2, e_1)$, and
3. $d(e_1, e_3) \leqslant d(e_1, e_2) + d(e_2, e_3)$; this property is called the *triangle inequality*.

We sometimes call elements in E points. For a point e and a positive number α, the closed ball of radius α around e, denoted by $\mathfrak{B}(e, \alpha)$, is the set $\{x \in E : d(e, x) \leqslant \alpha\}$. We overload d by defining $d(e, A) := \min_{x \in A} d(e, x)$ for $e \in E$ and $A \subseteq E$. The aspect ratio Δ of a metric is the ratio of the largest distance to the smallest in the metric, i.e., $\max_{x,y} d(x, y) / \min_{x,y} d(x, y)$.

The input for the matroid center problem is a matroid $\mathcal{M} = (E, \mathcal{I})$ of rank r and a metric d over E. The goal is to output an independent set S such that its cost $\max_{e \in E} d(e, S)$ is minimized. We are interested in algorithms that assume oracle (or black-box) accesses to the matroid and the metric. The algorithm can ask the matroid oracle whether a set is independent or not, and it can ask the metric oracle (or distance oracle) what the distance between given two points is. In the streaming model, elements of E arrive one by one, and we want to design an algorithm that uses small (sublinear in the input) space. The algorithm can query the oracles only with the elements of E. If the algorithm queries an oracle with an element not in E, then we say that it *fails*. A streaming algorithm can only remember a small part of the input, and the aforementioned restriction disallows plausible learning about forgotten elements indirectly from oracle calls. Also, an algorithm cannot just enumerate elements of E on the fly without looking at the stream, because it does not know the names of the elements in advance.

The input for matroid center with z outliers is also a matroid $\mathcal{M} = (E, \mathcal{I})$ and a metric d over E, but the goal is to output an independent set whose cost is computed with respect to $|E| - z$ closest points. Formally, cost of a set S is $\min\{\alpha \in \mathbb{R}_+ : |E \setminus (\bigcup_{s \in S} \mathfrak{B}(s, \alpha))| \leqslant z\}$.

We denote by OPT the cost of an optimum solution of the instance in the context.

3 Space Lower Bound for One Pass Matroid Center

We show that $\Omega(r^2)$ space is required to achieve better than Δ-approximation for a one-pass algorithm for matroid center. We reduce from the communication problem of INDEX. This reduction is based on the simple reduction for the maximum-matching-size problem: see Figure 1. In INDEX_N, Alice holds an N-bit string and Bob holds an index $I \in [N]$; Alice sends a message to Bob, who has to determine the bit at position I. It is known that Alice has to send a message of size at least $(1 - H_2(3/4))N \geqslant 2N/11$ for Bob to output correctly with a success probability of $3/4$, where H_2 is the binary entropy function.

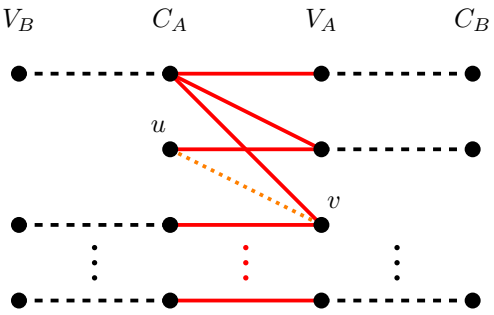


Figure 1 If we have a one-pass streaming algorithm that computes the size of a maximum matching of a k vertex bipartite graph using $o(k^2)$ space, then we can solve INDEX_N using $o(N)$ communication, which would be a contradiction. Alice and Bob agree on a bijection from $[N]$ to the edges of a complete bipartite graph $K_{k,k}$ and construct a graph G as follows. If ℓth bit is 1, Alice adds the corresponding edge (shown in solid red). If the index corresponds to the edge $\{u, v\}$ (shown as a dotted orange edge), Bob adds a new perfect matching between all but vertices u and v and $2k-2$ new vertices (shown as dashed black edges). Alice runs the matching-size estimation algorithm and sends the memory contents to Bob, who continues running it and computes the output. By design, if the index is 1, then maximum-matching-size is $2k-1$, otherwise it is $2k-2$, and an exact algorithm can distinguish between the two cases.

3.1 Reduction from Index to Partition-Matroid Center

We prove the following theorem.

▶ **Theorem 1.** *Any one-pass algorithm for partition-matroid center that outputs a better than Δ-approximation with probability at least $3/4$ must use at least $r^2/24$ bits of space.*

Proof. Assume, towards a contradiction, that there exists a one-pass algorithm for partition-matroid center that outputs a better than Δ-approximation using at most $r^2/24$ bits of space. Then we use it to solve the INDEX problem. Given an input for INDEX, Alice and Bob first construct a bipartite graph G just as described in Figure 1. Then they construct a partition-matroid center instance based on G. Before formalizing the construction, we emphasize that the metric does not correspond to the graph metric given by G, but each edge in G will become a point in the metric. The vertex set they use is union of four sets C_A, V_A, each of size q, and C_B, V_B, each of size $q-1$. Alice constructs a subset of edges between C_A and V_A based on her N-bit string, so we use $N = q^2$. We say that these edges are owned by Alice. If the index that Bob holds corresponds to an edge $\{u, v\}$ with $u \in C_A$ and $v \in V_A$, he adds a perfect matching M between $C_A \setminus \{u\}$ and V_B and a perfect matching M' between $V_A \setminus \{v\}$ and C_B. The edges in $M \cup M'$ are owned by Bob.

To each $u \in C_A \cup C_B$, we associate a cluster $C(u)$ of at most q points in the metric that we will construct, and to each $v \in V_A \cup V_B$, we associate a part $P(v)$ in the partition matroid with capacity 1. Thus, rank of the matroid $r = 2q-1$ because $|V_A \cup V_B| = 2q-1$. By our design, no two clusters will intersect and no two parts will intersect, i.e., $C(u) \cap C(u') = \emptyset$ for $u \neq u'$, and $P(v) \cap P(v') = \emptyset$ for $v \neq v'$. The metric is as follows. Any two points in the same cluster are a unit distance apart and any two points in two different clusters are distance Δ apart. This trivially forms a metric, because the clusters are disjoint. For each $u \in C_A$, Bob adds a point $p(u)$ in the cluster $C(u)$, so that it is nonempty. Add $P' := \{p(u) : u \in C_A\}$ as a part in the partition matroid with capacity 0, so no $p(u)$ can be a center. For each edge $\{u, v\}$ in G with $u \in C_A \cup C_B$ and $v \in V_A \cup V_B$, whoever owns that edge adds a point $p(\{u, v\})$ that goes in cluster $C(u)$ and part $P(v)$. Now, Alice runs the partition-matroid

center algorithm on the points she constructed. She can do this because she knows the metric and the part identity of each point, so she can simulate the distance and matroid oracles. Note that if the algorithm expects an explicit description of the partition matroid, Alice can also send along with each point the identity of the part to which it belongs and the capacity of the part (which is always 1 for her points). She then sends the memory contents to Bob, who continues running the algorithm on his points and computes the cost of the output. We note that Bob can also simulate the distance and matroid oracles. Any point he does not own corresponds to a red edge, and using the identity of that edge, he can figure out the part and cluster to which the point belongs.

Now we prove the correctness of the reduction. Say Bob holds the index corresponding to the edge $\{u, v\}$, where $u \in C_A$ and $v \in V_A$. If the index is 1, then $\{u, v\}$ exists in the graph, then opening centers at points corresponding to edges in $M \cup M' \cup \{u, v\}$ satisfies the partition matroid constraint and also for each $u \in C_A \cup C_B$, we have a center opened in $C(u)$, so the cost is 1. Let the index be 0. We want to show that there is no independent set of cost less than Δ. For a contradiction, assume there is such an independent set. Now, recall that $p(u)$ cannot be a center, so it has to be served by some center in $C(u)$, otherwise the cost will be Δ. Let $p(u)$ be served by some $p(\{u, v'\})$ for $v' \neq v$. Then $p(\{v', w\})$, where $\{v', w\} \in M'$, cannot be a center, because both $p(\{u, v'\}$ and $p(\{v', w\})$ belong to the part $P(v')$ with capacity 1. The point $p(\{v', w\})$ is the lone point in its cluster, and since it cannot be a center, the cost is Δ. If the algorithm is better than Δ-approximation, then Bob can distinguish between these two cases, and thus, solve INDEX_N using communication at most $r^2/24 \leqslant 4q^2/24 = N/6$ bits, which is a contradiction. ◀

After seeing the lower bound, a remark is in order. The difficulty in designing an algorithm is as follows. Even if we know that one center must lie in a ball of small radius centered at a known point, we do not know which points in that ball to store so as to recover an independent set of the matroid.

4 Matroid Center

Our algorithm for matroid center can be seen as a generalization of the algorithm by Hochbaum and Shmoys for k-center [21] adapted to the streaming setting. We first quickly describe the algorithm for k-center. Given an upper bound τ on the optimum cost, the algorithm stores a set C of up to k *pivots* such that distance between any two pivots is more than 2τ. When the algorithm sees a new point e in the stream such that distance between e and any pivot is more than 2τ, it makes e a pivot. The size of C cannot exceed k in this way, because τ is an upper bound on the optimum cost, so no two pivots are served by a single optimum center. Also, any other point is within distance 2τ of some pivot. In the end, the algorithm designates all pivots as centers. In generalizing this to matroid center, one obvious issue is that the set C of pivots constructed as above may not be an independent set for the given general matroid[2]. What we do know is that there has to be an optimum center within distance τ of each pivot. Formally, for $c \in C$, there exists s_c such that $d(c, s_c) \leqslant \tau$ and $\{s_c : c \in C\}$ is an independent set. For each pivot c, we maintain an independent set I_c of nearby points. We prove that it is enough to have each s_c be spanned by some $I_{c'}$ to get a good solution within $\bigcup_{c \in C} I_c$. Algorithm 1 gives a formal description.

[2] This is precisely why we call points in C "pivots" rather than "centers" in this paper.

Note that in Algorithm 1 if we try to add e to I_c under the condition that $d(e,c) \leqslant \tau$, then we may miss spanning some s_c. This will happen if $d(s_c, C) \in (\tau, 2\tau]$, where C is the set of pivots when s_c arrived. Using the condition $d(e,c) \leqslant \tau$ works if each s_c arrives after c though (we use it in the second pass of our two-pass algorithm).

Algorithm 1 One pass algorithm for matroid center.

```
1:  function MatroidCenter(τ,flag)
2:      Initialize pivot-set C ← ∅.
3:      for each point e in the stream do
4:          if there is a pivot c ∈ C such that d(e,c) ⩽ 2τ (pick arbitrary such c) then
5:              if I_c ∪ {e} is independent then
6:                  I_c ← I_c ∪ {e}.
7:          else if |C| = r then                ▷ We cannot have more pivots than the rank.
8:              Abort.      ▷ Because C ∪ {e} acts as a certificate that the guess is incorrect.
9:          else
10:             C ← C ∪ {e}.                                          ▷ Make e a pivot.
11:             If {e} is not a loop, I_e ← {e}, else I_e ← ∅.
12:     if flag = "brute force" then
13:         Find an independent set C'_B in ⋃_{c∈C} I_c such that d(c, C'_B) ⩽ 5τ for c ∈ C.
14:         If such C'_B does not exist, then abort, else return C'_B.
15:     return EfficientMatroidCenter(5τ, C, (I_c)_{c∈C}, M) (given in Algorithm 6
        in Appendix B).
```

First, we quickly bound the space usage.

▶ **Lemma 2.** *In any call to* MatroidCenter*, we store at most* $r^2 + r$ *points.*

Proof. The check on Line 7 ensures that $|C| \leqslant r$. For each pivot c, the size of its independent set I_c is at most r, hence the total number of points stored is at most $r^2 + r$. ◀

Consider a call to MatroidCenter with $\tau \geqslant \mathrm{OPT}$. Let C_E be the set of pivots at the end of the stream. As alluded to earlier, for an optimum independent set I^*, the following holds: for each $c \in C_E$, there exists $s_c \in I^*$ such that $d(c, s_c) \leqslant \tau$, and also $s_c \neq s_{c'}$ for $c \neq c'$, because $d(c, c') > 2\tau$. Now, we prove the following structural lemma that we need later.

▶ **Lemma 3.** *Let* $I_1, \ldots, I_t$ *and* $S = \{s_1, \ldots, s_u\}$ *be independent sets of a matroid such that there is an onto function* $f : [u] \to [t]$ *with the property that* s_i *is in the span of* $I_{f(i)}$ *for* $i \in [u]$*. Then there exists an independent set* B *such that* $|B \cap I_j| \geqslant 1$ *for* $j \in [t]$*.*

Proof. For each $\ell \in \{0, 1, \ldots, u\}$, we construct an independent set S_ℓ such that $|S_\ell| = u$, $|S_\ell \cap I_{f(j)}| \geqslant 1$ for $j \leqslant \ell$, and $s_{\ell+1}, \ldots, s_u \in S_\ell$, then S_u is our desired set B. Start with $S_0 = S$, and assume that we have constructed $S_0, S_1, \ldots, S_{\ell-1}$. If $s_\ell \in I_{f(\ell)}$, we are done, so let $s_\ell \notin I_{f(\ell)}$, then we claim that $\mathrm{rank}((S_{\ell-1} \setminus \{s_\ell\}) \cup I_{f(\ell)}) \geqslant u$. To see this, observe that $\mathrm{rank}(S_{\ell-1}) = u$, so by monotonicity of the rank function, $\mathrm{rank}(S_{\ell-1} \cup I_{f(\ell)}) \geqslant u$, but $s_\ell \in \mathrm{span}(I_{f(\ell)})$, so removing s_ℓ from $S_{\ell-1} \cup I_{f(\ell)}$ would not reduce its rank. We now give a formal argument for completeness. We have $(I_{f(\ell)} \cup \{s_\ell\}) \cup ((S_{\ell-1} \setminus \{s_\ell\}) \cup I_{f(\ell)}) = S_{\ell-1} \cup I_{f(\ell)}$, and $(I_{f(\ell)} \cup \{s_\ell\}) \cap ((S_{\ell-1} \setminus \{s_\ell\}) \cup I_{f(\ell)}) = I_{f(\ell)}$. By submodularity of the rank function (see (1) in Section 2), we have

$$\mathrm{rank}(S_{\ell-1} \cup I_{f(\ell)}) + \mathrm{rank}(I_{f(\ell)}) \leqslant \mathrm{rank}(I_{f(\ell)} \cup \{s_\ell\}) + \mathrm{rank}((S_{\ell-1} \setminus \{s_\ell\}) \cup I_{f(\ell)}) .$$

Let $q = \text{rank}(I_{f(\ell)})$. Since $s_\ell \in \text{span}(I_{f(\ell)})$, we have $\text{rank}(I_{f(\ell)} \cup \{s_\ell\}) = q$ and the above inequality gives

$$u + q \leqslant q + \text{rank}((S_{\ell-1} \setminus \{s_\ell\}) \cup I_{f(\ell)}),$$

which proves the claim. Now, $\text{rank}(S_{\ell-1} \setminus \{s_\ell\}) = u - 1 < \text{rank}((S_{\ell-1} \setminus \{s_\ell\}) \cup I_{f(\ell)})$, therefore there exists $a \in I_{f(\ell)}$ such that $S_\ell := (S_{\ell-1} \setminus \{s_\ell\}) \cup \{a\}$ is independent by the exchange axiom. ◀

▶ **Lemma 4** (Small stream summary for matroid center). *Consider a call to* MatroidCenter *with* $\tau \geqslant \text{OPT}$. *Then there exists an independent set* $B \subseteq \bigcup_{c \in C_E} I_c$ *such that* $d(e, B) \leqslant 7\tau$ *for any point* e *and* $d(c, B) \leqslant 5\tau$ *for any pivot* $c \in C_E$.

Proof. For $c \in C_E$, denote by s_c the optimum center that serves it, so $d(c, s_c) \leqslant \tau$. Let $c' \in C_E$ be such that we tried to add s_c to $I_{c'}$ either on Line 6 or on Line 11; note that c' may not be the same as c if we added it on Line 6. For an $x \in I^*$, let $a(x) \in C_E$ denote the pivot whose independent set $I_{a(x)}$ we tried to add x to. Either we succeeded, in which case $x \in I_{a(x)}$, or we failed, in which case $x \in \text{span}(I_{a(x)})$. In any case, by Lemma 3, for $\mathcal{A} := \{I_{a(x)} : x \in I^*\}$ there exists an independent set B such that $|I \cap B| \geqslant 1$ for all $I \in \mathcal{A}$.

Now, we will bound the cost of B. See Figure 2. Consider any point e in the stream. Let

- $c(e) \in C_E$ be such that $d(e, c(e)) \leqslant 2\tau$,
- $s_{c(e)}$ be the optimum center that serves $c(e)$, so $d(c(e), s_{c(e)}) \leqslant \tau$,
- $a(s_{c(e)}) \in C_E$ be the pivot whose independent set we tried to add $s_{c(e)}$, so $d(s_{c(e)}, a(s_{c(e)}))$ is at most 2τ,
- $c'(e)$ be an arbitrary point in $I_{a(s_{c(e)})} \cap B$, so $d(a(s_{c(e)}), c'(e)) \leqslant 2\tau$ because $c'(e) \in I_{a(s_{c(e)})}$.

Then by triangle inequality, $d(e, B)$ is at most

$$d(e, c'(e)) \leqslant d(e, c(e)) + d(c(e), s_{c(e)}) + d(s_{c(e)}, a(s_{c(e)})) + d(a(s_{c(e)}), c'(e)) \leqslant 2\tau + \tau + 2\tau + 2\tau,$$

which is 7τ; this proves the first part of the lemma.

For any $c \in C_E$, we can bound $d(c, B)$ in a similar way. Let s_c be the optimum center that serves c, and similarly define $a(s_c)$ to be the pivot such that $d(s_c, a(s_c)) \leqslant 2\tau$. Also, let c' be the point in B such that $d(a_{s_c}, c') \leqslant 2\tau$. This gives that $d(c, B) \leqslant d(c, c') \leqslant 5\tau$. ◀

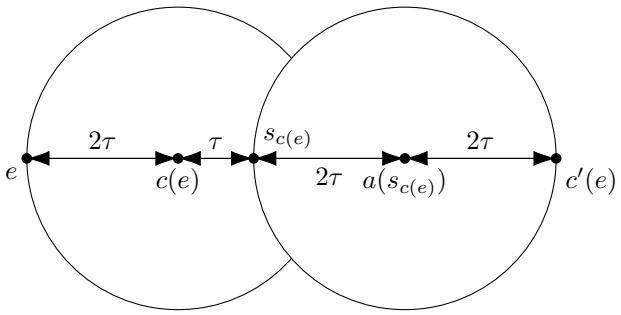


Figure 2 To see how to bound the cost of the independent set B, let e be any point in the stream, $c(e) \in C_E$ be the pivot close to e, $s_{c(e)}$ be the optimum center that covers $c(e)$, $a(s_{c(e)}) \in C_E$ be the pivot close to $s_{c(e)}$, and $c'(e)$ be a point in B that covers $a(s_{c(e)})$.

Before proving our main theorem, we need the following guarantee on the efficient offline 3-approximation algorithm denoted by EfficientMatroidCenter. This algorithm is based on the offline algorithm for matroid center by Chen et al. [11]. We give it as input $\alpha = 5\tau$,

the set C_E of pivots, their independent sets $(I_c)_{c\in C_E}$, and the underlying matroid $\mathcal{M}$ with the promise on the input that there is an independent set $B \subseteq \bigcup_{c\in C_E} I_c$ such that for $c \in C_E$, it holds that $d(c, B) \leqslant 5\tau = \alpha$.

▶ **Theorem 5.** *If* EfficientMatroidCenter *does not fail, then it outputs a set C' such that $d(c, C') \leqslant 3\alpha$ for each $c \in C_E$. If the input promise holds, then* EfficientMatroidCenter *does not fail.*

Proof. This theorem is proved as Theorem 22 in the appendix. See Appendix B. ◀

Now we prove the main result.

▶ **Theorem 6.** *There is an efficient $(17 + \varepsilon)$-approximation one-pass algorithm for matroid center that stores at most $2(r^2 + r) \log_{(1+\varepsilon/17)} \Delta$ points. With a brute force algorithm, one can get a $(7 + \varepsilon)$-approximation.*

Proof. The algorithm is as follows. Let δ be the distance between the first two points. Then for $2 \log_{1+\varepsilon/17} \Delta$ guesses τ of OPT starting from δ/Δ to $\delta\Delta$, we run MatroidCenter(τ, flag). We return the set of centers returned by the instance corresponding to the smallest guess τ. Lemma 2 gives the desired space bound.

Case 1. flag = "brute force".

Suppose the algorithm returned C'_B. Lemma 4 guarantees that for $\tau \in [\mathrm{OPT}, (1 + \varepsilon/17)\,\mathrm{OPT})$, the algorithm will not abort. Then, by the check on Line 14, cost of C'_B is at most $7\tau \leqslant (7 + \varepsilon)\,\mathrm{OPT}$.

Case 2. flag = "efficient algorithm".

Let the algorithm returned C'. Theorem 5 guarantees that for $\tau \in [\mathrm{OPT}, (1+\varepsilon/17)\,\mathrm{OPT})$, the algorithm will not abort. By Theorem 5 for any $c \in C_E$, we have $d(c, C') \leqslant 15\tau$. Since we forget only the points within distance 2τ of C_E, we get that for any point e in the stream, $d(e, C') \leqslant 17\tau \leqslant (17 + \varepsilon)\,\mathrm{OPT}$. ◀

We make some remarks.

▶ Remark 7. We do need to know the rank of the matroid (or an upper bound), otherwise we cannot control the space usage. The instances run using a very small guess may store a very large number of pivots without the check on Line 7.

▶ Remark 8. We can decrease the space usage to $O(r^2 \log(1/\varepsilon)/\varepsilon)$ points using the parallelization ideas of Guha [15]. To make the ideas work, we do need some properties of matroids. We give the details in Appendix A.

▶ Remark 9. By running $\binom{|E|}{2}$ guesses, EfficientMatroidCenter can be used to get an offline 3-approximation algorithm for a more general version of matroid center, where the cost is computed with respect to a subset C_E of E and any point in E can be a center.

4.1 Extension to Knapsack Center

Recall that in the knapsack center problem, each point e has a non-negative weight $w(e)$, and the goal is to select a set C of centers that minimizes the maximum distance between a point and its nearest center subject to the constraint that $\sum_{c\in C} w(c) \leqslant B$, where B is the *budget*. We modify Algorithm 1 slightly to give an algorithm for knapsack center using space r factor smaller than the matroid case, where, in this case, r is the size of a largest feasible set. We make sure that all I_c variables are singletons, so the algorithm stores at most $2r$ points. Instead of the if condition on Line 5, we replace the point x in I_c by e if $w(x) > w(e)$.

This idea works because replacing a point by a nearby point with a smaller weight does not affect the feasibility in the knapsack setting (which could destroy independence in the matroid setting). Let C_E be the set of pivots at the end of the stream. By almost the same argument as in the proof of Lemma 4, we get the following.

▶ **Lemma 10.** *Let $\tau \geqslant \mathrm{OPT}$. Then there exists a feasible set $K \subseteq \bigcup_{c \in C_E} I_c$ such that $d(e, K) \leqslant 7\tau$ for any point e and $d(c, K) \leqslant 5\tau$ for any pivot $c \in C_E$.*

For the efficient version, we then use the 3-approximation algorithm by Hochbaum and Shmoys [22].

▶ **Theorem 11.** *There is an efficient $(17+\varepsilon)$-approximation one-pass algorithm for knapsack center that stores at most $4r \log_{(1+\varepsilon/17)} \Delta$ points, where r is the size of a largest feasible set. With a brute force algorithm, one can get a $(7+\varepsilon)$-approximation.*

4.2 An Efficient Two Pass Algorithm

This algorithm is a streaming two-pass simulation of the offline 3-approximation algorithm of Chen et al. [11] for matroid center. We describe the algorithm and give the analysis below.

In our one-pass algorithm, i.e. Algorithm 1, say we are promised that for any pivot c, the optimum center that serves it appears after c. Then it is enough to try to add e to I_c whenever $d(e, c) \leqslant \tau$; we call this a modified check. Let C_E be the set of pivots in the end, then $(I_c)_{c \in C_E}$ form a partition such that if we pick one point from each I_c to get set B, we can serve each point in C_E using B with cost at most τ. With the modified check, for $c, c' \in C_E$ such that $c \neq c'$, the optimum points s_c and $s_{c'}$ that serve them are also different because $d(c, c') > 2\tau$. Now, $s_c \in \mathrm{span}(I_c)$ due to the promise that s_c arrived after c, and Lemma 3 gives us the required independent set B. We then define a partition matroid $\mathcal{M}_C$ with partition $(I_c)_{c \in C_E}$ and capacities 1 and solve the matroid intersection problem on $\mathcal{M}_c$ and $\mathcal{M}$ restricted to $\bigcup_{c \in C_E} I_c$ and get the output C'. Existence of B guarantees that $|C'| = |C_E|$, thus we are able to serve all points in C_E at a cost of τ. Since the points we forget are within distance 2τ of C_E, our total cost is at most 3τ by triangle inequality. We can get rid of the assumption that s_c arrives after c by having a second pass through the stream. We give a formal description in Algorithm 2.

As in the one-pass algorithm, we run $2 \log_{1+\varepsilon/3} \Delta$ guesses τ of OPT. We return the set of centers returned by the instance corresponding to the smallest guess. For $\tau \in [\mathrm{OPT}, (1+\varepsilon/3)\,\mathrm{OPT})$, the algorithm will not abort due to existence of the independent set B (which we argued earlier). This gives us the following theorem.

▶ **Theorem 12.** *There is an efficient $(3+\varepsilon)$-approximation two-pass algorithm for matroid center that stores at most $2(r^2 + r) \log_{(1+\varepsilon/3)} \Delta$ points.*

5 Matroid Center with Outliers

We first present a simplified analysis of McCutchen and Khuller's algorithm [28] for k-center with z outliers. This abstracts their ideas and sets the stage for the matroid version that we will see later.

5.1 McCutchen and Khuller's Algorithm

As usual, we start with a guess τ for the optimum cost. The algorithm maintains a set C of pivots such that $|\mathfrak{B}(c, 2\tau)| \geqslant z + 1$ for any $c \in C$, so the optimum has to serve at least one of these nearby points. (Recall that $\mathfrak{B}(e, \alpha) = \{x \in E : d(e, x) \leqslant \alpha\}$.) When a new point

Algorithm 2 Two pass algorithm for matroid center.

function MATROIDCENTER2P(τ)
  $C \leftarrow \emptyset$.
  for each point e in the stream **do** ▷ First pass.
    if $d(e, C) \geqslant 2\tau$ **then**
      $C \leftarrow C \cup \{e\}$.
      If $\{e\}$ is not a loop, $I_e \leftarrow \{e\}$, else $I_e \leftarrow \emptyset$.
  for each point e in the stream **do** ▷ Second pass.
    if $\exists c \in C$ such that $d(e, c) \leqslant \tau$ (there can be at most one such c) **then**
      if $I_c \cup \{e\}$ is independent **then**
        $I_c \leftarrow I_c \cup \{e\}$.
  Let $\mathcal{M}_C = (\bigcup_{c \in C} I_c, \mathcal{I}_C)$ be a partition matroid with partition $\{I_c : c \in C\}$ and capacities 1.
  Let $\mathcal{M}'$ be the matroid $\mathcal{M}$ restricted to $\bigcup_{c \in C} I_c$.
  $C' \leftarrow$ MATROID-INTERSECTION($\mathcal{M}_C$, $\mathcal{M}'$)
  if $|C'| < |C|$ **then**
    Return fail with C as certificate.
  Return C'.

arrives, it is ignored if it is within distance 4τ of C. Otherwise it is added to the set F of "free" points. As soon as the size of F reaches $(k - |C| + 1)z + 1$, we know for sure that, for a correct guess, the optimum will have to serve the free points with at most $k - |C|$ clusters, and one of those clusters will have more than z points by the generalized pigeonhole principle. Hence, there *must* exist a free point that has at least z other points within distance 2τ in F, because its cluster diameter is at most 2τ. This gives us a new pivot $c \in F$ with its support points. We remove those points in F that are within distance 4τ of c and continue to the next element in the stream. In the end, we will be left with at most $(k - |C| + 1)z$ free points, and they are served by at most $k - |C|$ optimum centers. On these remaining free points, we run an offline 2-approximation algorithm for $(k - |C|)$-center with z outliers, e.g., that of Chakrabarty et al.[4]. Algorithm 3 gives a formal description. We note that we do not need the sets A_c for $c \in C$ in the algorithm, but we need them in the analysis.

Let us bound the space usage first. The variable C contains at most k pivots, otherwise we abort on Section 5.1, and Section 5.1 make sure that the variable F contains at most $(k + 1)z + 1$ points. In total, we store at most $(k + 1)z + 1$ points at any moment.

▶ **Lemma 13.** *For $\tau \geqslant$ OPT, K-CENTER-Z-OUTLIERS(τ) stores at most $(k + 1)z + 1$ points, and the cost of C' returned by K-CENTER-Z-OUTLIERS(τ) is at most 4τ.*

Proof. Let C_E be the set of pivots and F_E be the set of free points when the stream ended, and let $|C_E| = \ell_E$. We claim that for any $c \neq c'$, where $c, c' \in C_E$, $e \in A_c$, and $e' \in A_{c'}$, we have $d(e, e') > 2\tau$. We now prove this claim. Assume without loss of generality that c was made a pivot before c' by the algorithm. So points within distance 4τ of c were removed from F. Any point that existed in F after this removal, in particular e', must be farther than 4τ from c. This implies that

$$4\tau < d(c, e') \leqslant d(c, e) + d(e, e'), \quad \text{and} \quad d(e, e') > 2\tau\,,$$

because $d(c, e) \leqslant 2\tau$. Now, we know that for $c \in C_E$, there exists $x_c \in A_c$ that has to be served by an optimum center, say s_c, because $|A_c| > z$, so not all of the points in A_c can be outliers. By the earlier claim, for $c \neq c'$, we have $d(x_c, x_{c'}) > 2\tau$ implying that $s_c \neq s_{c'}$ and

Algorithm 3 McCutchen and Khuller's algorithm [28] for k-center with z outliers.

```
1: function K-CENTER-Z-OUTLIERS(τ)
2:     Pivot-set C ← ∅, free-point set F ← ∅, and ℓ ← 0.
3:     for each point e in the stream do
4:         if d(e, C) > 4τ then
5:             F ← F ∪ {e}.
6:         if |F| = (k − ℓ + 1)z + 1 then        ▷ there is a new pivot among the free points;
7:             Let c ∈ F be s.t. |𝔅(c, 2τ) ∩ F| ⩾ z + 1      ▷ such c exists for a correct guess.
8:             If such c does not exist, then abort.
9:             C ← C ∪ {c}.
10:            F ← F \ 𝔅(c, 4τ).
11:            A_c ← {c}∪ arbitrary subset of 𝔅(c, 2τ) \ {c} of size z.
12:            ℓ ← ℓ + 1.
13:            If ℓ = k + 1, then abort.                ▷ guess is wrong.
14:    C_F ← 2-approx for (k − ℓ)-center with z outliers on F by an efficient offline algorithm.
15:    return C′ ← C ∪ C_F.
```

$\ell_E \leqslant k$. Also note that none of these optimum centers can serve a point in F_E, because by triangle inequality

$$d(s_c, F_E) \geqslant d(c, F_E) - d(c, x_c) - d(x_c, s_c) > 4\tau - 2\tau - \tau = \tau$$

for $c \in C_E$. This shows that all but z points in F_E have to be served by at most $k - \ell_E$ optimum centers with cost at most τ. For each of these optimum centers, there exists a free point in F_E within distance τ. So there exists a set B_F of $k - \ell_E$ points in F_E, such that B_F covers all but at most z points of F_E with cost 2τ. So a 2-approximation algorithm recovers $k - \ell_E$ centers with cost at most 4τ. Observing that we only forget points in the stream that are within distance 4τ of some pivot in C_E finishes the proof. ◀

By running K-CENTER-Z-OUTLIERS(τ) for at most $O(\log(1/\varepsilon)/\varepsilon)$ geometrically-increasing active guesses, we get the $(4+\varepsilon)$-approximation algorithm for k-center with z outliers. This analysis is based on that of McCutchen and Khuller [28].

5.2 Matroid Center with Outliers

It is now possible to naturally combine the ideas used for matroid center and those used for k-center with z outliers to develop an algorithm for matroid center with z outliers.

Whenever the free-point set becomes large enough, we create a pivot c and an independent set I_c to which we try to add all free points within distance 4τ of c. We do the same for a new point e in the stream that is within distance 4τ of some pivot $c \in C$, i.e., we try to add it to I_c keeping I_c independent in the matroid. Otherwise $d(e, C) > 4\tau$, so we make it a free point. The structural property of matroids that we proved as Lemma 3 then enables us to show that $\bigcup_{c \in C} I_c$ and the set of free points make a good summary of the stream. See Algorithm 4 for a formal description. Here, we note that we do not need the sets A_c for $c \in C$ in the algorithm if flag is set to "brute force", but we need them in the analysis in any case.

Let C_E be the set of pivots and F_E be the set of free points when the stream ended, and let $\ell_E = |C_E|$.

Algorithm 4 One-pass algorithm for matroid center with outliers.

function MATROID-CENTER-Z-OUTLIERS(τ, flag)
  Pivot-set $C \leftarrow \emptyset$, free-point set $F \leftarrow \emptyset$, and $\ell \leftarrow 0$.
  for each point e in the stream **do**
    if $\exists c \in C$ such that $d(e, c) \leqslant 4\tau$ **then**
      If $I_c \cup \{e\}$ is independent, then $I_c \leftarrow I_c \cup \{e\}$.
    else
      $F \leftarrow F \cup \{e\}$.
    if $|F| = (r - \ell + 1)z + 1$ **then**
      Let $c \in F$ be s. t. $|\mathfrak{B}(c, 2\tau) \cap F| \geqslant z + 1$ (if not, we guessed wrong, so abort).
      $C \leftarrow C \cup \{c\}$.
      $A_c \leftarrow \{c\}$ and if $\{c\}$ is not a loop, $I_c \leftarrow \{c\}$, else $I_c \leftarrow \emptyset$.
      $\ell \leftarrow \ell + 1$ (if ℓ becomes $r + 1$ here, we guessed wrong, so abort).
      for each $x \in F \cap \mathfrak{B}(c, 4\tau)$ **do**
        $F \leftarrow F \setminus \{x\}$.
        If $I_c \cup \{x\}$ is independent, then $I_c \leftarrow I_c \cup \{x\}$.
        If $|A_c| \leqslant z$, then $A_c \leftarrow A_c \cup \{x\}$.
  if flag = "brute force" **then**
    Find an independent set C'_B in $F \cup \bigcup_{c \in C} I_c$ by brute force such that cost of C'_B is $\leqslant 11\tau$ with respect to C and $\leqslant 9\tau$ with respect to all but at most z points of F.
    If such C'_B does not exist, abort, else return C'_B.
  if flag = "efficient" **then**
    Run the offline 3-approximation algorithm by Harris et al. [19] for matroid center with z outliers to get an independent set C' of centers in $F \cup \bigcup_{c \in C}(A_c \cup I_c)$ such that cost of C' is $\leqslant 47\tau$ with respect to C and $\leqslant 45\tau$ with respect to all but z points of F.
    If such C' does not exist, abort, else return C'.

▶ **Lemma 14** (Small summary for matroid center with outliers). *For $\tau \geqslant \mathrm{OPT}$, Algorithm 4 stores at most $O(r^2 + rz)$ points, and there exists an independent set $B \subseteq F_E \cup \bigcup_{c \in C_E} I_c$ such that cost of B is at most 15τ; also $d(c, B) \leqslant 11\tau$ for any pivot $c \in C_E$, and B covers all but at most z points of F_E with cost at most 9τ.*

Proof. Let I^* be an optimum independent set of centers. By the same argument as in the proof of Lemma 13, the following claim is true. For any $c \neq c'$, where $c, c' \in C_E$, $e \in A_c$, and $e' \in A_{c'}$, we have $d(e, e') > 2\tau$. Now, we know that for $c \in C_E$, there exists $x_c \in A_c$ that has to be served by an optimum center, say s_c, because $|A_c| > z$. By the earlier claim, for $c \neq c'$, we have $d(x_c, x_{c'}) > 2\tau$ implying that $s_c \neq s_{c'}$ and $\ell_E \leqslant r$. Let $I^*_{C_E} = \{s_c : c \in C_E\}$ be the set of optimum centers that serve some $x_c \in A_c$ for $c \in C_E$. None of the optimum centers in $I^*_{C_E}$ can serve a point in F_E, because $d(s_c, F_E) > \tau$ for $c \in C_E$. This shows that all but z points in F_E have to be served by at most $r - \ell_E$ optimum centers with cost at most τ. Since $|I_c| \leqslant r$ for any c in the variable C, size of $\bigcup_{c \in C} I_c$ is always bounded by r^2. Also, the check on the size of F ensures that $|F| \leqslant (r + 1)z + 1$, so total number of points stored is at most $O(r^2 + rz)$ at any moment.

When we first process a new point e in the stream, we either try to add it to some I_c or to F. If e is never removed from F, then $e \in F_E$, otherwise, we try to add it to some I_c. The same argument applies to any $x \in I^*$, so if $x \notin F_E$, then we did try to add it to some I_c. For an $x \in I^* \setminus F_E$, let $a(x) \in C_E$ denote the pivot whose independent set $I_{a(x)}$ we tried to add x to.

By Lemma 3, for $\mathcal{A} := \{I_{a(x)} : x \in I^* \setminus F_E\} \cup \{\{x\} : x \in I^* \cap F_E\}$, there exists an independent set B such that $|I \cap B| \geqslant 1$ for all $I \in \mathcal{A}$. Since for $x \in I^* \cap F_E$ the singleton $\{x\} \in \mathcal{A}$, the set B must contain $\{x\}$. For a free point e served by an optimum center s such that we tried to add s to some I_c, we have that $d(e, B) \leqslant d(e, s) + d(s, c) + d(c, B) \leqslant \tau + 4\tau + 4\tau = 9\tau$, which means that B serves all but z points of F_E with cost at most 9τ. Now, we claim that for any point e in the stream, $d(e, B) \leqslant 15\tau$. We just saw that if $e \in F_E$ is served by an optimum center, then $d(e, B) \leqslant 9\tau$, so assume that $e \notin F_E$, that means there is a $c \in C_E$ such that $d(e, c) \leqslant 4\tau$; denote this c by $c(e)$. See Figure 3. Let $s_{c(e)}$ be the optimum center that serves an $x_{c(e)} \in A_{c(e)}$ (recall that such a point exists because $|A_{c(e)}| > z$). So $d(c(e), s_{c(e)}) \leqslant 3\tau$, and $a(s_{c(e)}) \in C_E$ was the pivot such that $d(s_{c(e)}, a(s_{c(e)})) \leqslant 4\tau$. Let $c'(e)$ be an arbitrary point in $I_{a(s_{c(e)})} \cap B$, whose existence is guaranteed by the property of B. We have $d(a(s_{c(e)}), c'(e)) \leqslant 4\tau$, because $c'(e) \in I_{a(s_{c(e)})}$. Then by triangle inequality,

$$\begin{aligned} d(e, B) &\leqslant d(e, c'(e)) \\ &\leqslant d(e, c(e)) + d(c(e), x_{c(e)}) + d(x_{c(e)}, s_{c(e)}) + d(s_{c(e)}, a(s_{c(e)})) + d(a(s_{c(e)}), c'(e)) \\ &\leqslant 4\tau + 2\tau + \tau + 4\tau + 4\tau = 15\tau\,, \end{aligned}$$

hence, cost of B is at most 15τ.

For any $c \in C_E$, we can bound $d(c, B)$ in a similar way. Let s_c be the optimum center that serves an $x_c \in A_c$. Define $a(s_c)$ to be the pivot such that $d(s_c, a(s_c)) \leqslant 4\tau$. Also, let c' be the point in B such that $d(a_{s_c}, c') \leqslant 4\tau$. This gives that $d(c, B) \leqslant d(c, c') \leqslant 11\tau$. We already established that B covers all but at most z points of F_E with cost at most 9τ. The proof is now complete. ◀

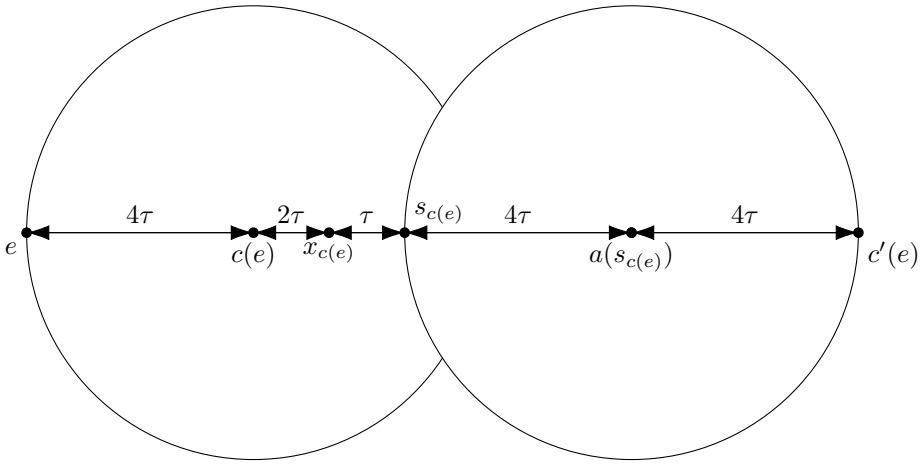


Figure 3 To see how to bound the cost of the independent set B, let e be any point in the stream, $c(e) \in C_E$ be the pivot close to e, $x_{c(e)}$ be the point in the support $A_{c(e)}$ of $c(e)$ that an optimum center serves, $s_{c(e)}$ be the optimum center that serves $x_{c(e)}$, $a(s_{c(e)}) \in C_E$ be the pivot close to $s_{c(e)}$, and $c'(e)$ be a point in B that covers $a(s_{c(e)})$.

▶ **Theorem 15.** *There is an efficient $(51 + \varepsilon)$-approximation one-pass algorithm for matroid center with z outliers that stores at most $O((r^2 + rz) \log \Delta/\varepsilon)$ points. With a brute force algorithm, one can get a $(15 + \varepsilon)$-approximation.*

Proof. We run $O(\log \Delta/\varepsilon)$ parallel copies of MATROID-CENTER-Z-OUTLIERS(τ, flag) and return the output of the copy for the smallest unaborted guess. We claim that the copy corresponding to guess $\tau' \in [\mathrm{OPT}, (1+\varepsilon/50)\,\mathrm{OPT})$, call it $\mathbb{I}(\tau')$, will not abort. Denote by C_E, F_E, and $(I_c)_{c\in C_E}$ contents of the corresponding variables in $\mathbb{I}(\tau')$ at the end of the stream (we will not abort mid-stream because $\tau' \geqslant \mathrm{OPT}$).

By Lemma 14, $F_E \cup \bigcup_{c\in C_E} I_c$ contains a solution that has cost $11\tau'$ with respect C_E and $9\tau'$ with respect to all but at most z of F_E. These checks can be performed by the brute force algorithm. Since any instance for guess τ forgets only those points within distance 4τ of its pivots, the brute force algorithm outputs a $(15+\varepsilon)$-approximation.

By Lemma 14, there exists a solution of cost $\leqslant 15\tau'$, and the efficient 3-approximation algorithm for matroid center with z outliers will return a solution C' with cost at most $45\tau'$. Note that C' has to cover at least one point from A_c for each $c \in C_E$, hence $d(c, C') \leqslant 47\tau'$. Since we forget points only within distance $4\tau'$ of C_E, we get the desired approximation ratio. ◀

References

1 Kook Jin Ahn, Graham Cormode, Sudipto Guha, Andrew McGregor, and Anthony Wirth. Correlation Clustering in Data Streams. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 2237–2246, 2015.

2 Ashwinkumar Badanidiyuru Varadaraja. Buyback problem: approximate matroid intersection with cancellation costs. In *Proceedings of the 38th international colloquium conference on Automata, languages and programming - Volume Part I*, ICALP'11, pages 379–390, 2011.

3 Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: matchings, matroids, and more. *Mathematical Programming*, 154(1):225–247, 2015. doi:10.1007/s10107-015-0900-7.

4 Deeparnab Chakrabarty, Prachi Goyal, and Ravishankar Krishnaswamy. The Non-Uniform k-Center Problem. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016*, pages 67:1–67:15, 2016.

5 Deeparnab Chakrabarty and Maryam Negahbani. Generalized Center Problems with Outliers. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107, pages 30:1–30:14, 2018.

6 T-H. Hubert Chan, Arnaud Guerqin, and Mauro Sozio. Fully Dynamic k-Center Clustering. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 579–587, 2018.

7 Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental Clustering and Dynamic Information Retrieval. In *Proc. 29th Annual ACM Symposium on the Theory of Computing*, STOC '97, pages 626–635, 1997.

8 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for Facility Location Problems with Outliers. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, pages 642–651, 2001.

9 Moses Charikar, Liadan O'Callaghan, and Rina Panigrahy. Better Streaming Algorithms for Clustering Problems. In *Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 30–39. ACM, 2003.

10 Chandra Chekuri, Shalmoli Gupta, and Kent Quanrud. Streaming Algorithms for Submodular Function Maximization. In *Proc. 42nd International Colloquium on Automata, Languages and Programming*, pages 318–330, 2015.

11 Danny Z. Chen, Jian Li, Hongyu Liang, and Haitao Wang. Matroid and Knapsack Center Problems. *Algorithmica*, 75(1):27–52, May 2016.

12 Vincent Cohen-Addad, Chris Schwiegelshohn, and Christian Sohler. Diameter and k-Center in Sliding Windows. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55, pages 19:1–19:12, 2016.

13 Teofilo F. Gonzalez. Clustering to Minimize the Maximum Intercluster Distance. *Theor. Comput. Sci.*, 38:293–306, 1985.

14 S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams: Theory and practice. *IEEE Transactions on Knowledge and Data Engineering*, 15(3):515–528, May 2003.

15 Sudipto Guha. Tight Results for Clustering and Summarizing Data Streams. In *Proc. 12th International Conference on Database Theory*, ICDT '09, pages 268–275, 2009.

16 MohammadTaghi Hajiaghayi, Rohit Khandekar, and Guy Kortsarz. Budgeted Red-blue Median and Its Generalizations. In *Proceedings of the 18th Annual European Conference on Algorithms: Part I*, ESA'10, pages 314–325. Springer-Verlag, 2010. URL: `http://dl.acm.org/citation.cfm?id=1888935.1888972`.

17 S. L. Hakimi. Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph. *Oper. Res.*, 12(3):450–459, June 1964. `doi:10.1287/opre.12.3.450`.

18 S. L. Hakimi. Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems. *Oper. Res.*, 13(3):462–475, June 1965. `doi:10.1287/opre.13.3.462`.

19 David G. Harris, Thomas Pensyl, Aravind Srinivasan, and Khoa Trinh. A Lottery Model for Center-Type Problems with Outliers. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 10:1–10:19, 2017. `doi:10.4230/LIPIcs.APPROX-RANDOM.2017.10`.

20 Behnam Hatami and Hamid Zarrabi-Zadeh. A Streaming Algorithm for 2-Center with Outliers in High Dimensions. *Comput. Geom.*, 60:26–36, 2017.

21 Dorit S. Hochbaum and David B. Shmoys. A Best Possible Heuristic for the k-Center Problem. *Math. Oper. Res.*, 10(2):180–184, May 1985. `doi:10.1287/moor.10.2.180`.

22 Dorit S. Hochbaum and David B. Shmoys. A Unified Approach to Approximation Algorithms for Bottleneck Problems. *J. ACM*, 33(3):533–550, May 1986. `doi:10.1145/5925.5933`.

23 Wen-Lian Hsu and George L. Nemhauser. Easy and hard bottleneck location problems. *Discrete Applied Mathematics*, 1(3):209–215, 1979. `doi:10.1016/0166-218X(79)90044-1`.

24 Sang-Sub Kim and Hee-Kap Ahn. An improved data stream algorithm for clustering. *Computational Geometry*, 48(9):635–645, 2015. `doi:10.1016/j.comgeo.2015.06.003`.

25 Ravishankar Krishnaswamy, Amit Kumar, Viswanath Nagarajan, Yogish Sabharwal, and Barna Saha. The Matroid Median Problem. In *Proceedings of the Twenty-second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 1117–1130, 2011.

26 Silvio Lattanzi, Stefano Leonardi, Vahab Mirrokni, and Ilya Razenshteyn. Robust Hierarchical k-Center Clustering. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS '15, pages 211–218, 2015.

27 Gustavo Malkomes, Matt J Kusner, Wenlin Chen, Kilian Q Weinberger, and Benjamin Moseley. Fast Distributed k-Center Clustering with Outliers on Massive Data. In *Advances in Neural Information Processing Systems 28*, pages 1063–1071. Curran Associates, Inc., 2015. URL: `http://papers.nips.cc/paper/5997-fast-distributed-k-center-clustering-with-outliers-on-massive-data.pdf`.

28 Richard Matthew McCutchen and Samir Khuller. Streaming Algorithms for k-Center Clustering with Outliers and with Anonymity. In *Proc. 11th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 165–178, 2008.

29 Hamid Zarrabi-Zadeh and Asish Mukhopadhyay. Streaming 1-Center with Outliers in High Dimensions. In *Proceedings of the 21st Annual Canadian Conference on Computational Geometry, Vancouver, British Columbia, Canada, August 17-19, 2009*, pages 83–86, 2009.

A Handling the Guesses

We extend the ideas of Guha [15] and McCutchen and Khuller [28] to run $O(\log(1/\varepsilon)/\varepsilon)$ active guesses. Although, to make this idea work for matroids, we do need a property of matroids (see Lemma 16). The way to do this is to start with a lower bound R on the optimum and spawn instances, which we call *original* instances, $\mathbb{I}(\tau)$ for guesses $\tau = R, R(1+\varepsilon), \ldots, R(1+\varepsilon)^{\beta} = R\alpha/\varepsilon$, for some α that depends on the basic algorithm that we use, e.g., for matroid center, we will use $\alpha = 2+\varepsilon$. When a guess τ' fails, we replace an instance $\mathbb{I} = \mathbb{I}(\tau)$ for $\tau \leqslant \tau'$ with a new instance, which we call its *child* instance, $\mathbb{I}_N = \mathbb{I}(\tau(1+\varepsilon)^{\beta})$. In the new instance $\mathbb{I}_N$, we treat the summary that we maintained for $\mathbb{I}(\tau)$ as the initial stream. Since the new guess in $\mathbb{I}_N$ is about $1/\varepsilon$ times larger than the old guess in $\mathbb{I}$, the distance between a point that we forgot and the summary stored by $\mathbb{I}$ is about ε times the new guess. Therefore, the cost analysis does not get much affected for a correct guess. If we forgot an optimum center, a nearby point in the summary can act as its replacement. This statement is obvious for a uniform matroid, because all points are treated the same way within the matroid, but it is not true for general matroids; in fact, as exhibited by our lower bound, it is not true even for partition matroids. So with each point in the summary, we pass to the new instance an independent set I_o. The following simple lemma shows that if an optimum center x is in the span of I_o, and if we construct I_c for a new pivot c such that $I_o \subseteq \text{span}(I_c)$, then I_c also spans the optimum center.

▶ **Lemma 16.** *Let I and J be independent sets of a matroid such that $J \subseteq \text{span}(I)$. If $e \in \text{span}(J)$, then $e \in \text{span}(I)$.*

Proof. Let $\text{rank}(I) = q$. Towards a contradiction, let $\text{rank}(I \cup \{e\}) = q+1$. Since $J \subseteq \text{span}(I)$, $\text{rank}(I \cup J) = q$. Now, $e \in \text{span}(J)$, so $\text{rank}(I \cup J \cup \{e\}) = q$, i.e., $\text{rank}(I \cup \{e\}) \leqslant q$, which gives us the desired contradiction. ◀

A.1 A Smaller Space Algorithm for Matroid Center

We modify the function MatroidCenter(τ,flag) from earlier to accept a starting stream and an independent set for each point in the starting stream: MatroidCenter(τ, C_o, $(J_{c_o})_{c_o \in C_o}$,flag). Before processing any new points in the stream we process the points in C_o as follows. When processing a $c_o \in C_o$, if $d(c_o, C) \leqslant 2\tau$, try to add points in J_{c_o} to I_c. Otherwise create a new pivot c in C and initialize $I_c = J_{c_o}$. Once C_o is processed, we continue with the stream and work exactly as in MatroidCenter(τ). We give complete pseudocode in Algorithm 5.

For an instance $\mathbb{I}(\tau)$ let $C_o(\tau)$ be the initial summary and $\mathcal{J}(\tau)$ be the collection of independent sets that we passed to it, and let $E(\tau)$ be the part of the actual stream that it processed. Also, let $\mathbb{I}(\tau_o)$ be the instance for $\tau_o = \varepsilon\tau/(2+\varepsilon)$ from which $\mathbb{I}(\tau)$ was spawned.

▶ **Lemma 17.** *Let e be a point that arrived before the substream $E(\tau)$. Then e has a nearby* representative *$\rho_e \in C_o(\tau)$ such that $d(e, \rho_e) \leqslant \varepsilon\tau$ and also the independent set J_{ρ_e} corresponding to ρ_e spans e.*

Proof. We prove this claim by induction on the number of ancestors. For an original instance, the claim holds trivially, because no point arrived before. Otherwise, there are two cases: either $e \in E(\tau_o)$ or e arrived before $E(\tau_o)$. If $e \in E(\tau_o)$, then by the logic of the algorithm, there exists $a(e) \in C_o(\tau)$ such that $d(e, a(e)) \leqslant 2\tau_o = 2\varepsilon\tau/(2+\varepsilon) \leqslant \varepsilon\tau$, and also we tried

Algorithm 5 One pass algorithm for matroid center with smaller space.

Let R be the minimum distance for some two points in the first $r+1$ points in the stream.
for $\tau \in \{R, R(1+\varepsilon), \ldots, R(1+\varepsilon)^{\beta} = (2+\varepsilon)R/\varepsilon\}$ in parallel **do**
    MatroidCenter(τ, $\emptyset$, $\emptyset$).
    if an instance with guess τ is aborted **then**
        for all active $\mathbb{I}(\tau')$ with guess $\tau' \leqslant \tau$, current pivots C_o, and independent sets $(J_{c_o})_{c_o \in C_o}$ **do**
            Replace it with the child instance MatroidCenter($\tau'(1+\varepsilon)^{\beta}$, C_o, $(J_{c_o})_{c_o \in C_o}$, flag).
Return the set C' of centers returned by the active instance with the smallest guess.

function MatroidCenter(τ, C_o, $(J_{c_o})_{c_o \in C_o}$, flag)
    $C \leftarrow \emptyset$.
    for each point c_o in C_o **do**
        if $\exists c \in C$ such that $d(c_o, c) \leqslant 2\tau$ (pick arbitrary such c if there are several) **then**
            for $e_o \in J_{c_o}$ **do**
                if $I_c \cup \{e_o\}$ is independent **then**
                    $I_c \leftarrow I_c \cup \{e_o\}$.
        else
            $C \leftarrow C \cup \{c_o\}$.
            $I_{c_o} \leftarrow J_{c_o}$.
    # Processing of the old pivots finished, continue with the actual stream.
    for each point e in the stream **do**
        if there is a pivot $c \in C$ such that $d(e, c) \leqslant 2\tau$ (pick arbitrary such c) **then**
            if $I_c \cup \{e\}$ is independent **then**
                $I_c \leftarrow I_c \cup \{e\}$.
        else if $|C| = r$ **then** ▷ We cannot have more pivots than the rank.
            Abort. ▷ Because $C \cup \{e\}$ acts as a certificate that the guess is incorrect.
        else
            $C \leftarrow C \cup \{e\}$. ▷ Make e a pivot.
            If $\{e\}$ is not a loop, $I_e \leftarrow \{e\}$, else $I_e \leftarrow \emptyset$.
    if flag = "brute force" **then**
        Find an independent set C'_B in $\bigcup_{c \in C} I_c$ such that $d(c, C'_B) \leqslant (5+2\varepsilon)\tau$ for $c \in C$.
        If such C'_B does not exist, then abort, else return C'_B.
    return EfficientMatroidCenter($(5+2\varepsilon)\tau$, C, $(I_c)_{c \in C}$, $\mathcal{M}$).

to add e to $I_{a(e)}$ (that became $J_{a(e)}$ for the next instance $\mathbb{I}(\tau)$). Otherwise, by induction hypothesis, there is a point $e' \in C_o(\tau_o)$ such that $d(e, e') \leqslant \varepsilon\tau_o$ and $J_{e'}$ spans e. Now, let $\rho_{e'} \in C_o(\tau)$ be such that $d(e', \rho_{e'})) \leqslant 2\tau_o$ (such $\rho_{e'}$ must exist by logic of the algorithm). Using triangle inequality and the above inequality that $d(e, e') \leqslant \varepsilon\tau_o$, we get

$$d(e, \rho_{e'}) \leqslant d(e, e') + d(e', \rho_{e'}) \leqslant \varepsilon\tau_o + 2\tau_o = (2+\varepsilon)\tau_o = (2+\varepsilon)\frac{\varepsilon\tau}{(2+\varepsilon)} = \varepsilon\tau\,.$$

Moreover, in the instance $\mathbb{I}(\tau_o)$, we tried to add all points in $J_{e'}$ to $I_{\rho_{e'}}$, so by Lemma 16, $e \in \mathrm{span}(I_{\rho_{e'}})$ (see that $I_{\rho_{e'}}$ became $J_{\rho_{e'}}$ for the next instance $\mathbb{I}(\tau)$), which proves the claim. ◀

▶ **Theorem 18.** *There is an efficient* $((17+7\varepsilon)(1+\varepsilon))$*-approximation one-pass algorithm for matroid center that stores at most* $O(r^2 \log(1/\varepsilon)/\varepsilon)$ *points. With a brute force algorithm, one can get a* $((7+3\varepsilon)(1+\varepsilon))$*-approximation.*

Proof. Space usage is easy to analyze. At any time, we have at most $O(\log_{1+\varepsilon}(1/\varepsilon)) = O(\log(1/\varepsilon)/\varepsilon)$ active instances and each instance stores at most $O(r^2)$ points.

Consider the instance $\mathbb{I}(\tau')$ for which we returned on Line 7 in Algorithm 5, and suppose the outputs were C' or C'_B (depending on "flag"). We note that some active copy *will* return, because τ cannot keep on increasing indefinitely. E.g., consider τ larger than the maximum distance between any two points. Let C_E be the contents of the variable C in $\mathbb{I}(\tau')$ at the end of the stream. Then we know that costs of C'_B and C' are at most $(5+2\varepsilon)\tau'$ and $(15+6\varepsilon)\tau'$ with respect to C_E due to the check that we do on Line 31 and by Theorem 5 for EfficientMatroidCenter. By Lemma 17, any point that arrived before $E(\tau')$ is within distance $\varepsilon\tau'$ of $C_o(\tau')$, and each point in $C_o(\tau')$ is within distance $2\tau'$ of C_E, which shows that costs of C'_B and C' are at most $(7+3\varepsilon)\tau'$ and $(17+7\varepsilon)\tau'$ with respect to the whole stream (by triangle inequality). Next, we show that $\tau' \leqslant (1+\varepsilon)\,\mathrm{OPT}$, and that will finish the proof.

Consider the guess $\tau \in (\mathrm{OPT}, (1+\varepsilon)\,\mathrm{OPT}]$. If τ was never active, that means $\tau' \leqslant \mathrm{OPT}$, and we are done. Otherwise, τ was active, and we will prove that it was not aborted. Since $\tau \leqslant \mathrm{OPT}$, we will not abort mid-stream in $\mathbb{I}(\tau)$, so let C_E be the set of pivots at the end of the stream in $\mathbb{I}(\tau)$. We will show that there is an independent set B such that cost of B with respect to C_E is at most $(5+2\varepsilon)\tau$. By Line 31 and by Theorem 5 for EfficientMatroidCenter, this would imply that $\mathbb{I}(\tau)$ cannot abort.

From here on, the proof follows that of Lemma 4. Let $c \in C_E$. Denote by s_c the optimum center that serves it, so $d(c, s_c) \leqslant \tau$. If $s_c \in E(\tau)$, then $s_c \in \mathrm{span}(I_{c'})$ for some $c' \in C_E$ and $d(s_c, c') \leqslant 2\tau$. Otherwise, s_c arrived before $E(\tau)$. Let ρ_{s_c} be the representative of s_c whose existence is guaranteed by Lemma 17, so $d(s_c, \rho_{s_c}) \leqslant \varepsilon\tau$. Then let $c' \in C_E$ be such that $d(\rho_{s_c}, c') \leqslant 2\tau$ and $J_{\rho_{s_c}}$ is spanned by $I_{c'}$. Thus, by triangle inequality

$$d(s_c, c') \leqslant d(s_c, \rho_{s_c}) + d(\rho_{s_c}, c') \leqslant \varepsilon\tau + 2\tau = (2+\varepsilon)\tau\,, \tag{2}$$

and by Lemma 16, s_c is spanned by $I_{c'}$. Denote by $\mathcal{A}$ the collection of such $I_{c'}$'s. Now, by Lemma 3, there exists an independent set B such that $|I \cap B| \geqslant 1$ for all $I \in \mathcal{A}$. Pick c_p from $I_{c'} \cap B$. Either $c_p \in E(\tau)$ or it arrived before. In any case, again using Lemma 17, we have $d(c_p, c') \leqslant (2+\varepsilon)\tau$ (we use this below), and

- $d(c, s_c) \leqslant \tau$, because s_c is the optimum center that covers c,
- $d(s_c, c') \leqslant (2+\varepsilon)\tau$, by Inequality (2), and
- $d(c', c_p) \leqslant (2+\varepsilon)\tau$.

Thus, by triangle inequality, $d(c, B) \leqslant (5+2\varepsilon)\tau$. So $\mathbb{I}(\tau)$ will not abort. This finishes the proof. ◀

Reducing the space usage for matroid center with z outliers can be done by naturally combining the techniques above and those in Section 5.2. We define a similar overloading matroid-center-z-outliers(τ, C_o, $(J_{c_o})_{c_o \in C_o}$, F_o, flag), where F_o contains the set of free points in $\mathbb{I}(\tau_o)$ when it aborted and this function was called with the updated guess τ. We skip the details and state the following theorem without proof.

▶ **Theorem 19.** *There is an efficient* $(51+\varepsilon)$*-approximation one-pass algorithm for matroid center with* z *outliers that stores at most* $O((r^2+rz)\log(1/\varepsilon)/\varepsilon)$ *points. With a brute force algorithm, one can get a* $(15+\varepsilon)$*-approximation.*

A.2 Extension to Knapsack Center

In Section 4.1, we saw how to modify Algorithm 1 to get an algorithm for knapsack center that stores at most $2r$ points, where r is the size of a largest feasible set. Using the same idea, algorithms for two-pass matroid center, matroid center with outliers, and smaller space matroid center, which are Algorithms 2, 4 and 6, can be extended to the knapsack center without losing the approximation ratio and with a space r factor smaller than the matroid case. For the outlier version of knapsack center, to get an efficient algorithm, we use the 3-approximation algorithm by Chakrabarty and Negahbani [5]. So we get the following theorems, where r is the size of a largest feasible set.

▶ **Theorem 20.** *There is an efficient $(17+\varepsilon)$-approximation one-pass algorithm for knapsack center that stores at most $O(r\log(1/\varepsilon)/\varepsilon)$ points. With a brute force algorithm, one can get a $(7+\varepsilon)$-approximation.*

▶ **Theorem 21.** *There is an efficient $(51+\varepsilon)$-approximation one-pass algorithm for knapsack center with z outliers that stores at most $O(rz\log(1/\varepsilon)/\varepsilon)$ points. With a brute force algorithm, one can get a $(15+\varepsilon)$-approximation.*

B An Implementation of Efficient Matroid Center

We now give an implementation of EfficientMatroidCenter. The input consists of α, C_E, X, such that $C_E \subseteq X$, and the underlying matroid $\mathcal{M}$ defined over X. Furthermore, the promise is that there is an independent set $B \subseteq X$ such that for each $c \in C_E$, we have $d(c,B) \leqslant \alpha$. Our implementation is based on the algorithm of Chen et al. [11] for matroid center. We show that it outputs a set C' such that, assuming the promise, $d(c,C') \leqslant 3\alpha$ for $c \in C_E$.

Algorithm 6 Efficient algorithm for matroid center based on the algorithm by [11].

```
1:  function EfficientMatroidCenter(α, C_E, X, M)
2:      Initialize: C ← ∅.
3:      while there is an unmarked point e in C_E do
4:          C ← C ∪ {e}, B_e ← 𝔅(e, α) ∩ X, and mark all points in 𝔅(e, 2α) ∩ C_E.
5:      Let M_C = (∪_{c∈C} B_c, I_C) be a partition matroid with partition {B_c : c ∈ C} and
        capacities 1.
6:      Let M' be the matroid M restricted to ∪_{c∈C} B_c.
7:      C' ← matroid-intersection(M_C, M')
8:      if |C'| < |C| then
9:          Return fail.
10:     Return C'.
```

▶ **Theorem 22.** *If* EfficientMatroidCenter *does not fail, then it outputs a set C' such that $d(c,C') \leqslant 3\alpha$ for each $c \in C_E$. If the input promise holds, then* EfficientMatroidCenter *does not fail.*

Proof. In this proof, we refer by C the contents of the variable C after the while loop ended, and let c_E be any arbitrary point in C_E. Define the function Marker : $C_E \to C$ such that Marker$(c_E) \in C$ is the "marker" of c_E, i.e., we marked c_E when processing Marker(c_E). In the end, all c_E's are marked, so Marker is a valid function. By the logic on Line 4, we have that

$$d(c_E, \text{Marker}(c_E)) \leqslant 2\alpha\,. \tag{3}$$

Let EfficientMatroidCenter does not fail, then $|C'| \geqslant |C|$ and C' satisfies the partition matroid constraint of $\mathcal{M}_C$. By definition of $\mathcal{M}_C$, $\operatorname{rank}(\mathcal{M}_C) = |C|$, hence $|C'| \leqslant |C|$, which implies that $|C'| = |C|$. Therefore, for each $c \in C$, the set C' must contain exactly one element in $\mathfrak{B}(c, \alpha)$ and $d(c, C') \leqslant \alpha$, in particular, $d(\text{Marker}(c_E), C') \leqslant \alpha$. This, triangle inequality, and Inequality (3) gives

$$d(c_E, C') \leqslant d(c_E, \text{Marker}(c_E)) + d(\text{Marker}(c_E), C') \leqslant 2\alpha + \alpha = 3\alpha\,,$$

which proves the first part of the statement of the lemma. We prove the second part next.

Assume that the promise holds. Then let B be the set such that cost of B is at most α with respect to C_E, in particular, with respect to C. For $c \in C$, define $\text{Coverer}(c) \in B$ to be an arbitrarily chosen "coverer" of c, i.e.,

$$d(c, \text{Coverer}(c)) \leqslant \alpha\,. \tag{4}$$

Then the set $B' := \{\text{Coverer}(c) : c \in C\}$ is a subset of B, so it is independent in $\mathcal{M}$. Now, for $c, c' \in C$, such that $c \neq c'$, we have $\text{Coverer}(c) \neq \text{Coverer}(c')$ by Inequality (4) because $d(c, c') > 2\alpha$. This implies that $|B'| = |C|$. Next, $\text{Coverer}(c) \in B' \cap B_c$ for each $c \in C$, hence the set B' is also independent in $\mathcal{M}_C$. Therefore $B' \in \mathcal{M}_C \cap \mathcal{M}'$, and matroid-intersection returns an independent set of size $|C|$, i.e., it does not fail. ◀

▶ Remark 23. By running $\binom{|X|}{2}$ guesses, EfficientMatroidCenter can be used to get an offline 3-approximation algorithm for a more general version of matroid center, where the cost is computed with respect to a subset C_E of X and any point in X can be a center.

Improved Bounds for Open Online Dial-a-Ride on the Line

Alexander Birx
Institute of Mathematics and Graduate School CE, TU Darmstadt, Germany
birx@gsc.tu-darmstadt.de

Yann Disser
Institute of Mathematics, TU Darmstadt, Germany
disser@mathematik.tu-darmstadt.de

Kevin Schewior
Institut für Informatik, Technische Universität München, Garching, Germany
kschewior@gmail.com

Abstract

We consider the open, non-preemptive online Dial-a-Ride problem on the real line, where transportation requests appear over time and need to be served by a single server. We give a lower bound of 2.0585 on the competitive ratio, which is the first bound that strictly separates online Dial-a-Ride on the line from online TSP on the line in terms of competitive analysis, and is the best currently known lower bound even for general metric spaces. On the other hand, we present an algorithm that improves the best known upper bound from 2.9377 to 2.6662. The analysis of our algorithm is tight.

2012 ACM Subject Classification Theory of computation → Online algorithms; Mathematics of computing → Combinatorial optimization

Keywords and phrases dial-a-ride on the line, elevator problem, online algorithms, competitive analysis, smartstart, competitive ratio

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.21

Category APPROX

Related Version A full version of the paper is available at `http://arxiv.org/abs/1907.02858`.

Funding *Alexander Birx*: This work was supported by the "Excellence Initiative" of the German Federal and State Governments and the Graduate School CE at TU Darmstadt.
Yann Disser: This work was supported by the "Excellence Initiative" of the German Federal and State Governments and the Graduate School CE at TU Darmstadt.
Kevin Schewior: Supported by the DAAD within the PRIME program using funds of BMBF and the EU Marie Curie Actions.

1 Introduction

We consider the online Dial-a-Ride problem on the line, where transportation requests appear over time and need to be transported to their respective destinations by a single server. More precisely, each request is of the form $\sigma_i = (a_i, b_i; r_i)$ and appears in position $a_i \in \mathbb{R}$ along the real line at time $r_i \geq 0$ and needs to be transported to position $b_i \in \mathbb{R}$. The server starts at the origin, can move at unit speed, and has a capacity $c \in \mathbb{N} \cup \{\infty\}$ that bounds the number of requests it can carry simultaneously. The objective is to minimize the completion time, i.e., the time until all requests have been served. In this paper, we focus on the *non-preemptive* and *open* setting, where the former means that requests can only be unloaded at their destinations, and the latter means that we do not require the server to return to the origin after serving all requests.


© Alexander Birx, Yann Disser, and Kevin Schewior;
licensed under Creative Commons License CC-BY



We aim to bound the *competitive ratio* of the problem, i.e., the smallest ratio any online algorithm can guarantee between the completion time of its solution compared to an (offline) optimum solution that knows all requests ahead of time. To date, the best known lower bound of 2.0346 on this ratio was shown by Bjelde et al. [5], already for online TSP, where $a_i = b_i$ for all requests (i.e., requests only need to be visited). The best known upper bound of 2.9377 was achieved by the SMARTSTART algorithm [4].

Our results. Our first result is an improved lower bound for online DIAL-A-RIDE on the line. Importantly, since the bound of roughly 2.0346 was shown to be tight for online TSP [5], our new bound is the first time that DIAL-A-RIDE on the line can be strictly separated from online TSP in terms of competitive analysis. In addition, our bound is the currently best known lower bound even for general metric spaces. Specifically, we show the following.

▶ **Theorem 1.1.** *Let $\rho \approx 2.0585$ be the second largest root of the polynomial $4\rho^3-26\rho^2+39\rho-5$. There is no $(\rho-\varepsilon)$-competitive algorithm for open, non-preemptive ($c < \infty$) online* DIAL-A-RIDE *on the line for any $\varepsilon > 0$.*

Our construction is a non-trivial variation of the construction achieving roughly 2.0346 for online TSP [5]. This construction is comprised of an initial request, a first stage consisting in turn of different iterations, and a second stage. We show that, by using a proper transportation requests as initial requests, we can adapt a single iteration of the first stage as well as the second stage to achieve the bound of roughly 2.0585 in the DIAL-A-RIDE setting.

Our second result is an improved algorithm SMARTERSTART for online DIAL-A-RIDE on the line. This algorithm improves the waiting strategy of the SMARTSTART algorithm, which was identified as a weakness in [4]. We show that this modification improves the competitive ratio of the algorithm and give a tight analysis. Specifically, we show the following.

▶ **Theorem 1.2.** *The competitive ratio of* SMARTERSTART *is (roughly)* 2.6662.

The general idea of SMARTERSTART is to improve the tradeoff between the case when the algorithm waits before starting its final schedule and the case when it starts the final schedule immediately. Our modification of SMARTSTART significantly improves the performance in the former case, while only moderately degrading the performance in the latter case. Overall, this results in an improved worst-case performance.

Related Work. The online DIAL-A-RIDE problem has received considerable attention in the past (e.g. [1, 4, 5, 6, 9, 13]). Table 1 gives an overview of the currently best known bounds on the line for open online DIAL-A-RIDE and its special case open online TSP.

The following results are known for *closed* online DIAL-A-RIDE: For general metric spaces, the competitive ratio is exactly 2, both for online DIAL-A-RIDE as well as online TSP [1, 3, 9]. On the line, a better upper bound is known only for online TSP, where the competitive ratio is exactly $(9+\sqrt{17})/8 \approx 1.6404$ [3, 5]. The best known lower bound for closed, non-preemptive DIAL-A-RIDE on the line is 1.75 [5].

When the objective is to minimize the maximum flow time, on many metric spaces no online algorithm can be competitive [15, 16]. Hauptmeier et al. [12] showed that a competitive algorithm is possible if we restrict ourselves to instances with "reasonable" load. Yi and Tian [18] considered online DIAL-A-RIDE with deadlines, where the objective is to maximize the number of requests that are served in time. Other interesting variants of online DIAL-A-RIDE where destinations of requests are only revealed upon their collection were studied by Lipmann et al. [17] as well as Yi and Tian [19].

Table 1 Overview of the best known bounds for online Dial-a-Ride on the line (top), and online Dial-a-Ride on general metric spaces (bottom). Results are split into the non-preemptive case (with $c < \infty$), the preemptive case, and the TSP-case, where source and destination of each request coincide. Bold results are original, all other results follow immediately.

		open		closed	
		lower bound	upper bound	lower bound	upper bound
line	non-preemptive	**2.0585** (Thm 1.1)	**2.6662** (Thm 1.2)	**1.75** [5]	2
	preemptive	2.04	**2.41** [5]	1.64	2
	TSP	**2.04** [5]	**2.04** [5]	**1.64** [3]	**1.64** [5]
general	non-preemptive	2.0585 (Thm 1.1)	**3.41** [14]	2	**2** [1, 9]
	preemptive	2.04	3.41	2	2
	TSP	2.04	**2.5** [3]	**2** [3]	**2** [3]

For an overview of results for the offline version of Dial-a-Ride on the line, see [8]. Without release times, Gilmore and Gomory [10] and Atallah and Kosaraju [2] gave a polynomial time algorithm for closed, non-preemptive Dial-a-Ride on the line with capacity $c = 1$. Guan [11] showed that the closed, non-preemptive problem is hard for $c = 2$, and Bjelde et al. [5] extended this result for any finite capacity $c \geq 2$ in both the open and the closed variant. Bjelde et al. [5] also showed that the problem with release times is already hard for finite $c \geq 1$ in both variants, and Krumke [14] gave a 3-approximation algorithm for the closed variant. The complexity for the case $c = \infty$ remains open. For closed, preemptive Dial-a-Ride on the line without release times, Atallah and Kosaraju [2] gave a polynomial time algorithm for $c = 1$ and Guan [11] for $c \geq 2$. Charikar and Raghavachari [7] presented approximation algorithms for the closed case without release times on general metric spaces.

2 General Lower Bound

In this section, we prove Theorem 1.1. Let $c < \infty$ and Alg be a deterministic online algorithm for open online Dial-a-Ride. Let $\rho \approx 2.0585$, be the second largest root of the polynomial $4\rho^3 - 26\rho^2 + 39\rho - 5$. We describe a request sequence σ_ρ such that $\text{Alg}(\sigma_\rho) \geq \rho\text{Opt}(\sigma_\rho)$.

We first give a high-level description of our construction disregarding many technical details. Our construction is based on that in [5] for the TSP version of the problem. That construction consists of two *stages*: After an initial request $(1, 1; 1)$ (assuming w.l.o.g. Alg's position at time 1 is at most 0), the first stage starts. This stage consists of a loop, which ends as soon as two so-called critical requests are established. The second stage consists of augmenting the critical requests by suitable additional ones to show the desired competitive ratio. A single iteration of the loop only yields a lower bound of roughly 2.0298, but as the number of iterations approaches infinity one can show the tight bound of roughly 2.0346 in the limit.

In the Dial-a-Ride setting, we show a lower bound of roughly 2.0585 using the same general structure but only a single iteration. Our additional leeway stems from replacing the initial request $(1, 1; 1)$ with c initial requests of the form $(1, \delta; 1)$ where $\delta > 1$: At the time when an initial request is loaded, we show that w.l.o.g. all c requests are loaded and then proceed as we did when $(1, 1; 1)$ was served. In the new situation, the algorithm has to first deliver the c initial requests to be able to serve additional requests. For the optimum, the two situations however do not differ, because in the new situation there will be an additional

request to the right of δ later anyway. Interestingly, this leeway turns out to be sufficient not only to create critical requests (w.r.t. a slightly varied notion of criticality) for a competitive ratio of larger than 2.0298 but even strictly larger than 2.0346. The second stage has to be slightly adapted to match the new notion of criticality. It remains unclear how to use multiple iterations in our setting.

We start by making observations that will simplify the exposition. Consider a situation in which the server is fully loaded. First note that it is essentially irrelevant whether we assume that the server, without delivering any of the loaded requests, can still serve requests $(a_i, b_i; t_i)$ for which $a_i = b_i$: If it can, we simply move a_i and b_i by $\varepsilon > 0$ apart, forbidding the server to serve it before delivering one of the loaded requests first. Therefore, we assume for simplicity that, when fully loaded, the server has to first deliver a request before it can serve any other one. We note that, in our construction, the above idea can be implemented without loss, not even in terms of ε.

The latter discussion also motivates restricting the space of considered algorithms: We call Alg *eager* if it, when fully loaded with requests with identical destinations, immediately delivers these requests without detour. It is clear that we can transform every algorithm Alg' into an eager algorithm $\text{Alg}'_{\text{eager}}$ by letting it deliver the requests right away, waiting until Alg' would have delivered them, and then letting it continue like Alg'. Since Alg' cannot collect or serve other requests while being fully loaded, we have $\text{Alg}'_{\text{eager}}(\sigma) \leq \text{Alg}'(\sigma)$ for every request sequence σ.

▶ **Observation 2.1.** *Every algorithm for online* Dial-a-Ride *can be turned into an eager algorithm with the same competitive ratio.*

Thus, we may assume that Alg is eager. We now consider the second stage and then design a first stage to match the second stage. Suppose we have two requests $\sigma^R = (t^R, t^R; t^R)$ and $\sigma^L = (-t^L, -t^L, t^L)$ with $t^L \leq t^R$ to the right and to the left of the origin, respectively. We assume that Alg serves σ^R first at some time $t^* \geq (2\rho - 2)t^L + (\rho - 2)t^R$. Now suppose we could force Alg to serve σ^L directly after σ^R, even if additional requests are released. Then we could just release the request $\sigma^R_* = (t^R, t^R, 2t^L + t^R)$ and we would have

$$\text{Alg}(\sigma_\rho) = t^* + 2t^L + 2t^R \geq 2\rho t^L + \rho t^R = \rho \text{Opt}(\sigma_\rho),$$

since Opt can serve the three requests in time $2t^L + t^R$ by serving σ^L first. In fact, we will show that we can force Alg into this situation (or a worse situation) if the requests $\sigma^R = (t^R, t^R; t^R)$ and $\sigma^L = (-t^L, -t^L, t^L)$ satisfy the following properties. To describe the trajectory of a server, we use the notation "move(a)" for the tour that moves the server from its current position with unit speed to the point $a \in \mathbb{R}$.

▶ **Definition 2.2.** *We call the last two requests $\sigma^R = (t^R, t^R; t^R)$ and $\sigma^L = (-t^L, -t^L, t^L)$ of a request sequence with $0 < t^L \leq t^R$* critical *for* Alg *if the following conditions hold:*

- **(i)** *Both tours move$(-t^L) \oplus$move(t^R) and move$(t^R) \oplus$move$(-t^L)$ serve all requests presented until time t^R.*
- **(ii)** Alg *serves both σ^R and σ^L after time t^R and* Alg*'s position at time t^R lies between t^R and $-t^L$.*
- **(iii)** *If* Alg *serves σ^R before σ^L, it does so no earlier than $t^R_* := (2\rho - 2)t^L + (\rho - 2)t^R$.*
- **(iv)** *If* Alg *serves σ^L before σ^R, it does so no earlier than $t^L_* := (2\rho - 2)t^R + (\rho - 2)t^L$.*
- **(v)** *It holds that $\frac{t^R}{t^L} \leq \frac{4\rho^2 - 30\rho + 50}{-8\rho^2 + 50\rho - 66}$.*

▶ **Lemma 2.3.** *If there is a request sequence with two critical requests for* ALG*, we can release additional requests such that* ALG *is not* $(\rho-\varepsilon)$*-competitive on the resulting instance for any* $\varepsilon > 0$*.*

Definition 2.2 differs from [5, Definition 5] only in property (v), which is $\frac{t^R}{t^L} \leq 2$ in the original paper. Lemma 2.3 has been proved in [5, Lemma 6] for request sequences that satisfy the properties of [5, Definition 5], however, a careful inspection of the proof of [5, Lemma 6] shows that the statement of Lemma 2.3 also holds for request sequences that only satisfy (v) instead of $\frac{t^R}{t^L} \leq 2$. For a detailed proof, see Appendix A. Thus, our goal is to construct a request sequence σ_ρ that satisfies all properties of Definition 2.2.

The remaining part of this section focusses on establishing critical requests. There are no requests released until time 1. Without loss of generality, we assume that ALG's position at time 1 is $\text{pos}(1) \leq 0$ (the other case is symmetric). Here and throughout, we let $\text{pos}(t)$ denote the position of ALG's server at time t. Now, let

$$\delta := \frac{3\rho^2 - 11}{-3\rho^3 + 15\rho - 4}$$

and let c initial requests $\sigma^R_{(j)} = (1, \delta; 1)$ with $j \in \{1, \dots, c\}$ appear. These are the only requests appearing in the entire construction with a starting point differing from the destination. We make a basic observation on how ALG has to serve these requests.[1]

▶ **Lemma 2.4.** ALG *cannot collect any of the requests* $\sigma^R_{(j)}$ *before time* 2*. If* ALG *collects the requests after time* $\rho\delta - (\delta - 1)$ *or serves* $c' < c$ *requests before loading the remaining* $c - c'$*, it is not* $(\rho - \varepsilon)$*-competitive.*

We hence may assume that ALG loads all c requests $\sigma^R_{(j)}$ at the same time. Let $t^L \in [2, \rho\delta - (\delta - 1))$ be the time ALG loads the c requests $\sigma^R_{(j)}$. We start the first stage and present a variant of a single iteration of the construction in [5]: We let the request $\sigma^L = (-t^L, -t^L; t^L)$ appear and define the function

$$\ell(t) = (4 - \rho) \cdot t - (2\rho - 2) \cdot t^L,$$

which can be viewed as a line in the path-time diagram. Because of $\rho > 2$, we have $\ell(t^L) = (6 - 3\rho)t^L < 0 < \text{pos}\left(t^L\right)$, i.e., ALG's position at time t^L is to the right of the line ℓ. Thus, ALG crosses the line ℓ before it serves σ^L. Let t^R be the time ALG crosses ℓ for the first time and let the request $\sigma^R = (t^R, t^R; t^R)$ appear. Assume ALG crosses the line ℓ and serves σ^R before σ^L. Then it does not serve σ^R before time

$$t^R + |\ell(t^R) - t^R| = (2\rho - 2)t^L + (\rho - 2)t^R = t^R_*. \tag{1}$$

Now assume ALG crosses ℓ at time $t^R \geq \frac{3\rho-5}{7-3\rho}t^L$ and serves σ^L before σ^R. Then it does not serve serve σ^L before time

$$\begin{aligned} t^R + |\ell(t^R) - (-t^L)| &= (5 - \rho)t^R - (2\rho - 3)t^L \\ &\geq (2\rho - 2)t^R + (7 - 3\rho)\frac{3\rho - 5}{7 - 3\rho}t^L - (2\rho - 3)t^L \\ &= (2\rho - 2)t^R + (\rho - 2)t^L = t^L_*. \end{aligned} \tag{2}$$

The following lemma shows that the two requests cannot be served before these respective times by establishing that indeed $t^R \geq \frac{3\rho-5}{7-3\rho}t^L$.

[1] The full proof and other omitted proofs can be found at `http://arxiv.org/abs/1907.02858`.

▶ **Lemma 2.5.** ALG *can neither serve* σ^L *before time* t^L_* *nor can it serve* σ^R *before time* t^R_*.

Proof. Since ALG is eager, it delivers the c requests $\sigma^R_{(j)}$ without waiting or detour, i.e., we have $\text{pos}\left(t^L + (\delta - 1)\right) = \delta$. Furthermore, we have

$$\begin{aligned}
\ell(t^L + (\delta - 1)) &= (4-\rho)(t^L + (\delta - 1)) - (2\rho - 2)t^L \\
&= (6 - 3\rho)t^L + (4 - \rho)(\delta - 1) \\
&\leq (6 - 3\rho)(\rho\delta - (\delta - 1)) + (4 - \rho)(\delta - 1) \\
&= \frac{3\rho^4 - 18\rho^3 + 3\rho^2 + 50\rho - 14}{3\rho^3 - 15\rho + 4} \\
&\overset{\rho < 2.06}{<} \delta = \text{pos}\left(t^L + (\delta - 1)\right),
\end{aligned}$$

i.e., ALG's position at time $t^L + (\delta - 1)$ is to the right of ℓ. The earliest possible time ALG crosses ℓ is the solution of

$$\ell(t^R) = (4-\rho)t^R - (2\rho - 2)t^L = \text{pos}\left(t^L + (\delta - 1)\right) + t^L + (\delta - 1) - t^R,$$

which is $t^R = \frac{2\rho - 1}{5 - \rho}t^L + \frac{2\delta - 1}{5 - \rho}$. The inequality

$$\begin{aligned}
\left(\frac{3\rho - 5}{7 - 3\rho} - \frac{2\rho - 1}{5 - \rho}\right) t^L &= \frac{3\rho^2 + 3\rho - 18}{3\rho^2 - 22\rho + 35} t^L \\
&\leq \frac{3\rho^2 + 3\rho - 18}{3\rho^2 - 22\rho + 35}(\rho\delta - (\delta - 1))) \\
&= \frac{3\rho^3 + 6\rho^2 - 15\rho - 18}{3\rho^4 - 15\rho^3 - 15\rho^2 + 79\rho - 20} \\
&= \frac{2\delta - 1}{5 - \rho},
\end{aligned}$$

implies that we have

$$t^R \geq \frac{3\rho - 5}{7 - 3\rho} t^L. \tag{3}$$

Because of inequality (1) ALG does not serve σ^R before t^R_* and because of the inequalities (3) and (2) it does not serve σ^L before time t^L_*. ◀

In fact, also the other properties of critical requests are satisfied.

▶ **Lemma 2.6.** *The requests* σ^R *and* σ^L *of the request sequence* σ_ρ *are critical.*

Proof. We have to show that the requests σ^R and σ^L of the request sequence σ_ρ satisfy the properties (i) to (v) of Definition 2.2. The release time of every request is equal to its starting position, thus every request can be served/loaded immediately once its starting position is visited and (i) of Definition 2.2 is satisfied. At time t^R ALG has not served σ^R, because for that it would have needed to go right from time 0 on; it has not served σ^L either, because during the period of time $[t_L, t_R]$ ALG and σ^L were on different sides of ℓ. This establishes the first part of (ii) of Definition 2.2. Furthermore at time t^R ALG is at position $\text{pos}\left(t^R\right) = (4-\rho)t^R - (2\rho - 2)t^L$ with

$$-t^L \leq (4-\rho)t^R - (2\rho - 2)t^L \leq t^R$$

Therefore, the second part of (ii) of Definition 2.2 is satisfied as well.

Lemma 2.5 shows that (iii) and (iv) of Definition 2.2 are satisfied. It remains to show that property (v) is satisfied. For this we need to examine the release time t^R of σ^R. The time t^R is largest if ALG tries to avoid crossing the line ℓ for as long as possible, i.e., it continues to move right after serving the requests $\sigma^R_{(j)}$. Then, we have $\operatorname{pos}(t) = 1 - t^L + t$ for $t \in [t^L, t^R]$ and t^R is the solution of

$$1 - t^L + t^R = (4-\rho)t^R - (2\rho - 2)t^L.$$

Thus, in general, we have $t^R \leq \frac{2\rho-3}{3-\rho}t^L + \frac{1}{3-\rho}$, i.e.,

$$\frac{t^R}{t^L} \leq \frac{2\rho-3}{3-\rho} + \frac{1}{(3-\rho)t^L} \overset{t^L \geq 2}{\leq} \frac{4\rho-5}{6-2\rho}. \tag{4}$$

For property (v), we need $\frac{t^R}{t^L} \leq \frac{4\rho^2-30\rho+50}{-8\rho^2+50\rho-66}$. This is satisfied if

$$\frac{4\rho-5}{6-2\rho} \leq \frac{4\rho^2-30\rho+50}{-8\rho^2+50\rho-66},$$

which is equivalent to

$$4\rho^3 - 26\rho^2 + 39\rho - 5 \geq 0,$$

which is true by definition of ρ. ◀

Together with Lemma 2.3, this completes the proof of Theorem 1.1.

3 An Improved Algorithm

One of the simplest approaches for an online algorithm to solve DIAL-A-RIDE is the following: Always serve the set of currently unserved requests in an optimum offline schedule and ignore all new incoming request while doing so. Afterwards, repeat this procedure with all ignored unserved requests until no new requests arrive. This simple algorithm that is often called IGNORE [1] has a competitive ratio of exactly 4 [4, 14]. The main weakness of IGNORE is that it always starts its schedule immediately. Ascheuer et al. showed that it is beneficial if the server waits sometimes before starting a schedule and introduced the SMARTSTART algorithm [1], which has a competitive ratio of roughly 2.94 [4].

We define $L(t, p, R)$ to be the smallest makespan of a schedule that starts at position p at time t and serves all requests in $R \subseteq \sigma$ after they appeared (i.e., the schedule must respect release times). For the description of online algorithms, we denote by t the current time and by R_t the set of requests that have appeared until time t but have not been served yet.

The algorithm SMARTSTART is given in Algorithm 1. Essentially, at time t, SMARTSTART waits before starting an optimal schedule to serve all available requests at time

$$\min_{t' \geq t} \left\{ t' \geq \frac{L(t', p, R_{t'})}{\Theta - 1} \right\}, \tag{5}$$

where p is the current position of the server and $\Theta > 1$ is a parameter of the algorithm that scales the waiting time. Importantly, like IGNORE, SMARTSTART ignores incoming requests while executing a schedule.

Birx and Disser identified that SMARTSTART's waiting routine defined by inequality (5) has a critical weakness [4, Lemma 4.1]. It is possible to lure the server to any position q in time $q + \varepsilon$ for every $\varepsilon > 0$. Roughly speaking, a request $\sigma_1 = ((\Theta-1)\varepsilon, (\Theta-1)\varepsilon; (\Theta-1)\varepsilon)$ is

Algorithm 1 SMARTSTART.

```
p_1 ← 0
for j = 1, 2, ... do
    while current time t < L(t, p_j, R_t)/(Θ − 1) do
        wait
    t_j ← t
    S_j ← optimal offline schedule serving R_t starting from p_j
    execute S_j
    p_{j+1} ← current position
```

released first and then for every $i \in \{2, \ldots, \frac{q}{\varepsilon}\}$ a request $\sigma_i = (i\varepsilon, i\varepsilon; i\varepsilon)$ follows. The schedule to serve the request σ_1 is started at time ε and finished at time 2ε. The schedule to serve the request at position $i\varepsilon$ is not started earlier than time

$$\frac{L(i\varepsilon, (i-1)\varepsilon, \{\sigma_i\})}{\Theta - 1} = \frac{|(i-1)\varepsilon - i\varepsilon|}{\Theta - 1} = \frac{\varepsilon}{\Theta - 1}. \tag{6}$$

This time is (depending on the choice of Θ) later than the current time $i\varepsilon$ for every $i \geq 2$. Thus there is no waiting time for any schedule except the first one and the server reaches position q at time $q + \varepsilon$. We see that the request sequence to lure the server away heavily uses that inequality (5) relies on SMARTSTART's current position p, when computing the waiting time. Thus, we modify the waiting routine of SMARTSTART to avoid luring accordingly. Denote by $\sigma_{\leq t}$ the set of requests that have been released until time t.

Algorithm 2 SMARTERSTART.

```
p_1 ← 0
for j = 1, 2, ... do
    while current time t < L(t, 0, σ_{≤t})/(Θ − 1) do
        wait
    t_j ← t
    S_j ← optimal offline schedule serving R_t starting from p_j
    execute S_j
    p_{j+1} ← current position
```

The improved algorithm SMARTERSTART is given in Algorithm 2. At time t, it waits before starting an optimal schedule to serve all available requests at time

$$\min_{t' \geq t} \left\{ t' \geq \frac{L(t', 0, \sigma_{\leq t'})}{\Theta - 1} \right\}. \tag{7}$$

Again, $\Theta > 1$ is a parameter of the algorithm that scales the waiting time. In contrast to SMARTSTART, the waiting time is dependent on the length of the optimum offline schedule serving all requests appeared until the current time and starting from the origin. This guarantees that the server cannot be forced to reach any position q before time $q/(\Theta - 1)$ since we always have $L(t, 0, \sigma_{\leq t}) > q$ if $\sigma_{\leq t}$ contains a request with destination in position q.

Whenever we need to distinguish the behavior of SMARTERSTART for different values of $\Theta > 1$, we write SMARTERSTART$_\Theta$ to make the choice of Θ explicit. The length of SMARTERSTART's trajectory is denoted by SMARTERSTART(σ). Note that the schedules used by IGNORE, SMARTSTART and SMARTERSTART are NP-hard to compute for $1 < c < \infty$, see [5].

We let $N \in \mathbb{N}$ be the number of schedules needed by SmarterStart to serve σ. The j-th schedule is denoted by S_j, its starting time by t_j, its starting point by p_j, its ending point by p_{j+1}, and the set of requests served in S_j by σ_{S_j}. For convenience, we set $t_0 = p_0 = 0$.

3.1 Upper Bound for SmarterStart

We show the upper bound of Theorem 1.2. The completion time of SmarterStart is

$$\textsc{SmarterStart}(\sigma) = t_N + L(t_N, p_N, \sigma_{S_N}). \tag{8}$$

First, observe that, for all $0 \le t \le t'$, $p, p' \in \mathbb{R}$, and $R \subseteq \sigma$, we have

$$L(t, p, R) \ge L(t', p, R), \tag{9}$$
$$L(t, p, R) \le |p - p'| + L(t, p', R), \tag{10}$$
$$L(t, 0, \sigma_{\le t}) \le L(t, 0, \sigma) \le L(0, 0, \sigma) \le \textsc{Opt}(\sigma). \tag{11}$$

Similar to [4], we distinguish between two cases, depending on whether or not SmarterStart waits after finishing schedule S_{N-1} and before starting the final schedule S_N. If the algorithm SmarterStart waits, the starting time of schedule S_N is given by

$$t_N = \frac{1}{\Theta - 1} L(t_N, 0, \sigma_{\le t_N}), \tag{12}$$

otherwise, we have

$$t_N = t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}). \tag{13}$$

We start by giving a lower bound on the starting time of a schedule. It was shown in [4] that the schedule S_j of Smartstart is never started earlier than time $\frac{|p_{j+1}|}{\Theta}$. This changes slightly for SmarterStart.[1]

▶ **Lemma 3.1.** *Algorithm* SmarterStart *does not start schedule S_j earlier than time $\frac{|p_{j+1}|}{\Theta - 1}$, i.e., we have $t_j \ge \frac{|p_{j+1}|}{\Theta - 1}$.*

Using Lemma 3.1, we can give an upper bound on the length of SmarterStart's schedules, which is an essential ingredient in our upper bounds. The following lemma is proved similarly to [4, Lemma 3.2], which yields an upper bound of $(1 + \frac{\Theta}{\Theta+2})\textsc{Opt}(\sigma)$ for the length of every schedule S_j of Smartstart.[1]

▶ **Lemma 3.2.** *For every schedule S_j of* SmarterStart*, we have*

$$L(t_j, p_j, \sigma_{S_j}) \le \left(1 + \frac{\Theta - 1}{\Theta + 1}\right) \textsc{Opt}(\sigma).$$

Proof sketch. To proof the claim we have to show the two inequalities

$$L(t_j, p_j, \sigma_{S_j}) \le \textsc{Opt}(\sigma) + |p_j| \quad \text{and} \quad L(t_j, p_j, \sigma_{S_j}) \le 2\textsc{Opt}(\sigma) - 2\frac{|p_j|}{\Theta - 1}. \tag{14}$$

This implies

$$\begin{aligned} L(t_j, p_j, \sigma_{S_j}) &\overset{(14)}{\le} \min\left\{\textsc{Opt}(\sigma) + |p_j|, 2\textsc{Opt}(\sigma) - \frac{2}{\Theta - 1}|p_j|\right\} \\ &\le \left(1 + \frac{\Theta - 1}{\Theta + 1}\right) \textsc{Opt}(\sigma), \end{aligned}$$

since the minimum above is largest for $|p_j| = \frac{\Theta - 1}{\Theta + 1}\textsc{Opt}(\sigma)$. ◀

The following proposition uses Lemma 3.2 to provide an upper bound for the competitive ratio of SmarterStart, in the case that SmarterStart does have a waiting period before starting the final schedule.

▶ **Proposition 3.3.** *In case* SmarterStart *waits before executing S_N, we have*

$$\frac{\textsc{SmarterStart}(\sigma)}{\textsc{Opt}(\sigma)} \leq f_1(\Theta) := \frac{2\Theta^2 - \Theta + 1}{\Theta^2 - 1}.$$

Proof. Assume SmarterStart waits before starting the final schedule. Lemma 3.2 yields the claimed bound:

$$\begin{aligned}
\textsc{SmarterStart}(\sigma) &\overset{(8)}{=} t_N + L(t_N, p_N, \sigma_{S_N}) \\
&\overset{(12)}{=} \frac{1}{\Theta - 1} L(t_N, 0, \sigma_{\leq t_N}) + L(t_N, p_N, \sigma_{S_N}) \\
&\overset{(11)}{\leq} \frac{1}{\Theta - 1} \textsc{Opt}(\sigma) + L(t_N, p_N, \sigma_{S_N}) \\
&\overset{\text{Lem. 3.2}}{\leq} \left(\frac{1}{\Theta - 1} + 1 + \frac{\Theta - 1}{\Theta + 1} \right) \textsc{Opt}(\sigma) \\
&= \frac{2\Theta^2 - \Theta + 1}{\Theta^2 - 1} \textsc{Opt}(\sigma).
\end{aligned}$$

◀

In comparison, the upper bound for the competitive ratio of Smartstart, in case Smartstart has a waiting period before starting the final schedule is $\frac{2\Theta^2+2\Theta}{\Theta^2+\Theta-2}\textsc{Opt}(\sigma)$ [4, Proposition 3.2]. Note that SmarterStart's bound is better than Smartstart's bound for $\Theta > 1$.

It remains to examine the case that the algorithm SmarterStart has no waiting period before starting the final schedule. For this we use two lemmas from [4] originally proved for Smartstart, which are still valid for SmarterStart since they give bounds on the optimum offline schedules independently of the waiting routine.

By $x_- := \min\{0, \min_{i=1,\dots,n}\{a_i\}, \min_{i=1,\dots,n}\{b_i\}\}$ we denote the leftmost position that needs to be visited by the server and by $x_+ := \max\{0, \max_{i=1,\dots,n}\{a_i\}, \max_{i=1,\dots,n}\{b_i\}\}$ the rightmost. We denote by $y_-^{S_j}$ the leftmost and by $y_+^{S_j}$ the rightmost position that occurs in the requests σ_{S_j}. Note that $y_-^{S_j}$ and $y_+^{S_j}$ need not lie on different sides of the origin, in contrast to $x_{-/+}$.

▶ **Lemma 3.4** (Lemma 3.4, Full Version of [4])**.** *Let S_j with $j \in \{1, \dots, N\}$ be a schedule of* SmarterStart*. Moreover, let $\textsc{Opt}(\sigma) = |x_-| + x_+ + y$ for some $y \geq 0$. Then, we have*

$$L(t_j, 0, \sigma_{S_j}) \leq |\min\{0, y_-^{S_j}\}| + \max\{0, y_+^{S_j}\} + y.$$

▶ **Lemma 3.5** (Lemma 3.6, Full Version of [4])**.** *Let S_j with $j \in \{1, \dots, N\}$ be a schedule of* SmarterStart*. Moreover, let $|x_-| \leq x_+$ and $\textsc{Opt}(\sigma) = |x_-| + x_+ + y$ for some $y \geq 0$. Then, for every point p that is visited by S_j we have*

$$p \leq |p_j| + |p_j - p_{j+1}| + y - |\min\{0, y_-^{S_j}\}|.$$

Using the bounds established by Lemma 3.4 and Lemma 3.5, we can give an upper bound for the competitive ratio of SmarterStart if the server is not waiting before starting the final schedule.

▶ **Proposition 3.6.** *If* SmarterStart *does not wait before executing S_N, we have*

$$\frac{\textsc{SmarterStart}(\sigma)}{\textsc{Opt}(\sigma)} \leq f_2(\Theta) := \frac{3\Theta^2 + 3}{2\Theta + 1}.$$

Proof. Assume algorithm SmarterStart does not have a waiting period before the last schedule, i.e., SmarterStart starts the final schedule S_N immediately after finishing S_{N-1}. Without loss of generality, we assume $|x_-| \leq x_+$ throughout the entire proof by symmetry.

First of all, we notice that we may assume that SmarterStart executes at least two schedules in this case. Otherwise either the only schedule has length 0, which would imply $\textsc{Opt}(\sigma) = \textsc{SmarterStart}(\sigma) = 0$, or the only schedule would have a positive length, implying a waiting period. Let $\sigma_{S_N}^{\textsc{Opt}}$ be the first request of σ_{S_N} that is served by Opt and let $a_N^{\textsc{Opt}}$ be its starting point and $r_N^{\textsc{Opt}}$ be its release time. We have

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(8)}{=} t_N + L(t_N, p_N, \sigma_{S_N}) \\ &\overset{(13)}{=} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + L(t_N, p_N, \sigma_{S_N}) \\ &\overset{t_N \geq r_N^{\textsc{Opt}}}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + L(r_N^{\textsc{Opt}}, p_N, \sigma_{S_N}). \end{aligned} \tag{15}$$

Since Opt serves all requests of σ_{S_N} after time $r_N^{\textsc{Opt}}$, starting with a request with starting point $a_N^{\textsc{Opt}}$, we also have

$$\textsc{Opt}(\sigma) \geq r_N^{\textsc{Opt}} + L(r_N^{\textsc{Opt}}, a_N^{\textsc{Opt}}, \sigma_{S_N}). \tag{16}$$

Furthermore, we have

$$r_N^{\textsc{Opt}} > t_{N-1} \tag{17}$$

since otherwise $\sigma_{S_N}^{\textsc{Opt}} \in \sigma_{S_{N-1}}$ would hold. This gives us

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(15)}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + L(r_N^{\textsc{Opt}}, p_N, \sigma_{S_N}) \\ &\overset{(10)}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| \\ &\quad + L(r_N^{\textsc{Opt}}, a_N^{\textsc{Opt}}, \sigma_{S_N}) \\ &\overset{(16)}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| \\ &\quad + \textsc{Opt}(\sigma) - r_N^{\textsc{Opt}} \\ &\overset{(17)}{<} L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| + \textsc{Opt}(\sigma) \end{aligned} \tag{18}$$

$$\begin{aligned} &\overset{(10)}{\leq} |p_{N-1}| + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| + \textsc{Opt}(\sigma) \\ &\overset{\text{Lem. 3.1}}{\leq} (\Theta - 1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| + \textsc{Opt}(\sigma). \end{aligned} \tag{19}$$

We have

$$\textsc{Opt}(\sigma) \geq t_{N-2} + |a_N^{\textsc{Opt}} - p_N|, \tag{20}$$

because Opt has to visit both $a_N^{\textsc{Opt}}$ and p_N after time t_{N-2}: It has to visit $a_N^{\textsc{Opt}}$ to collect $\sigma_{S_N}^{\textsc{Opt}}$ and it has to visit p_N to deliver some request of $\sigma_{S_{N-1}}$. Using the above inequalitiy, we get

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(19)}{<} (\Theta - 1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| + \textsc{Opt}(\sigma) \\ &\overset{(20)}{\leq} 2\textsc{Opt}(\sigma) + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + (\Theta - 2)t_{N-2}. \end{aligned} \tag{21}$$

In the case $\Theta \geq 2$, we have

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(21)}{<} 2\textsc{Opt}(\sigma) + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + (\Theta - 2)t_{N-2} \\ &\overset{(11)}{\leq} (\Theta + 1)\textsc{Opt}(\sigma) \\ &\overset{\Theta \geq 2}{\leq} \frac{3\Theta^2 + 3}{2\Theta + 1}\textsc{Opt}(\sigma).\end{aligned}$$

Thus, we may assume $\Theta < 2$. Similarly as in inequality (21), we get

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(19)}{<} (\Theta - 1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| + \textsc{Opt}(\sigma) \\ &\overset{(20)}{\leq} \Theta\textsc{Opt}(\sigma) + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N| \\ &\overset{(7)}{\leq} \Theta\textsc{Opt}(\sigma) + (\Theta - 1)t_{N-1} + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N| \\ &\leq (2\Theta - 1)\textsc{Opt}(\sigma) + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N|, \end{aligned} \tag{22}$$

where the last inequality follows, because there exists a request in σ with release date later than t_{N-1}. This means the claim is shown if we have

$$|p_N - a_N^{\textsc{Opt}}| \leq \textsc{Opt}(\sigma) - \frac{\Theta - 1}{2\Theta + 1}\textsc{Opt}(\sigma) \tag{23}$$

since then we have

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(22)}{<} (2\Theta - 1)\textsc{Opt}(\sigma) + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N| \\ &\overset{(23)}{\leq} (2\Theta - 1)\textsc{Opt}(\sigma) + (2 - \Theta)\left(1 - \frac{\Theta - 1}{2\Theta + 1}\right)\textsc{Opt}(\sigma) \\ &= \frac{3\Theta^2 + 3}{2\Theta + 1}\textsc{Opt}(\sigma).\end{aligned}$$

Therefore, we may assume in the following that

$$|p_N - a_N^{\textsc{Opt}}| > \textsc{Opt}(\sigma) - \frac{\Theta - 1}{2\Theta + 1}\textsc{Opt}(\sigma). \tag{24}$$

Let $\textsc{Opt}(\sigma) = |x_-| + x_+ + y$ for some $y \geq 0$. By definition of x_- and x_+ we have

$$|p_N - a_N^{\textsc{Opt}}| + y \leq \textsc{Opt}(\sigma). \tag{25}$$

In the case that Opt visits position p_N before it collects $\sigma_{S_N}^{\textsc{Opt}}$, we have

$$|a_N^{\textsc{Opt}} - p_N| + |p_N| \leq \textsc{Opt}(\sigma). \tag{26}$$

Similarly, if Opt collects $\sigma_{S_N}^{\textsc{Opt}}$ before it visits position p_N for the first time, we have

$$\begin{aligned}\textsc{Opt}(\sigma) &\geq r_N^{\textsc{Opt}} + |a_N^{\textsc{Opt}} - p_N| \\ &\overset{(17)}{>} t_{N-1} + |a_N^{\textsc{Opt}} - p_N| \\ &\overset{\text{Lem. 3.1}}{\geq} \frac{|p_N|}{\Theta - 1} + |a_N^{\textsc{Opt}} - p_N| \\ &\overset{\Theta < 2}{\geq} |p_N| + |a_N^{\textsc{Opt}} - p_N|.\end{aligned}$$

Thus, inequality (26) holds in general. To sum it up, we may assume that

$$\max\{y, |p_N|, t_{N-2}\} \overset{(24),(25),(26),(20)}{<} \frac{\Theta-1}{2\Theta+1}\textsc{Opt}(\sigma) \tag{27}$$

holds. In the following, denote by $y_-^{S_{N-1}}$ the leftmost starting or ending point and by $y_+^{S_{N-1}}$ the rightmost starting or ending point of the requests in $\sigma_{S_{N-1}}$. We compute

$$\begin{aligned}
\textsc{SmarterStart}(\sigma) &\overset{(18)}{<} L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |p_N - a_N^{\textsc{Opt}}| + \textsc{Opt}(\sigma)\\
&\overset{(26)}{<} L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + 2\textsc{Opt}(\sigma) - |p_N|\\
&\overset{(9)}{\leq} |p_{N-1}| + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + 2\textsc{Opt}(\sigma) - |p_N|\\
&\overset{\text{Lem. 3.1}}{\leq} (\Theta-1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + 2\textsc{Opt}(\sigma) - |p_N|\\
&\overset{\text{Lem. 3.4}}{\leq} (\Theta-1)t_{N-2} + \max\{0, |y_-^{S_{N-1}}|\} + \max\{0, y_+^{S_{N-1}}\} + y\\
&\qquad +2\textsc{Opt}(\sigma) - |p_N|.
\end{aligned} \tag{28}$$

Obviously, position $y_+^{S_{N-1}}$ is visited by SmarterStart in schedule S_{N-1}. Therefore, $y_+^{S_{N-1}}$ is smaller than or equal to the rightmost point that is visited by SmarterStart during schedule S_{N-1}, which gives us

$$y_+^{S_{N-1}} \overset{\text{Lem. 3.5}}{\leq} |p_{N-1}| + |p_{N-1} - p_N| + y - \max\{0, |y_-^{S_{N-1}}|\}. \tag{29}$$

On the other hand, because of $|x_-| \leq x_+$, we have $\textsc{Opt}(\sigma) \geq 2|x_-| + x_+$, which implies $y \geq |x_-|$. By definition of x_- and $y_-^{S_{N-1}}$, we have $|x_-| \geq \max\{0, |y_-^{S_{N-1}}|\}$. This gives us $y \geq \max\{0, |y_-^{S_{N-1}}|\}$ and

$$0 \leq |p_{N-1}| + |p_{N-1} - p_N| + y - \max\{0, |y_-^{S_{N-1}}|\}. \tag{30}$$

To sum it up, we have

$$\max\{0, y_+^{S_{N-1}}\} \overset{(29),(30)}{\leq} |p_{N-1}| + |p_{N-1} - p_N| + y - \max\{0, |y_-^{S_{N-1}}|\}. \tag{31}$$

The inequality above gives us

$$\begin{aligned}
\textsc{SmarterStart}(\sigma) &\overset{(28)}{<} (\Theta-1)t_{N-2} + \max\{0, |y_-^{S_{N-1}}|\} + \max\{0, y_+^{S_{N-1}}\}\\
&\qquad +y + 2\textsc{Opt}(\sigma) - |p_N|\\
&\overset{(31)}{\leq} (\Theta-1)t_{N-2} + |p_{N-1}| + |p_{N-1} - p_N| + 2y + 2\textsc{Opt}(\sigma) - |p_N|\\
&\leq (\Theta-1)t_{N-2} + |p_{N-1}| + |p_{N-1}| + |p_N| + 2y + 2\textsc{Opt}(\sigma) - |p_N|\\
&\overset{\text{Lem. 3.1}}{\leq} (\Theta-1)t_{N-2} + 2(\Theta-1)t_{N-2} + 2y + 2\textsc{Opt}(\sigma)\\
&\overset{(27)}{\leq} (3\Theta-3)\frac{\Theta-1}{2\Theta+1}\textsc{Opt}(\sigma) + 2\frac{\Theta-1}{2\Theta+1}\textsc{Opt}(\sigma) + 2\textsc{Opt}(\sigma)\\
&= \frac{3\Theta^2+3}{2\Theta+1}\textsc{Opt}(\sigma).
\end{aligned}$$

◀

In comparison, the upper bound for the competitive ratio of SmartStart in case it does not have a waiting period before starting the final schedule is $\Theta + 1 - \frac{\Theta-1}{3\Theta+3}\textsc{Opt}(\sigma)$ [4, Proposition 3.4]. Note that SmarterStart's bound is slightly worse than SmartStart's bound for $\Theta > 1.47$. However, in combination with the bound of Proposition 3.3, SmarterStart has a better worst-case than SmartStart.

▶ **Theorem 3.7.** *Let Θ^* be the largest solution of $f_1(\Theta) = f_2(\Theta)$, i.e.,*

$$\frac{3\Theta^{*2}+3}{2\Theta^*+1} = \frac{2\Theta^{*2}-\Theta^*+1}{\Theta^{*2}-1}.$$

Then, $\textsc{SmarterStart}_{\Theta^*}$ is ρ^*-competitive with $\rho^* := f_1(\Theta^*) = f_2(\Theta^*) \approx 2.6662$.

Proof. According to Proposition 3.3 and Proposition 3.6, if it exists,

$$\Theta^* = \operatorname*{argmin}_{\Theta>1} \{\max\{f_1(\Theta), f_2(\Theta)\}\}$$

is the parameter for SmarterStart with the smallest upper bound. We note that f_1 is strictly decreasing for $\Theta > 1$ and that f_2 is strictly increasing for $\Theta > 1$. Therefore, if an intersection point of f_1 and f_2 that is larger than 1 exists, then this is at Θ^*. Indeed, the intersection point exists, which is the largest solution of

$$\frac{3\Theta^2+3}{2\Theta+1} = \frac{2\Theta^2-\Theta+1}{\Theta^2-1}.$$

The resulting upper bound for the competitive ratio is

$$\rho^* = f_1(\Theta^*) = f_2(\Theta^*) \approx 2.6662.$$ ◀

3.2 Lower Bound for SmarterStart

We show the lower bound of Theorem 1.2. In this section, we explicitly construct instances that demonstrate that the upper bounds given in the previous section are tight for certain ranges of $\Theta > 1$, in particular for $\Theta = \Theta^*$ (as in Theorem 3.7). Further, we show that choices of $\Theta > 1$ different from Θ^* yield competitive ratios worse than $\rho^* \approx 2.67$. Together, this implies that ρ^* is exactly the best possible competitive ratio for SmarterStart.[1]

▶ **Proposition 3.8.** *Let $1 < \Theta < 2$. For every sufficiently small $\varepsilon > 0$, there is a set of requests σ such that* SmarterStart *waits before starting the final schedule and such that the inequality*

$$\frac{\textsc{SmarterStart}(\sigma)}{\textsc{Opt}(\sigma)} \geq \frac{2\Theta^2-\Theta+1}{\Theta^2-1} - \varepsilon$$

holds, i.e., the upper bound established in Proposition 3.3 is tight for $\Theta \in (1,2)$.

Proof sketch. Let $\varepsilon > 0$ with $\varepsilon < \frac{\Theta}{\Theta+1}$ and $\varepsilon' = \frac{\Theta+1}{2\Theta}\varepsilon$. The request sequence $\sigma = \{\sigma_1, \sigma_2\}$ with

$$\sigma_1 = (1, 1; 0) \qquad \text{and} \qquad \sigma_2 = (-\frac{1}{\Theta-1} + \varepsilon', 1; \frac{1}{\Theta-1} + \varepsilon')$$

achieves the desired result. ◀

▶ **Proposition 3.9.** *Let $\frac{1}{2}(1+\sqrt{5}) \leq \Theta \leq 2$. For every sufficiently small $\varepsilon > 0$ there is a set of requests σ such that* SmarterStart *immediately starts S_N after S_{N-1} and such that*

$$\frac{\textsc{SmarterStart}(\sigma)}{\textsc{Opt}(\sigma)} \geq \frac{3\Theta^2+3}{2\Theta+1} - \varepsilon,$$

i.e., the upper bound established in Proposition 3.6 is tight for $\Theta \in [\frac{1}{2}(1+\sqrt{5}), 2] \approx [1.6180, 2]$.

Proof sketch. Let $\varepsilon > 0$ with $\varepsilon < \frac{1}{4}(\frac{5\Theta^2-9\Theta+4}{2\Theta+1})$ and $\varepsilon' = \frac{2\Theta+1}{5\Theta^2-9\Theta+4}\varepsilon$. The request sequence $\sigma = \{\sigma_1, \sigma_2\}$ with

$$\begin{aligned}
\sigma_1 &= (1, 1; 0),\\
\sigma_2^{(1)} &= \left(2 + \frac{1}{\Theta - 1} - 2\varepsilon', 2 + \frac{1}{\Theta - 1} - 2\varepsilon'; \frac{1}{\Theta - 1} + \varepsilon'\right),\\
\sigma_2^{(2)} &= \left(-\frac{1}{\Theta - 1}, -\frac{1}{\Theta - 1}; \frac{1}{\Theta - 1} + \varepsilon'\right),\\
\sigma_3 &= \left(\frac{3}{(\Theta - 1)^2} - \varepsilon', \frac{3}{(\Theta - 1)^2} - \varepsilon'; \frac{3}{(\Theta - 1)^2} + \frac{2}{\Theta - 1}\right)
\end{aligned}$$

achieves the desired result. ◀

Recall that the optimal parameter Θ^* established in Theorem 3.7 is the only positive, real solution of the equation

$$\frac{3\Theta^2 + 3}{2\Theta + 1} = \frac{2\Theta^2 - \Theta + 1}{\Theta^2 - 1},$$

which is $\Theta^* \approx 1.7125$. Therefore, according to Proposition 3.8 and Proposition 3.9 the parameter Θ^* lies in the range where the upper bounds of Propositions 3.3 and 3.6 are both tight. It remains to make sure that for all Θ that lie outside of this range the competitive ratio of SmarterStart$_\Theta$ is larger than $\rho^* \approx 2.6662$.[1]

▶ **Lemma 3.10.** *Let $\Theta > 2$. There is a set of requests $\sigma_{\Theta>2}$ such that*

$$\frac{\textsc{SmarterStart}(\sigma_{\Theta>2})}{\textsc{Opt}(\sigma_{\Theta>2})} > \rho^* \approx 2.6662.$$

Figure 1 shows the upper and lower bounds that we have established. Theorem 1.2 now follows from Theorem 3.7 combined with Propositions 3.8 and 3.9, as well as Lemma 3.10.

Proof of Theorem 1.2. We have shown in Proposition 3.8 that the upper bound

$$\frac{\textsc{SmarterStart}(\sigma)}{\textsc{Opt}(\sigma)} \leq f_1(\Theta) = \frac{2\Theta^2 - \Theta + 1}{\Theta^2 - 1}$$

established in Proposition 3.3 for the case, where SmarterStart waits before starting the final schedule, is tight for all $\Theta \in (1, 2)$. Furthermore, we have shown in Proposition 3.9 that the upper bound

$$\frac{\textsc{SmarterStart}(\sigma)}{\textsc{Opt}(\sigma)} \leq f_2(\Theta) = \frac{3\Theta^2 + 3}{2\Theta + 1}$$

established in Proposition 3.6 for the case, where SmarterStart does not wait before starting the final schedule, is tight for all $\Theta \in (\frac{1}{2}(1 + \sqrt{5}), 2]$. Since $\Theta^* \approx 1.71249$ lies in those ranges, the competitive ratio of SmarterStart$_{\Theta^*}$ is indeed exactly ρ^*.

It remains to show that for every $\Theta > 1$ with $\Theta \neq \Theta^*$ the competitive ratio is larger. First, according to Lemma 3.10, the competitive ratio of SmarterStart with parameter $\Theta \in (2, \infty)$ is larger than ρ^*. By monotonicity of f_1, every function value in $(1, \Theta^*)$ is larger than $f_1(\Theta^*) = \rho^*$. Thus, the competitive ratio of SmarterStart with parameter $\Theta \in (1, \Theta^*)$ is larger than ρ^*, since f_1 is tight on $(1, \Theta^*)$ by Proposition 3.8. Similarly, by monotonicity of f_2, every function value in $(\Theta^*, 2]$ is larger than $f_2(\Theta^*) = \rho^*$. Thus, the competitive ratio of SmarterStart with parameter $\Theta \in (\Theta^*, 2]$ is larger than ρ^*, since f_2 is tight on $(\Theta^*, 2]$ by Proposition 3.9. ◀

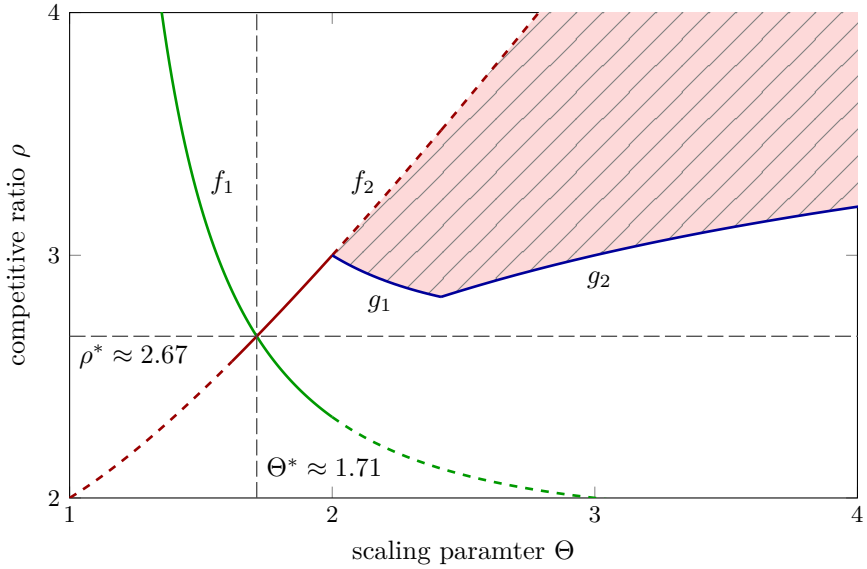


Figure 1 Overview of our bounds for SmarterStart. The functions f_1 (green) / f_2 (red) are upper bounds for the cases where SmarterStart waits / does not wait before starting the final schedule, respectively. The upper bounds are drawn solid in the domains where they are tight for their corresponding case. The functions g_1 and g_2 (blue) are general lower bounds.

References

1 Norbert Ascheuer, Sven Oliver Krumke, and Jörg Rambau. Online Dial-a-Ride Problems: Minimizing the Completion Time. In *Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 639–650, 2000.

2 Mikhail J. Atallah and S. Rao Kosaraju. Efficient Solutions to Some Transportation Problems with Applications to Minimizing Robot Arm Travel. *SIAM Journal on Computing*, 17(5), 1988.

3 G. Ausiello, E. Feuerstein, S. Leonardi, L. Stougie, and M. Talamo. Algorithms for the On-Line Travelling Salesman. *Algorithmica*, 29(4):560–581, 2001.

4 A. Birx and Y. Disser. Tight analysis of the Smartstart algorithm for online Dial-a-Ride on the line. In *Proceedings of the 36th International Symposium on Theoretical Aspects of Computer Science (STACS)*, 2019. Full version: https://arxiv.org/abs/1901.04272.

5 Antje Bjelde, Yann Disser, Jan Hackfeld, Christoph Hansknecht, Maarten Lipmann, Julie Meißner, Kevin Schewior, Miriam Schlöter, and Leen Stougie. Tight Bounds for Online TSP on the Line. In *Proceedings of the 28th Annual Symposium on Discrete Algorithms (SODA)*, pages 994–1005, 2017.

6 Michiel Blom, Sven O. Krumke, Willem E. de Paepe, and Leen Stougie. The Online TSP Against Fair Adversaries. *INFORMS Journal on Computing*, 13(2):138–148, 2001.

7 Moses Charikar and Balaji Raghavachari. The Finite Capacity Dial-A-Ride Problem. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 458–467, 1998.

8 Willem E. de Paepe, Jan Karel Lenstra, Jiri Sgall, René A. Sitters, and Leen Stougie. Computer-Aided Complexity Classification of Dial-a-Ride Problems. *INFORMS Journal on Computing*, 16(2):120–132, 2004.

9 Esteban Feuerstein and Leen Stougie. On-line Single-server Dial-a-ride Problems. *Theoretical Computer Science*, 268(1):91–105, 2001.

10 Paul C. Gilmore and Ralph E. Gomory. Sequencing a One State-Variable Machine: A Solvable Case of the Traveling Salesman Problem. *Operations Research*, 12(5):655–679, 1964.

11 D. J. Guan. Routing a Vehicle of Capacity Greater Than One. *Discrete Applied Mathematics*, 81(1-3):41–57, 1998.

12 Dietrich Hauptmeier, Sven Oliver Krumke, and Jörg Rambau. The Online Dial-a-Ride Problem Under Reasonable Load. In *Proceedings of the 4th Italian Conference on Algorithms and Complexity (CIAC)*, pages 125–136, 2000.

13 Patrick Jaillet and Michael R. Wagner. Generalized Online Routing: New Competitive Ratios, Resource Augmentation, and Asymptotic Analyses. *Operations Research*, 56(3):745–757, 2008.

14 Sven O. Krumke. Online Optimization Competitive Analysis and Beyond, 2001. Habilitation thesis.

15 Sven O. Krumke, Willem E. de Paepe, Diana Poensgen, Maarten Lipmann, Alberto Marchetti-Spaccamela, and Leen Stougie. On Minimizing the Maximum Flow Time in the Online Dial-a-ride Problem. In *Proceedings of the Third International Conference on Approximation and Online Algorithms (WAOA)*, pages 258–269, 2006.

16 Sven O. Krumke, Luigi Laura, Maarten Lipmann, Alberto Marchetti-Spaccamela, Willem de Paepe, Diana Poensgen, and Leen Stougie. Non-abusiveness Helps: An O(1)-Competitive Algorithm for Minimizing the Maximum Flow Time in the Online Traveling Salesman Problem. In *Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pages 200–214, 2002.

17 Maarten Lipmann, Xiwen Lu, Willem E. de Paepe, Rene A. Sitters, and Leen Stougie. On-Line Dial-a-Ride Problems Under a Restricted Information Model. *Algorithmica*, 40(4):319–329, 2004.

18 Fanglei Yi and Lei Tian. On the Online Dial-a-ride Problem with Time-windows. In *Proceedings of the 1st International Conference on Algorithmic Applications in Management (AAIM)*, pages 85–94, 2005.

19 Fanglei Yi, Yinfeng Xu, and Chunlin Xin. Online Dial-a-ride Problem with Time-windows Under a Restricted Information Model. In *Proceedings of the 2nd International Conference on Algorithmic Aspects in Information and Management (AAIM)*, pages 22–31, 2006.

A Proof of Lemma 2.3

In this section we prove Lemma 2.3. The proof is almost identical to the proof of [5, Lemma 6]. Since there are however several parts where inequalities change slightly, we decided to present the full proof here.

▶ **Lemma 2.3.** *If there is a request sequence with two critical requests for* ALG*, we can release additional requests such that* ALG *is not* $(\rho-\varepsilon)$*-competitive on the resulting instance for any* $\varepsilon > 0$*.*

Let the requests σ^L and σ^R be critical. Furthermore, let $p_0 \in \{t^L, t^R\}$ be the starting position of the request $\sigma_0 \in \{\sigma^L, \sigma^R\}$ that is served first by ALG and let $p_1 \in \{t^L, t^R\}$ be the starting position of the request $\sigma_1 \in \{\sigma^L, \sigma^R\}$ that is not served first by ALG. By properties (iii) and (iv) of Definition 2.2, ALG cannot serve σ_0 before time $(2\rho-2)|p_1| + (\rho-2)|p_0|$. Thus, we have

$$\text{ALG}(\sigma_\rho) \geq (2\rho-2)p_1 + (\rho-2)p_0 + |p_0 - p_1| = (2\rho-1)|p_1| + (\rho-1)|p_0|. \tag{32}$$

We have equality in inequality (32) if ALG serves σ_0 the earliest possible time and then moves directly to position p_1. However, in general ALG does not need to do this and instead can wait. At time $t \geq \max\{|p_0|, |p_1|\}$, we have $\text{ALG}(\sigma_\rho) \geq t + |\text{pos}(t) - p_0| + |p_0 - p_1|$ if ALG still has to serve σ_0 and $\text{ALG}(\sigma_\rho) \geq t + |\text{pos}(t) - p_1|$ if σ_0 is served and only σ_1 is left

to be served. We want to measure the delay of ALG at a time $t \geq \max\{|p_0|, |p_1|\}$, i.e. the difference between the time ALG needs at least to serve both requests σ_0 and σ_1 and the time $(2\rho-1)|p_1| + (\rho-1)|p_0|$. We define for $t \geq \max\{|p_0|, |p_1|\}$ the function

$$\text{delay}(t) := \begin{cases} t + |\text{pos}\,(t) - p_0| - (\rho-2)|p_0| - (2\rho-2)|p_1| & \text{if } \sigma_0 \text{ is not served at } t, \\ t + |\text{pos}\,(t) - p_1| - (\rho-1)|p_0| - (2\rho-1)|p_1| & \text{if } \sigma_0 \text{ is served at } t, \text{ but } \sigma_1 \text{ not}, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

We make the following observation about delay.

▶ **Observation A.1.** *Let $t \geq \max\{|p_0|, |p_1|\}$ be a time at which σ_1 is not served yet. The earliest time* ALG *can serve σ_1 is $(2\rho-1)|p_1| + (\rho-1)|p_0| + \text{delay}(t)$.*

▶ **Lemma A.2.** *There is a $W \geq 0$ with*

$$\text{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho-1}\right) = W$$

Proof. Because of property (ii) of Definition 2.2, at time $\max\{|p_0|, |p_1|\}$ neither σ_0 nor σ_1 has been served by ALG yet. Since ALG serves σ_1 after σ_0, the request σ_1 is not served before time $\max\{|p_0|, |p_1|\} + |p_0| + |p_1| \geq 2|p_1| + |p_0|$, i.e, $\text{delay}(2p_1 + p_0)$ is defined. Because of properties (iii) and (iv) of Definition 2.2, σ_0 is not served before time $(2\rho-2)|p_1| + (\rho-2)|p_0|$. Thus, for $t \geq (2\rho-2)p_1 + (\rho-2)p_0$, we have $\text{delay}(t) \geq 0$. We have

$$\begin{aligned} 2p_1 + p_0 &\overset{\text{Def 2.2 (v)}}{\geq} 2p_1 + (3-\rho)\frac{-8\rho^2 + 50\rho - 66}{4\rho^2 - 30\rho + 50}|p_1| + (\rho-2)|p_0| \\ &\overset{2 < \rho < 2.5}{>} (2\rho-2)|p_1| + (\rho-2)|p_0|, \end{aligned} \tag{33}$$

i.e. $\text{delay}(2p_1 + p_0) \geq 0$. If $\text{delay}(2p_1 + p_0) = 0$, we have $W = 0$ and are done. Otherwise, by inequality (33), we have $\text{delay}(2p_1 + p_0) > 0$. Note that ALG needs to serve σ_1 at some point to be $(\rho-\varepsilon)$-competitive. Let W^* be chosen such that ALG serves σ_1 at time $2|p_1| + |p_0| + \frac{W^*}{\rho-1}$. Therefore $\text{delay}(2|p_1| + |p_0| + \frac{W^*}{\rho-1} - \varepsilon')$ is defined for some sufficiently small $\varepsilon' \leq |p_1|$. We define the function

$$f(W) := \text{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho-1}\right) - W.$$

Note that f is continuous and we have $f(0) > 0$. If

$$\text{delay}\left(2|p_1| + |p_0| + \frac{W^*}{\rho-1} - \varepsilon'\right) \leq \frac{W^*}{\rho-1} - \varepsilon' \overset{\rho > 1}{<} W^* - (\rho-1)\varepsilon',$$

we have $f(W^* - (\rho-1)\varepsilon') < 0$ and we find W in the interval $(0, W^* - (\rho-1)\varepsilon']$. Otherwise, we have

$$\text{delay}\left(2|p_1| + |p_0| + \frac{W^*}{\rho-1} - \varepsilon'\right) > \frac{W^*}{\rho-1} - \varepsilon'.$$

By Observation A.1 ALG has not served σ_1 at time

$$(2\rho-1)|p_1| + (\rho-1)|p_0| + \frac{W^*}{\rho-1} - \varepsilon' \overset{\rho > 2, \varepsilon' \leq |p_1|}{>} 2|p_1| + |p_0| + \frac{W^*}{\rho-1}.$$

This is a contradiction to the fact, that W^* was chosen such that ALG serves σ_1 at time $2|p_1| + |p_0| + \frac{W^*}{\rho-1}$. ◀

▶ **Lemma A.3.** *Let $W \geq 0$ with*

$$\mathit{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) = W.$$

ALG *serves σ_0 no later than time $2|p_1| + |p_0| + \frac{W}{\rho-1}$.*

Proof. Assume we have

$$2|p_1| + |p_0| + \frac{W}{\rho - 1} \geq (2\rho - 2)|p_1| + (\rho - 2)|p_0| + W. \tag{34}$$

Then, by definition of W and Observation A.1, ALG can serve σ_1 at time

$$(2\rho-1)|p_1|+(\rho-1)|p_0|+\text{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) = (2\rho-1)|p_1|+(\rho-1)|p_0|+W. \tag{35}$$

Because of inequality (34), this can only be the case if ALG serves σ_0 no later than time

$$(2\rho-1)|p_1|+(\rho-1)|p_0|+W-|p_1|-|p_0| = (2\rho-2)|p_1|+(\rho-2)|p_0|+W \overset{(34)}{\leq} 2|p_1|+|p_0|+\frac{W}{\rho-1}.$$

Thus, it remains to show inequality (34). Because of property (i) of Definition 2.2 all requests can be served the tours $\text{move}(p_0) \oplus \text{move}(p_1)$ and $\text{move}(p_1) \oplus \text{move}(p_0)$. By inequality 35, we have $\text{ALG}(\sigma_\rho) \geq (2\rho - 1)|p_1| + (\rho - 1)|p_0| + W$. Thus, if we have

$$\text{ALG}(\sigma_\rho) \geq (2\rho - 1)|p_1| + (\rho - 1)|p_0| + W > (\rho - \varepsilon)(2|p_1| + |p_0|) \geq (\rho - \varepsilon)\text{OPT}(\sigma_\rho),$$

ALG is not $(\rho - \varepsilon)$-competitive. Therefore, we may assume

$$(2\rho - 1)|p_1| + (\rho - 1)|p_0| + W \leq (\rho - \varepsilon)(2|p_1| + |p_0|),$$

and thus

$$\begin{aligned} W &\leq (\rho - \varepsilon)(2|p_1| + |p_0|) - (2\rho - 1)|p_1| - (\rho - 1)|p_0| \\ &= (1 - 2\varepsilon)|p_1| + (1 - \varepsilon)|p_0| \\ &< |p_1| + |p_0|. \end{aligned} \tag{36}$$

Inequality (34) now is equivalent to the inequality

$$\begin{aligned} \frac{2|p_1| + |p_0| - ((2\rho - 2)|p_1| + (\rho - 2)|p_0|)}{1 - \frac{1}{\rho-1}} &= \frac{(\rho - 1)(4 - 2\rho)}{\rho - 2}|p_1| + \frac{(\rho - 1)(3 - \rho)}{\rho - 2}|p_0| \\ &\overset{\text{Def 2.2 (v)}}{\geq} |p_0| + (2 - 2\rho)|p_1| \\ &\quad + \frac{(-\rho^2 + 3\rho - 1)(-8\rho^2 + 50\rho - 66)}{(\rho - 2)(4\rho^2 - 30\rho + 50)}|p_1| \\ &\geq |p_0| + \frac{5\rho^3 - 36\rho^2 + 86\rho - 67}{2\rho^3 - 19\rho^2 + 55\rho - 50}|p_1| \\ &\overset{2 < \rho < 2.5}{>} |p_0| + |p_1| \\ &\overset{(36)}{>} W \end{aligned}$$

if we solve inequality (34) for W. ◀

Now we have all ingredients to proof Lemma 2.3.

Proof of Lemma 2.3. Let $W \geq 0$ with $\text{delay}(2|p_1| + |p_0| + \frac{W}{\rho-1}) = W$. We present the request

$$\sigma_0^+ = (p_0^+, p_0^+; t_0^+) := \left(p_0 + \text{sgn}(p_0)\frac{W}{\rho-1}, p_0 + \text{sgn}(p_0)\frac{W}{\rho-1}; 2|p_1| + |p_0| + \frac{W}{\rho-1}\right)$$

and distinguish two cases.

Case 1: At time t_0^+, ALG is at least as close to p_1 as to p_0^+ or it serves σ_1 before σ_0^+. In this case, we do not present additional requests. By Lemma A.3, ALG has served σ_0 at time t_0^+ or before and by Observation A.1 it does not serve σ_1 earlier than time $(2\rho-1)|p_1| + (\rho-1)|p_0| + W$. Thus, we have

$$\begin{aligned}\text{ALG}(\sigma_\rho) &\geq (2\rho-1)|p_1| + (\rho-1)|p_0| + W + |p_1| + |p_0| + \frac{W}{\rho-1} \\ &\geq \rho\left(2|p_1| + |p_0| + \frac{W}{\rho-1}\right) \\ &= \rho\text{OPT}(\sigma_\rho).\end{aligned}$$

Case 2: At time t_0^+, ALG is closer to p_0^+ than to p_1 and it serves σ_0^+ first. We assume that the offline server continues moving away from the origin after serving σ_0^+ at time p_0^+. Then, the position of the offline serve at time $t \geq |p_1|$ is $\text{sgn}(p_0)t + 2p_1$. We denote by

$$M(t) := \frac{\text{sgn}(p_0)t + 3p_1}{2}$$

the midpoint between the current position of the offline server and the position p_1. Note that the time $M^{-1}(p)$, when the midpoint is at position p is given by

$$M^{-1}(p) := |2p - 3p_1|.$$

We again distinguish two cases

Case 2.1: ALG does not serve σ_0^+ until time $M^{-1}(p_0^+)$. In this case, we do not present additional requests. Since we are in Case 2, neither σ_0^+ nor σ_1 is served at time $M^{-1}(p_0^+)$. Thus, we have

$$\begin{aligned}\text{ALG}(\sigma_\rho) &\geq M^{-1}(p_0^+) + |p_0^+| + |p_1| \\ &= |2p_0^+ - 3p_1| + |p_0^+| + |p_1| \\ &= |2p_0 + 2\text{sgn}(p_0)\frac{W}{\rho-1} - 3p_1| + |p_0| + \frac{W}{\rho-1} + |p_1| \\ &= 3|p_0| + 4|p_1| + 3\frac{W}{\rho-1} \\ &\overset{2<\rho<2.5}{>} \rho|p_0| + 2\rho|p_1| + 3\frac{W}{\rho-1} \\ &> \rho\left(|p_0| + 2|p_1| + \frac{W}{\rho-1}\right) \\ &= \rho\text{OPT}(\sigma_\rho).\end{aligned}$$

Case 2.2: ALG serves σ_0^+ before time $M^{-1}(p_0^+)$. By definition of W, the delay function is defined for time p_0^+, hence ALG has not served σ_1 before time p_0^+. Since ALG is to the right of the midpoint $M(p_0^+)$ at time p_0^+, there is a first time t_{mid} at which $M(t_{\text{mid}}) = \text{pos}\,(t_{\text{mid}})$. We present the request

$$\sigma_0^{++} = (p_0^{++}, p_0^{++}; t_0^{++}) := (\text{sgn}(p_0)t_{\text{mid}} + 2p_1, \text{sgn}(p_0)t_{\text{mid}} + 2p_1; t_{\text{mid}}).$$

Note that ALG is at the midpoint between p_0^{++} and p_1 and thus, both tours $\text{move}(p_0^{++}) \oplus \text{move}(p_1)$ and $\text{move}(p_1) \oplus \text{move}(p_0^{++})$ incur identical costs for ALG. We have

$$\text{ALG}(\sigma_\rho) \geq t_{\text{mid}} + 3\left(\frac{|\text{sgn}(p_0)t_{\text{mid}} + 2p_1 - p_1|}{2}\right) = \frac{5t_{\text{mid}} + 3|p_1|}{2}$$

We have $\text{OPT}(\sigma_\rho) = t_{\text{mid}}$, i.e., if we want to show

$$\text{ALG}(\sigma_\rho) \geq \frac{5t_{\text{mid}} + 3|p_1|}{2} \geq \rho t_{\text{mid}} = \rho\text{OPT}(\sigma_\rho) \tag{37}$$

Inequality (37) is equivalent to

$$(5 - 2\rho)t_{\text{mid}} \geq 3|p_1|. \tag{38}$$

Since $2\rho < 2.5$, the coefficient $(5 - 2\rho)$ of t_{mid} is positive. Thus we may assume t_{mid} is minimal to show the inequality (38). By assumption, σ_0^+ is already served at time t_{mid}. Hence, t_{mid} is minimum if, starting at time t_0^+ at position $\text{pos}\,(t_0^+)$, ALG serves σ_0^+ and then moves towards the origin. Then, t_{mid} is the solution of the equation

$$\text{sgn}(p_0)t_0^+ + |\text{pos}\,(t_0^+) - p_0^+| + p_0^+ - \text{sgn}(p_0)t_{\text{mid}} = \frac{\text{sgn}(p_0)t_{\text{mid}} + 3p_1}{2}. \tag{39}$$

Because of Lemma A.3, the request σ_0 is already served at time t_0^+. Furthermore, since the position of σ_1 has not been visited yet at time t_0^+, we have $\text{sgn}(p_0)\text{pos}\,(t_0^+) > \text{sgn}(p_0)p_1$, i.e.,

$$|\text{pos}\,(t_0^+) - p_1| = \text{sgn}(p_0)(\text{pos}\,(t_0^+) - p_1) > 0$$

and thus, because of $-\text{sgn}(p_0)p_1 = |p_1|$, we get

$$\begin{aligned}
\text{delay}(t_0^+) &= t_0^+ + |\text{pos}\,(t_0^+) - p_1| - (\rho - 1)|p_0| - (2\rho - 1)|p_1| \\
&= t_0^+ + \text{sgn}(p_0)\text{pos}\,(t_0^+) - \text{sgn}(p_0)p_1 - (\rho - 1)|p_0| - (2\rho - 1)|p_1| \\
&= t_0^+ + \text{sgn}(p_0)\text{pos}\,(t_0^+) + |p_1| - (\rho - 1)|p_0| - (2\rho - 1)|p_1|.
\end{aligned} \tag{40}$$

Solving equation (40) for $\text{sgn}(p_0)\text{pos}\,(t_0^+)$ gives

$$\begin{aligned}
\text{sgn}(p_0)\text{pos}\,(t_0^+) &= \text{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) - \frac{W}{\rho - 1} \\
&\quad +(\rho - 2)|p_0| + (2\rho - 4)|p_1| \\
&= W - \frac{W}{\rho - 1} + (\rho - 2)|p_0| + (2\rho - 4)|p_1| \\
&= \frac{\rho - 2}{\rho - 1}W + (\rho - 2)|p_0| + (2\rho - 4)|p_1| \qquad (41) \\
&\overset{\rho < 3}{<} \frac{W}{\rho - 1} + (\rho - 2)|p_0| + (2\rho - 4)|p_1| \\
&\overset{\text{Def 2.2 (v)}}{\leq} \frac{W}{\rho - 1} + \left((\rho - 2) + (2\rho - 4)\frac{4\rho^2 - 30\rho + 50}{-8\rho^2 + 50\rho - 66}\right)|p_0| \\
&\overset{1.9 < \rho < 4.3}{<} \frac{W}{\rho - 1} + |p_0| \\
&\overset{\text{sgn}(p_0) = \text{sgn}(p_0^+)}{=} \text{sgn}(p_0)p_0^+.
\end{aligned}$$

Thus, we have

$$|\text{pos}\left(t_0^+\right) - p_0^+| = \text{sgn}(p_0)(p_0^+ - \text{pos}\left(t_0^+\right)) > 0 \tag{42}$$

Using inequality (42) and plugging inequality (41) into inequality (39) gives us

$$\begin{aligned}
\text{sgn}(p_0)t_{\text{mid}} &= \frac{1}{3}(2\text{sgn}(p_0)t_0^+ + 2|\text{pos}\left(t_0^+\right) - 2p_0^+| + 2p_0^+ - 3p_1) \\
&\overset{(42)}{=} \frac{1}{3}(2\text{sgn}(p_0)t_0^+ + 2\text{sgn}(p_0)p_0^+ - 2\text{sgn}(p_0)\text{pos}\left(t_0^+\right) + 2p_0^+ - 3p_1) \\
&= \frac{1}{3}\left(-7p_1 + 6p_0 + \frac{(6\text{sgn}(p_0))W}{\rho - 1} - 2\text{sgn}(p_0)\text{pos}\left(t_0^+\right)\right) \\
&\overset{(41)}{=} \frac{1}{3}\left(-(15 - 4\rho)p_1 + (10 - 2\rho)p_0 + \frac{(10 - 2\rho)\text{sgn}(p_0)W}{\rho - 1}\right)
\end{aligned} \tag{43}$$

Note that we also used $\text{sgn}(p_0) = \text{sgn}(p_0^+) = -\text{sgn}(p_1)$. Multiplying equality (43) with $\text{sgn}(p_0)$ gives us

$$t_{\text{mid}} = \frac{1}{3}\left((15 - 4\rho)|p_1| + (10 - 2\rho)|p_0| + \frac{(10 - 2\rho)W}{\rho - 1}\right). \tag{44}$$

By substituting (44) into (38) and noting that it is hardest to satisfy, when $W = 0$, we get

$$\frac{|p_0|}{|p_1|} \leq \frac{4\rho^2 - 30\rho + 50}{-8\rho^2 + 50\rho - 66},$$

which is true due to Definition 2.2 (v). ◀

Improved Online Algorithms for Knapsack and GAP in the Random Order Model

Susanne Albers
Technical University of Munich, Germany
albers@in.tum.de

Arindam Khan
Indian Institute of Science, Bangalore, India[1]
arindamkhan@iisc.ac.in

Leon Ladewig
Technical University of Munich, Germany
ladewig@in.tum.de

Abstract

The *knapsack problem* is one of the classical problems in combinatorial optimization: Given a set of items, each specified by its size and profit, the goal is to find a maximum profit packing into a knapsack of bounded capacity. In the online setting, items are revealed one by one and the decision, if the current item is packed or discarded forever, must be done immediately and irrevocably upon arrival. We study the online variant in the random order model where the input sequence is a uniform random permutation of the item set.

We develop a randomized (1/6.65)-competitive algorithm for this problem, outperforming the current best algorithm of competitive ratio 1/8.06 [Kesselheim et al. SIAM J. Comp. 47(5)]. Our algorithm is based on two new insights: We introduce a novel algorithmic approach that employs two given algorithms, optimized for restricted item classes, sequentially on the input sequence. In addition, we study and exploit the relationship of the knapsack problem to the 2-secretary problem.

The *generalized assignment problem* (GAP) includes, besides the knapsack problem, several important problems related to scheduling and matching. We show that in the same online setting, applying the proposed sequential approach yields a (1/6.99)-competitive randomized algorithm for GAP. Again, our proposed algorithm outperforms the current best result of competitive ratio 1/8.06 [Kesselheim et al. SIAM J. Comp. 47(5)].

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Online algorithms, knapsack problem, random order model

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.22

Category APPROX

Funding Work supported by the European Research Council, Grant Agreement No. 691672.

1 Introduction

Many real-world problems can be considered resource allocation problems. For example, consider the loading of cargo planes with (potential) goods of different weights. Each item raises a certain profit for the airline if it is transported; however, not all goods can be loaded due to airplane weight restrictions. Clearly, the dispatcher seeks for a maximum profit packing fulfilling the capacity constraint. This example from [24] illustrates the *knapsack problem*: Given a set of n items, specified by a size and a profit value, and a resource (called knapsack) of fixed capacity, the goal is to find a subset of items (called packing) with maximum total

[1] A part of this work was done when the author was at Technical University of Munich.


© Susanne Albers, Arindam Khan, and Leon Ladewig;
licensed under Creative Commons License CC-BY



profit and whose total size does not exceed the capacity. Besides being a fundamental and extensively studied problem in combinatorial optimization, knapsack problems arise in many and various practical settings. We refer the readers to textbooks [24, 35] and to the surveys of previous work in [14, 19] for further references.

In the *generalized assignment problem* (GAP) [35], resources of different capacities are given, and the size and the profit of an item depend on the resource to which it is assigned. The GAP includes many prominent problems, such as the (multiple) knapsack problem [13], weighted bipartite matching [28], AdWords [36], and the display ads problem [17]. Further applications of GAP are outlined in the survey articles [11, 41].

We study online variants of the knapsack and GAP problems. Here, n items are presented sequentially, and the decision for each item must be made immediately upon arrival. In fact, many real-world optimization problems occur as online problems, as often decisions must be made under uncertain conditions. For example, consider the introducing logistics example, if the airline needs to answer customer requests immediately without knowing future requests. The online knapsack problem has been studied in particular in the context of online auctions [9, 45].

Typically, the performance measure for online algorithms is the *competitive ratio*, which is defined as the ratio between the values of the algorithmic solution and an optimal offline solution for a worst-case input. It can be shown that, even for the knapsack problem, the general online setting admits no algorithms with bounded competitive ratio [34, 45]. However, most hardness results are based on a worst-case input presented in adversarial order. In the *random order model*, the performance of an algorithm is evaluated for a worst-case input, but the adversary has no control over the input order; the input sequence is drawn uniformly at random among all permutations. This model is known from the secretary problem [15, 31] and its generalizations [7, 12, 18]; it has been successfully applied to other online problems, for example, scheduling and packing [1, 16, 20, 25, 27, 39], graph problems [8, 26, 33], facility location [37], budgeted allocation [38], and submodular welfare maximization [30].

1.1 Related Work

Online knapsack problem. The problem was first studied by Marchetti-Spaccamela and Vercellis [34], who showed that no deterministic online algorithm for this problem can obtain a constant competitive ratio. Moreover, Chakrabarty et al. [45] demonstrated that this fact cannot be overcome by randomization.

Given such hardness results, several relaxations have been introduced and investigated. Most relevant to our work are results in the random order model. Introduced as the *secretary knapsack problem* [6], Babaioff et al. developed a randomized algorithm of competitive ratio $1/(10e) < 1/27$. Kesselheim et al. [27] achieved a significant improvement by developing a $(1/8.06)$-competitive randomized algorithm for the generalized assignment problem. Finally, Vaze [43] showed that there exists a deterministic algorithm of competitive ratio $1/(2e) < 1/5.44$, assuming that the maximum profit of a single item is small compared to the profit of the optimal solution.

Apart from the random order model, different further relaxations have been considered. Marchetti-Spaccamela and Vercellis [34] studied a stochastic model wherein item sizes and profits are drawn from a fixed distribution. Lueker [32] obtained improved bounds in this model. Chakrabarty et al. [45] studied the problem when the density (profit-size ratio) of each item is in a fixed range $[L, U]$. Under the further assumption that item sizes are small compared to the knapsack capacity, Chakrabarty et al. proposed an algorithm of competitive ratio $\ln(U/L) + 1$ and provided a lower bound of $\ln(U/L)$. Another branch of

research considers removable models, where the algorithm can remove previously packed items. Removing such items can incur no cost [22,23] or a cancellation cost (*buyback model*, [4,5,21]). Recently, Vaze [44] considered the problem under a (weaker) expected capacity constraint. This variant admits a competitive ratio of $1/4e$.

Online GAP. Since all hardness results for online knapsack also hold for online GAP, research focuses on stochastic variants or modified online settings. Currently, the only result for the random order model is the previously mentioned (1/8.06)-competitive randomized algorithm proposed by Kesselheim et al. [27]. To the best of our knowledge, the earliest paper considering online GAP is due to Feldman et al. [17]. They obtained an algorithm of competitive ratio tending to $1 - 1/e$ in the *free disposal model*. In this model, the total size of items assigned to a resource might exceed its capacity; in addition, no item consumes more than a small fraction of any resource. A stochastic variant of online GAP was studied by Alaei et al. [2]. Here, the size of an item is drawn from an individual distribution that is revealed upon arrival of the item, together with its profit. However, the algorithm learns the actual item size only after the assignment. If no item consumes more than a $(1/k)$-fraction of any resource, the algorithm proposed by Alaei et al. has competitive ratio $1 - 1/\sqrt{k}$.

Online packing LPs. In contrast to GAP, general packing LPs describe problems where requests can consume more than one resource. The study of online packing LPs was initiated by Buchbinder and Naor [10] in the adversarial model. In several papers [1, 16, 27, 39] it has been shown that the random order model admits $(1 - \varepsilon)$-competitive algorithms assuming large capacity ratios, i.e., when the capacity of any resource is large compared to the maximum demand for it. Most recently, Kesselheim et al. [27] showed that there is a $(1 - \varepsilon)$-competitive algorithm if $B = \Omega((\log d)/\varepsilon^2)$, where B is the capacity ratio and d is the column sparsity (the maximum number of resources occurring in a single column).

1.2 Our Contributions

As outlined above, for online knapsack and GAP in the adversarial input model, nearly all previous works attain constant competitive ratios at the cost of either (a) imposing structural constraints on the input or (b) significantly relaxing the original online model. Therefore, we study both problems in the random order model, which is less pessimistic than the adversarial model but still considers worst-case instances without further constraints on the item properties. For the knapsack problem, our main result is the following.

▶ **Theorem 1.1.** *There exists a* (1/6.65)*-competitive randomized algorithm for the online knapsack problem in the random order model assuming* $n \to \infty$.

One challenge in the design of knapsack algorithms is that the optimal packing can have, on a high level, at least two different structures. Either there are few large items, constituting the majority of the packing's profit, or there are many small such items. Previous work [6,27] is based on splitting the input according to item sizes and then employing algorithms tailored for these restricted instances. However, the algorithms from [6,27] choose a single item type via an initial random choice, and then pack items of that type exclusively. In contrast, our approach considers different item types in distinct time intervals, rather than discarding items of a specific type in advance. More precisely, we develop algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ which are combined in a novel *sequential approach*: While large items appearing in early rounds are packed using $\mathcal{A}_L$, algorithm $\mathcal{A}_S$ is applied to pack small items revealed in later rounds. We think that this approach may be helpful for other problems in similar online settings as well.

The proposed algorithm $\mathcal{A}_L$ deals with the knapsack problem where all items consume more than $1/3$ of the capacity (we call this problem 2-KS). The 2-KS problem is closely related to the k-secretary problem [29] for $k = 2$. We also develop a general framework that allows to employ any algorithm for the 2-secretary problem to obtain an algorithm for 2-KS. As a side product, we obtain a simple $(1/3.08)$-competitive deterministic algorithm for 2-KS in the random order model. For items whose size is at most $1/3$ of the resource capacity, we give a simple and efficient algorithm $\mathcal{A}_S$. Here, a challenging constraint is that $\mathcal{A}_L$ and $\mathcal{A}_S$ share the same resource, so we need to argue carefully that the decisions of $\mathcal{A}_S$ are feasible, given the packing of $\mathcal{A}_L$ from previous rounds.

Finally, we show that the proposed sequential approach also improves the current best result for GAP [27] from competitive ratio $1/8.06$ to $1/6.99$.

▶ **Theorem 1.2.** *There exists a* $(1/6.99)$*-competitive randomized algorithm for the online generalized assignment problem in the random order model assuming* $n \to \infty$*.*

For this problem we use the algorithmic building blocks $\mathcal{A}_L$, $\mathcal{A}_S$ developed in [26,27]. However, we need to verify that $\mathcal{A}_L$, an algorithm for edge-weighted bipartite matching [26], satisfies the desired properties for the sequential approach. We point out that the assignments of our algorithm differ structurally from the assignments of the algorithm proposed in [27]. In the assignments of the latter algorithm, all items are either large or small compared to the capacity of the assigned resource. In our approach, both situations can occur, because resources are managed independently.

Roadmap. We focus on the result on the knapsack problem (Theorem 1.1) in the first chapters of this paper. For this purpose, we provide elementary definitions in Section 2. Our main technical contribution is formally introduced in Section 3: Here, we describe an algorithmic framework performing two algorithms $\mathcal{A}_L$, $\mathcal{A}_S$ sequentially. In Sections 4 and 5, we design and analyze the algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ for the knapsack problem. Finally, in Section 6 we describe how the sequential approach can be applied to GAP. Due to space constraints, some proofs are deferred to Appendix A (knapsack) and to Appendix B (GAP).

2 Preliminaries

Let $[n] := \{1, \dots, n\}$. Further, let $\mathbb{Q}_{\geq 0}$ and $\mathbb{Q}_{>0}$ denote the set of non-negative and positive rational numbers, respectively.

Knapsack problem. We are given a set of items $I = [n]$, each item $i \in I$ has *size* $s_i \in \mathbb{Q}_{>0}$ and a *profit* (*value*) $v_i \in \mathbb{Q}_{\geq 0}$. The goal is to find a maximum profit packing into a knapsack of size $W \in \mathbb{Q}_{>0}$, i.e., a subset $M \subseteq I$ such that $\sum_{i \in M} s_i \leq W$ and $\sum_{i \in M} v_i$ is maximized. W.l.o.g. we can assume $s_i \leq W$ for all $i \in I$. In the online variant of the problem, in each *round* $\ell \in [n]$ a single item i is revealed together with its size and profit. The online algorithm must decide immediately and irrevocably whether to pack i. We call an item *visible in round* ℓ if it arrived in round ℓ or earlier.

Random order performance. We analyze the performance of algorithms in the *random order model*. Given a worst case input $\mathcal{I}$, the order in which $\mathcal{I}$ is presented is drawn uniformly at random from the set of all permutations. For an algorithm $\mathcal{A}$, its *competitive ratio* is defined as $\mathbf{E}[\mathcal{A}(\mathcal{I})] / \mathrm{OPT}(\mathcal{I})$, where $\mathcal{A}(\mathcal{I})$ and $\mathrm{OPT}(\mathcal{I})$ denote the profits of the solutions of $\mathcal{A}$ and an optimal offline algorithm, respectively. Here, the expectation is taken over

Algorithm 1 Sequential approach.

Input : Random permutation π of n items in I, a knapsack of capacity W, parameters $c, d \in (0,1)$ with $c < d$, algorithms $\mathcal{A}_L$, $\mathcal{A}_S$.
Output : A feasible (integral) knapsack packing.
Let ℓ be the current round.
if $\ell \leq cn$ **then**
  Sampling phase – discard all items;
if $cn + 1 \leq \ell \leq dn$ **then**
  Pack $\pi(\ell)$ iff $\mathcal{A}_L$ packs $\pi_L(\ell)$;
if $dn + 1 \leq \ell \leq n$ **then**
  Pack $\pi(\ell)$ iff $\mathcal{A}_S$ packs $\pi_S(\ell)$ and the remaining capacity is sufficiently large.

all permutations and random choices of the algorithm. As above, we slightly overload the notation and also use $\mathcal{A}$ as a random variable for the profit of the solution returned by an algorithm $\mathcal{A}$.

We classify items as large or small, depending on their size compared to W and a parameter $\delta \in (0,1)$ to be determined later.

▶ **Definition 2.1.** *We say an item i is δ-large if $s_i > \delta W$ and δ-small if $s_i \leq \delta W$. Whenever δ is clear from the context, we say an item is large or small for short. Based on the given item set I, we define two modified item sets I_L and I_S, which are obtained as follows:*

- *I_L: Replace each small item by a large item of profit 0*
- *I_S: Replace each large item by a small item of profit 0.*

Therefore, I_L only contains large items and I_S only contains small items. We can assume that no algorithm packs a zero-profit item, thus any algorithmic packing of I_L or I_S can be turned into a packing of I having the same profit. Let OPT, OPT_L, and OPT_S be the total profits of optimal packings for I, I_L, and I_S, respectively. A useful upper bound for OPT is

$$\mathrm{OPT} \leq \mathrm{OPT}_L + \mathrm{OPT}_S\,. \tag{1}$$

3 Sequential Approach

A common approach in the design of algorithms for secretary problems is to set two phases: a *sampling phase*, where all items are rejected, followed by a *decision phase*, where some items are accepted according to a decision rule. Typically, this rule is based on the information gathered in the sampling phase. We take this concept a step further: The key idea of our sequential approach is to use a part of the sampling phase of one algorithm as decision phase of another algorithm, which itself can have a sampling phase. This way, two algorithms are performed in a sequential way, which makes better use of the entire instance. We combine this idea with using different strategies for small and large items.

Formally, let $\mathcal{A}_L$ and $\mathcal{A}_S$ be two online knapsack algorithms and I_L and I_S be the item sets constructed according to Definition 2.1. Further, let $0 < c < d < 1$ be two parameters to be specified later. Our proposed algorithm samples the first cn rounds; during this time no item is packed. From round $cn + 1$ to dn, the algorithm considers large items exclusively. In this interval we follow the decisions of $\mathcal{A}_L$. After round dn, the algorithm processes only small items and follows the decisions of $\mathcal{A}_S$. However, it might be the case that an item accepted by $\mathcal{A}_S$ cannot be packed because the knapsack capacity is exhausted due to the packing of $\mathcal{A}_L$ in earlier rounds. Note that all rounds $1, \ldots, dn$ can be considered as the

Algorithm 2 Algorithm $\mathcal{A}_L$ for large items.

Input : Random permutation of n $(1/3)$-large items, a knapsack of capacity W, parameters $c, d \in (0,1)$ with $c < d$.
Output : A feasible (integral) packing of the knapsack.
Let ℓ be the current round.
if $\ell \leq cn$ **then**
  Sampling phase – discard all items.
Let v^* be the maximum profit seen up to round cn.
if $cn + 1 \leq \ell \leq dn$ **then**
  Pack the first two items of profit higher than v^*, if feasible.
if $\ell > dn$ **then**
  Discard all items.

sampling phase for $\mathcal{A}_S$. A formal description is given in Algorithm 1. Here, for a given input sequence π of I, let π_L and π_S denote the corresponding sequences from I_L and I_S, respectively. Note that π is revealed sequentially and π_L, π_S can be constructed online. For any input sequence π, let $\pi(\ell)$ denote the item at position $\ell \in [n]$.

In the final algorithm we set the threshold for small items to $\delta = 1/3$ and use Algorithm 1 with parameters $c = 0.42291$ and $d = 0.64570$. Under the assumption $n \to \infty$ we can assume $cn, dn \in \mathbb{N}$. We next give a high-level description of the proof of Theorem 1.1.

Proof of Theorem 1.1. Let $\mathcal{A}$ be Algorithm 1 and $\mathcal{A}_L$, $\mathcal{A}_S$ be the algorithms developed in Sections 4 and 5. In the next sections we prove the following results (see Lemmas 4.7 and 5.5): The expected profit from $\mathcal{A}_L$ in rounds $cn+1, \dots, dn$ is at least $\frac{1}{6.65}\,\mathrm{OPT}_L$, and the expected profit from $\mathcal{A}_S$ in rounds $dn+1, \dots, n$ is at least $\frac{1}{6.65}\,\mathrm{OPT}_S$. Together with inequality (1), we obtain

$$\mathbf{E}[\mathcal{A}] \geq \mathbf{E}[\mathcal{A}_L] + \mathbf{E}[\mathcal{A}_S] \geq \frac{1}{6.65}\,\mathrm{OPT}_L + \frac{1}{6.65}\,\mathrm{OPT}_S \geq \frac{1}{6.65}\,\mathrm{OPT}\,. \qquad \blacktriangleleft$$

The order in which $\mathcal{A}_L$ and $\mathcal{A}_S$ are arranged in Algorithm 1 follows from two observations. Algorithm $\mathcal{A}_S$ is powerful if it samples roughly $(2/3)n$ rounds; a part of this long sampling phase can be used as the decision phase of $\mathcal{A}_L$, for which a shorter sampling phase is sufficient. Moreover, the first algorithm should either pack high-profit items, or should leave the knapsack empty for the following algorithm with high probability. The algorithm $\mathcal{A}_L$ we propose in Section 4 has this property (see Lemma 4.8). In contrast, if $\mathcal{A}_S$ would precede $\mathcal{A}_L$, the knapsack would be empty at the beginning of $\mathcal{A}_L$ with very small probability, in which case we would not benefit from $\mathcal{A}_L$.

Finally, note that better algorithms and parameterizations for the respective sub-problems exist (see Lemma 4.6 and [27]). However, for the overall performance we need algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ that perform well evaluated in the sequential framework.

4 Large Items

The approach presented in this section is based on the connection between the online knapsack problem under random arrival order and the k-secretary problem [29]. In the latter problem, the algorithm can accept up to k items and the goal is to maximize the sum of their profits. The k-secretary problem generalizes the classical secretary problem [15, 31] and is itself a special case of the online knapsack problem under random arrival order (if all knapsack items have size W/k).

Table 1 Definition of packing types A-M. We use set notation $\{i, j\}$ if i and j can be packed in any order, and tuple notation (i, j) if the packing order must be as given.

type	content	constraint on j	probability p_X
A	$\{1, 2\}$	-	$p_{12} + p_{21}$
B	$\{1, 3\}$	-	$p_{13} + p_{31}$
C	$\{2, 3\}$	-	$p_{23} + p_{32}$
D	$(1, j)$	-	p_1
E	$(2, j)$	-	p_2
F	$(3, j)$	-	p_3
G	$(4, j)$	-	p_4
H	$(1, j)$	$j \neq 2$	$p_1 - p_{12}$
I	$(1, j)$	$j \neq 3$	$p_1 - p_{13}$
J	$(2, j)$	$j \neq 1$	$p_2 - p_{21}$
K	$(2, j)$	$j \neq 3$	$p_2 - p_{23}$
L	$(3, j)$	$j \neq 1$	$p_3 - p_{31}$
M	$(3, j)$	$j \neq 2$	$p_3 - p_{32}$

In our setting, each large item consumes more than $\delta = 1/3$ of the knapsack capacity. We call this problem 2-KS, since at most two items can be packed completely. Therefore, any 2-secretary algorithm can be employed to identify high-profit items and pack them if feasible. Although this idea applies to any δ and corresponding k, the approach seems stronger for small k: Intuitively, the characteristics of k-KS and k-secretary deviate with growing k, while 1-KS is exactly 1-secretary. Furthermore, the k-secretary problem is for $k = 2$ rather well studied [3, 12], while the exact optimal competitive ratios for $k \geq 3$ are still unknown.

In the following, let $\mathcal{A}_L$ be Algorithm 2. This is an adaptation of the algorithm SINGLE-REF developed for the k-secretary problem in [3]. As discussed above, 2-secretary and 2-KS are similar, but different problems. Therefore, in our setting it is not possible to apply the existing analysis from [3] or from any other k-secretary algorithm directly.

Assumption. For this section we assume that all profits are distinct. This is without loss of generality, as ties can be broken by adjusting the profits slightly, using the items' identifiers. Further, we assume $v_1 > v_2 > \ldots > v_n$ and say that i is the *rank* of item i.

4.1 Packing Types

As outlined above, in contrast to the 2-secretary problem, not all combinations of two knapsack items can be packed completely. Therefore, we analyze the probability that $\mathcal{A}_L$ selects a feasible set of items whose profit can be bounded from below. We restrict our analysis to packings where an item $i \in \{1, 2, 3, 4\}$ is packed as the first item and group such packings into several packing types A-M defined in the following. Although covering more packings might lead to further insights into the problem and to a stronger result, we expect the improvement to be marginal.

Let p_X be the probability that $\mathcal{A}_L$ returns a packing of type $X \in \{\mathrm{A}, \ldots, \mathrm{M}\}$. In addition, let p_i for $i \in [n]$ be the probability that $\mathcal{A}_L$ packs i as the first item. Finally, let p_{ij} for $i, j \in [n]$ be the probability that $\mathcal{A}_L$ packs i as the first item and j as the second item.

In a packing of type A, the items 1 and 2 are packed in any order. Therefore, $p_{\mathrm{A}} = p_{12} + p_{21}$. The types B and C are defined analogously using the items $\{1, 3\}$ and $\{2, 3\}$, respectively. In a packing of type D, the item 1 is accepted as the first item, together with no or any second

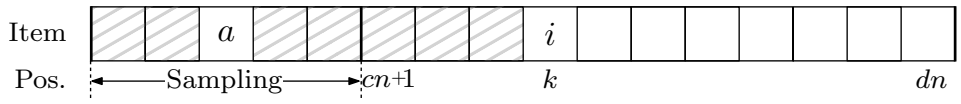


Figure 1 Input sequence considered in Lemma 4.2. The gray dashed slots represent items of rank greater than a.

item j. This happens with probability $p_{\mathrm{D}} = p_1$. Accordingly, we define types E,F, and G using the items 2,3, and 4, respectively. Finally, for each item $i \in \{1,2,3\}$, we introduce two further packing types. For $i = 1$, types H and I characterize packings where the first accepted item is 1, the second accepted item j is not 2 (type H) and not 3 (type I), respectively. Therefore, we get $p_{\mathrm{H}} = p_1 - p_{12}$ and $p_{\mathrm{I}} = p_1 - p_{13}$. Packing types J-K and L-M describe analogous packings for $i = 2$ and $i = 3$, respectively. Table 1 shows all packing types A-M and their probabilities expressed by p_i and p_{ij}.

The packing types defined above allow to describe all packings where a specific item $i \in \{1,2,3,4\}$ is packed as the first item, without covering the same packing multiple times. For example, packing types A and D (with $j = 2$) both include the packing $(1,2)$; however, we can consider the disjoint packing types A and H.

4.2 Acceptance Probabilities of Algorithm 2

In the following we compute the probabilities p_i and p_{ij} from Table 1 as functions of c and d. Throughout the following proofs, we denote the position of an item i in a given permutation with $\mathrm{pos}(i) \in [n]$. Further, let a be the maximum profit item from sampling.

We think of the random permutation as being sequentially constructed. The fact given below follows from the hypergeometric distribution and becomes helpful in the proofs of Lemmas 4.2 and 4.3.

▶ **Fact 4.1.** *Suppose there are N balls in an urn from which M are blue and $N - M$ red. The probability of drawing K blue balls without replacement in a sequence of length K is $h(N,M,K) := \binom{M}{K}/\binom{N}{K}$.*

In the first lemma, we provide the probabilities p_i for $i \in [4]$ assuming $n \to \infty$.

▶ **Lemma 4.2.** *Assuming $n \to \infty$, it holds that*

$$p_i = \begin{cases} c \ln \frac{d}{c} & i = 1 \\ c\left(\ln \frac{d}{c} - d + c\right) & i = 2 \\ c\left(\ln \frac{d}{c} - 2(d-c) + \frac{1}{2}(d^2 - c^2)\right) & i = 3 \\ c\left(\ln \frac{d}{c} - 3(d-c) + \frac{3}{2}(d^2 - c^2) - \frac{1}{3}(d^3 - c^3)\right) & i = 4\,. \end{cases}$$

Proof. We construct the random permutation by drawing the positions for items sequentially, starting with the items i and a. For any position $k \geq cn + 1$, the permutation fulfills $\mathrm{pos}(i) = k$ and $\mathrm{pos}(a) \leq cn$ with probability $\frac{1}{n}\frac{cn}{n-1} = \frac{c}{n-1}$. Next, we draw the remaining $k - 2$ items for the slots up to position k. Since i is packed as the first item, all previous items (except for a) must have rank greater than a (see Figure 1). As these items are drawn from the remaining $n - 2$ items (of which $n - a$ have rank greater than a), the probability for this step is $h(n-2, n-a, k-2)$ according to Fact 4.1. Using the law of total probability for $k \in \{cn+1, \ldots, dn\}$ and $a \in \{i+1, \ldots, n\}$ we obtain

$$p_i = \frac{c}{n-1} \sum_{k=cn+1}^{dn} \sum_{a=i+1}^{n} h(n-2, n-a, k-2) = \frac{c}{n-1} \sum_{k=cn+1}^{dn} \frac{1}{\binom{n-2}{k-2}} \sum_{a=i+1}^{n} \binom{n-a}{k-2}.$$

We can simplify this term further by observing

$$\sum_{a=i+1}^{n} \binom{n-a}{k-2} = \sum_{a=0}^{n-i-1} \binom{a}{k-2} = \binom{n-i}{k-1}.$$

Therefore, $p_i = \frac{c}{n-1} \sum_{k=cn+1}^{dn} \binom{n-i}{k-1} / \binom{n-2}{k-2}$.

Asymptotics. It holds that

$$\lim_{n\to\infty} \frac{\binom{n-i}{k-1}}{\binom{n-2}{k-2}} = \lim_{n\to\infty} \frac{(n-i)!}{(n-2)!} \frac{(n-k)!}{(n-i-k+1)!} \frac{1}{k-1} = \frac{(n-k)^{i-1}}{n^{i-2}} \frac{1}{k}.$$

Hence, $\lim_{n\to\infty} p_i = (c/n^{i-1}) \sum_{k=cn+1}^{dn} f(k)$ where $f(k) := (n-k)^{i-1}/k$. Since f is monotonically decreasing in k, we have $\int_{cn+1}^{dn+1} f(k)\,\mathrm{d}k \leq \sum_{k=cn+1}^{dn} f(k) \leq \int_{cn}^{dn} f(k)\,\mathrm{d}k$. Let F be a function such that $\int_a^b f(k)\,\mathrm{d}k = F(b) - F(a)$ for $0 < a < b$. As it holds that $\lim_{n\to\infty} F(dn+1) - F(dn) = \lim_{n\to\infty} F(cn+1) - F(cn) = 0$, the above bounds are asymptotically tight, i.e., $\lim_{n\to\infty} \sum_{k=cn+1}^{dn} f(k) = F(dn) - F(cn)$. Below we give functions F for $i \in [4]$.

i	$f(k)$	$F(k)$	$F(dn) - F(cn)$
1	$\frac{1}{k}$	$\ln k$	$\ln \frac{d}{c}$
2	$\frac{n-k}{k}$	$n \ln k - k$	$n \ln \frac{d}{c} - dn + cn$
3	$\frac{(n-k)^2}{k}$	$n^2 \ln k - 2nk + \frac{k^2}{2}$	$n^2 \ln \frac{d}{c} - 2n(dn - cn) + \frac{d^2n^2 - c^2n^2}{2}$
4	$\frac{(n-k)^3}{k}$	$n^3 \ln k - 3n^2k + \frac{3}{2}nk^2 - \frac{k^3}{3}$	$n^3 \ln \frac{d}{c} - 3n^3(d-c) + \frac{3}{2}n^3(d^2 - c^2) - \frac{1}{3}n^3(d^3 - c^3)$

The claims follow by multiplying the respective terms with c/n^{i-1}. ◀

Next, we analyze the probabilities p_{ij} for $i \neq j$ and $i, j \in [3]$. The next lemma deals with the cases where $j = i + 1$.

▶ **Lemma 4.3.** *For $n \to \infty$ it holds that*

$$p_{12} = c\left(d - c\ln\frac{d}{c} - c\right),$$
$$p_{23} = c\left(d - c\ln\frac{d}{c} - c - \frac{d^2}{2} + cd - \frac{c^2}{2}\right).$$

The proof of Lemma 4.3 is technically similar to the proof of Lemma 4.2 and thus deferred to Appendix A. It remains to analyze the probabilities p_{13}, p_{31}, p_{21}, and p_{32}. Interestingly, they all reduce to the two probabilities considered in Lemma 4.3. The following two lemmas should be intuitively clear from the description of Algorithm 2. For completeness, we give formal proofs in Appendix A.

▶ **Lemma 4.4.** *For any two items i and j it holds that $p_{ij} = p_{ji}$.*

▶ **Lemma 4.5.** *For any three items $i < k < j$ it holds that $p_{ij} = p_{kj}$.*

Therefore, we have $p_{13} = p_{23}$ by Lemma 4.5 and $p_{31} = p_{13}$, $p_{21} = p_{12}$, and $p_{32} = p_{23}$ by Lemma 4.4.

4.3 Analysis

Let T be the set of items in the optimal packing of I_L. This set may contain a single item, may be a two-item subset of $\{1,2,3\}$, or may be a two-item subset containing an item $j \geq 4$. In the following we analyze the performance of Algorithm 2 for each case.

Single-item case. Let case 1 be the case where $T = \{1\}$. In case 1, $\mathbf{E}[\mathcal{A}_L] \geq p_{\mathrm{D}} \operatorname{OPT}_L$.

Two-item cases. In cases 2–4, we consider packings of the form $T = \{i,j\}$ with $1 \leq i < j \leq 3$. We define cases 2, 3, and 4 as $T = \{1,2\}$, $T = \{1,3\}$, and $T = \{2,3\}$, respectively. We want to consider all algorithmic packings whose profit can be bounded in terms of $\operatorname{OPT}_L = v_i + v_j$. For this purpose, for each case 2-4 we build three groups of feasible packing types, according to whether the profit of a packing is OPT_L, at least v_i, or in the interval $(v_i, v_j]$. We ensure that no packing is counted multiple times by (a) choosing appropriate packing types and (b) grouping these packing types in a disjoint way, according to their profit. Let α_w be the probability that the algorithm returns the optimal packing in case $w \in \{2,3,4\}$. It holds that $\alpha_2 = p_{\mathrm{A}}$, $\alpha_3 = p_{\mathrm{B}}$, and $\alpha_4 = p_{\mathrm{C}}$. In addition, let β_w be the probability that an item $k \leq i$ is packed as the first item in case $w \in \{2,3,4\}$. We have $\beta_2 = p_{\mathrm{H}}$, $\beta_3 = p_{\mathrm{I}}$, and $\beta_4 = p_{\mathrm{D}} + p_{\mathrm{K}}$. Finally, let γ_w be the probability that an item k with $i < k \leq j$ is packed as the first item in case $w \in \{2,3,4\}$. It holds that $\gamma_2 = p_{\mathrm{J}}$, $\gamma_3 = p_{\mathrm{E}} + p_{\mathrm{L}}$, and $\gamma_4 = p_{\mathrm{M}}$.

Finally, we define case 5 as $T = \{i,j\}$ with $i \geq 1$, $j \geq 4$, and $i < j$. In this case, note that packings of type D contain an item of value at least v_i, and packings of type E, F, and G contain an item of value at least v_j. Hence, we can slightly abuse the notation and set $\alpha_5 = 0$, $\beta_5 = p_{\mathrm{D}}$, and $\gamma_5 = p_{\mathrm{E}} + p_{\mathrm{F}} + p_{\mathrm{G}}$, such that it holds that

$$\mathbf{E}[\mathcal{A}_L] \geq \alpha_w(v_i + v_j) + \beta_w v_i + \gamma_w v_j \qquad \text{in case } w \in \{2,3,4,5\}.$$

To bound this term against $\operatorname{OPT}_L = v_i + v_j$, consider the following two cases: If $\beta_w \geq \gamma_w$, we obtain from Chebyshev's sum inequality $\beta_w v_i + \gamma_w v_j \geq \frac{1}{2}(\beta_w + \gamma_w)(v_i + v_j)$. If $\beta_w < \gamma_w$, we trivially have $\beta_w v_i + \gamma_w v_j > \beta_w(v_i + v_j)$. Thus, we obtain

$$\mathbf{E}[\mathcal{A}_L] \geq \left(\alpha_w + \min\left\{\frac{\beta_w + \gamma_w}{2}, \beta_w\right\}\right) \operatorname{OPT}_L \qquad \text{in case } w \in \{2,3,4,5\}. \tag{2}$$

The competitive ratio of $\mathcal{A}_L$ is the minimum over all cases 1-5. We obtain the following two lemmas. If the algorithm is allowed to use the entire input sequence ($d = 1$), $\mathcal{A}_L$ has a competitive ratio of $1/3.08$.

▶ **Lemma 4.6.** *With $c = 0.23053$ and $d = 1$, algorithm $\mathcal{A}_L$ satisfies $\mathbf{E}[\mathcal{A}_L] \geq \frac{1}{3.08} \operatorname{OPT}_L$.*

Note that 2-KS includes the secretary problem (case 1); thus, no algorithm for 2-KS can have a better competitive ratio than $1/e < 1/2.71$. In the final algorithm we set $d < 1$ to benefit from $\mathcal{A}_S$. The next lemma has already been used to prove Theorem 1.1 in Section 3.

▶ **Lemma 4.7.** *With $c = 0.42291$ and $d = 0.64570$, algorithm $\mathcal{A}_L$ satisfies $\mathbf{E}[\mathcal{A}_L] \geq \frac{1}{6.65} \operatorname{OPT}_L$.*

Proof of Lemmas 4.6 and 4.7. For the overall competitive ratio, we build the minimum over all cases. According to inequality (2), the competitive ratios for the two-item cases depend on $\beta_w \geq \gamma_w$ or $\beta_w < \gamma_w$. However, for the parameter pairs $(c,d) = (0.23053, 1)$ from

Table 2 Competitive ratios of Algorithm 2 for the parameters from Lemmas 4.6 and 4.7 in different cases. Bold values indicate the minimum over all cases and thus the competitive ratio.

				two-item cases			
	c	d	case 1	case 2	case 3	case 4	case 5
Lemma 4.6	0.23053	1	0.33827	0.34898	0.32705	0.32705	**0.32471**
Lemma 4.7	0.42291	0.64570	0.17897	**0.15039**	0.16033	0.16033	0.16231

Lemma 4.6 and $(c, d) = (0.42291, 0.64570)$ from Lemma 4.7 we have $\beta_w \geq \gamma_w$ for any case $w \in \{2, 3, 4, 5\}$. This follows from a technical lemma provided in Appendix A (Lemma A.1). Hence, inequality (2) simplifies to $\mathbf{E}[\mathcal{A}_L] \geq \left(\alpha_w + \frac{\beta_w + \gamma_w}{2}\right) \mathrm{OPT}_L$ in case $w \in \{2, 3, 4, 5\}$. Using the definitions of p_X from Table 1 and the symmetry property of Lemma 4.4 we get

$$\mathbf{E}[\mathcal{A}_L] / \mathrm{OPT}_L \geq \begin{cases} p_1 & \text{case 1} \\ p_{12} + (p_1 + p_2)/2 & \text{case 2} \\ p_{13} + (p_1 + p_2 + p_3)/2 & \text{case 3} \\ p_{23} + (p_1 + p_2 + p_3)/2 & \text{case 4} \\ (p_1 + p_2 + p_3 + p_4)/2 & \text{case 5} \end{cases} . \tag{3}$$

Note that the algorithm attains the same competitive ratio in case 3 and 4, since $p_{13} = p_{23}$. Table 2 shows the competitive ratios for all five cases obtained from Equation (3). For the overall competitive ratio, we have

$$\mathbf{E}[\mathcal{A}_L] \geq \min\left\{p_1, p_{12} + \frac{p_1 + p_2}{2}, p_{23} + \frac{p_1 + p_2 + p_3}{2}, \frac{p_1 + p_2 + p_3 + p_4}{2}\right\} \mathrm{OPT}_L .$$

Hence, the competitive ratios are $0.32471 \geq 1/3.08$ and $0.15039 \geq 1/6.65$ for Lemma 4.6 and Lemma 4.7, respectively. ◀

Recall that in Algorithm 1, we can only benefit from $\mathcal{A}_S$ if $\mathcal{A}_L$ has not filled the knapsack completely. Thus, the following property is crucial in the final analysis.

▶ **Lemma 4.8.** *With probability of at least c/d, no item is packed by $\mathcal{A}_L$.*

Proof. Fix any set of dn items arriving in rounds $1, \ldots, dn$. The most profitable item v^* from this set arrives in the sampling phase with probability c/d. If this event occurs, no item in rounds $cn + 1, \ldots, dn$ beats v^* and $\mathcal{A}_L$ will not select any item. ◀

We finally note that our approach from Section 4.1 provides a general framework to obtain algorithms for 2-KS using secretary algorithms with two choices. Although stronger algorithms than Algorithm 2 exist for the 2-secretary objective [3, 12] and similar objectives [40, 42], it is not clear if they would improve the performance of the overall algorithm. More sophisticated algorithms may use weaker thresholds to accept the first item, which decreases the probability considered in Lemma 4.8. This, in turn, reduces the expected profit gained from $\mathcal{A}_S$, as described above.

5 Small Items

For small items, we use solutions for the fractional problem variant and obtain an integral packing via randomized rounding. This approach has been applied successfully to packing LPs [27]; however, for the knapsack problem it is not required to solve LP relaxations in each

round (as in [27]). Instead, here, we build upon solutions of the classical greedy algorithm, which is well-known to be optimal for the fractional knapsack problem. Particularly, this algorithm is both efficient in running time and easy to analyze.

We next formalize the greedy solution for any set T of items. Let the *density* of an item be the ratio of its profit to its size. Consider any list L containing the items from T ordered by non-increasing density. We define the *rank* $\rho(i)$ of item i as its position in L and $\sigma(l)$ as the item at position l in L. Thus, $\sigma(l) = \rho^{-1}(l)$ denotes the l-th densest item. Let k be such that $\sum_{i=1}^{k-1} s_{\sigma(i)} < W \leq \sum_{i=1}^{k} s_{\sigma(i)}$. The fraction of item i in the greedy solution α is now defined as

$$\alpha_i = \begin{cases} 1 & \text{if } \rho(i) < k \\ \left(W - \sum_{i=1}^{k-1} s_{\sigma(i)}\right) / s_i & \text{if } \rho(i) = k \\ 0 & \text{else}\,, \end{cases}$$

i.e., we pack the $k-1$ densest items integrally and fill the remaining space by the maximum feasible fraction of the k-th densest item. Let $\mathrm{OPT}(T)$ and $\mathrm{OPT}^*(T)$ denote the profits of optimal integral and fractional packings of T, respectively. It is not hard to see that α satisfies $\sum_{i \in T} \alpha_i v_i = \mathrm{OPT}^*(T) \geq \mathrm{OPT}(T)$ and $\sum_{i \in T} \alpha_i s_i = W$.

5.1 Algorithm

The algorithm $\mathcal{A}_S$ for small items, which is formally defined in Algorithm 3, works as follows. After a sampling phase of dn rounds, in each round $\ell \geq dn + 1$ the algorithm computes a greedy solution $x^{(\ell)}$ for $I_S(\ell)$. Here, $I_S(\ell)$ denotes the subset of I_S revealed up to round ℓ. The algorithm packs the current online item i with probability $x_i^{(\ell)}$. However, generally, this can only be done if the remaining capacity of the knapsack is at least $\delta W \geq s_i$.

Note that in case of an integral coefficient $x_i^{(\ell)} \in \{0, 1\}$, the packing step is completely deterministic. Moreover, in any greedy solution $x^{(\ell)}$, there is at most one item i with fractional coefficient $x_i^{(\ell)} \in (0, 1)$. Therefore, in expectation, there is only a small number of rounds where the algorithm actually requests randomness.

▶ **Observation 5.1.** *Let X denote the number of rounds where Algorithm 3 packs an item with probability $x_i \in (0, 1)$. It holds that $\mathbf{E}[X] \leq \ln(1/d) \leq 0.44$.*

Proof. Consider any round ℓ and let $x^{(\ell)}$ be the greedy knapsack solution computed by Algorithm 3. By definition of $x^{(\ell)}$, at most one of the ℓ visible items has a fractional coefficient $x_i^{(\ell)} \in (0, 1)$. The probability that this item i arrives in round ℓ is $1/\ell$ in a random permutation. Let X_ℓ be an indicator variable for the event that Algorithm 3 packs an item at random in round ℓ. By the above argument, we have $\mathbf{Pr}[X_\ell = 1] \leq 1/\ell$. Since Algorithm 3 selects items starting in round $dn+1$, we obtain $\mathbf{E}[X] = \sum_{\ell=dn+1}^{n} \mathbf{E}[X_\ell] \leq \sum_{\ell=dn+1}^{n} \frac{1}{\ell} \leq \ln \frac{1}{d} \leq 0.44\,.$ ◀

Note that Algorithm 2 and the sequential approach (Algorithm 1) are deterministic algorithms. Therefore, our overall algorithm requests randomness in expectation in less than one round.

5.2 Analysis

Let α be the greedy (offline) solution for I_S and set $\Delta = \frac{1}{1-\delta}$. Recall that in round $dn + 1$, the knapsack might already have been filled by $\mathcal{A}_L$ with large items in previous rounds. For now, we assume an empty knapsack after round dn and define this event as ξ. In the final analysis, we will use the fact that $\mathbf{Pr}[\xi]$ can be bounded from below, which is according to Lemma 4.8.

Algorithm 3 Algorithm $\mathcal{A}_S$ for small items.

Input : Random permutation of n (1/3)-small items, a knapsack of capacity W, parameter $d \in (0,1)$.
Output : A feasible (integral) packing of the knapsack.
Let ℓ be the current round and i be the online item of round ℓ.
if $\ell \leq dn$ **then**
  Sampling phase – discard all items.
if $dn+1 \leq \ell \leq n$ **then**
  Let $x^{(\ell)}$ be the greedy solution for $I_S(\ell)$.
  if *the remaining capacity is at least* δW **then**
    Pack i with probability $x_i^{(\ell)}$.

▶ **Lemma 5.2.** *Let $i \in I_S$ and $E_i(\ell)$ be the event that the item i is packed by $\mathcal{A}_S$ in round ℓ. For $\ell \geq dn+1$, it holds that* $\mathbf{Pr}\left[E_i(\ell) \mid \xi\right] \geq \frac{1}{n}\alpha_i(1-\Delta\ln\frac{\ell}{dn})$.

Proof. In a random permutation, item i arrives in round ℓ with probability $1/n$. In round $\ell \geq dn+1$, the algorithm decides to pack i with probability $x_i^{(\ell)}$. Note that the rank of item i in $I_S(\ell)$ is less or equal to its rank in I_S. According to the greedy solution's definition, this implies $x_i^{(\ell)} \geq \alpha_i$. Finally, the δ-small item i can be packed successfully if the current resource consumption X is at most $(1-\delta)W$. In the following, we investigate the expectation of X to give a probability bound using Markov's inequality at the end of this proof.

Let X_k be the resource consumption in round $k < \ell$. By assumption, the knapsack is empty after round dn, we have $X = \sum_{k=dn+1}^{\ell-1} X_k$. Let Q be the set of k visible items in round k. The set Q can be seen as uniformly drawn from all k-item subsets and any item $j \in Q$ is the current online item of round k with probability $1/k$. The algorithm packs any item j with probability $x_j^{(k)}$, thus

$$\mathbf{E}\left[X_k\right] = \sum_{j\in Q} \mathbf{Pr}\left[j \text{ occurs in round } k\right] s_j x_j^{(k)} = \frac{1}{k}\sum_{j\in Q} s_j x_j^{(k)} \leq \frac{W}{k}\,,$$

where the last inequality holds because $x^{(k)}$ is a feasible solution for a knapsack of size W. By the linearity of expectation and the previous equation, the expected resource consumption up to round ℓ is $\mathbf{E}\left[X\right] = \sum_{k=dn+1}^{\ell-1}\mathbf{E}\left[X_k\right] \leq \sum_{k=dn+1}^{\ell-1}\frac{W}{k} \leq W\ln\frac{\ell}{dn}$. Using Markov's inequality, we obtain finally

$$\mathbf{Pr}\left[X < (1-\delta)W\right] = 1 - \mathbf{Pr}\left[X \geq (1-\delta)W\right] \geq 1 - \frac{\mathbf{E}\left[X\right]}{(1-\delta)W} \geq 1 - \Delta\ln\frac{\ell}{dn}\,. \qquad \blacktriangleleft$$

Using Lemma 5.2 we easily obtain the total probability that a specific item will be packed.

▶ **Lemma 5.3.** *Let $i \in I_S$ and E_i be the event that the item i is packed by $\mathcal{A}_S$. It holds that* $\mathbf{Pr}\left[E_i \mid \xi\right] \geq \alpha_i\left((1-d)(1+\Delta) - \Delta\ln\frac{1}{d}\right)$.

Proof. Summing the probabilities from Lemma 5.2 over all rounds $\ell \geq dn+1$ gives

$$\begin{aligned}\mathbf{Pr}\left[E_i \mid \xi\right] &= \sum_{\ell=dn+1}^{n}\mathbf{Pr}\left[E_i(\ell)\mid\xi\right] \geq \sum_{\ell=dn+1}^{n}\frac{1}{n}\alpha_i\left(1-\Delta\ln\frac{\ell}{dn}\right)\\ &= \frac{1}{n}\alpha_i\left(n - dn - \Delta\sum_{\ell=dn+1}^{n}\ln\frac{\ell}{dn}\right) = \alpha_i\left(1-d-\frac{\Delta}{n}\sum_{\ell=dn+1}^{n}\ln\frac{\ell}{dn}\right).\end{aligned}$$

Since $\ln \frac{\ell}{dn}$ is monotonically increasing in ℓ, we can bound the last sum by the corresponding integral:

$$\sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn} \leq \int_{\ell=dn+1}^{n+1} \ln \frac{\ell}{dn} \, \mathrm{d}\ell = (n+1) \ln \frac{n+1}{dn} - (n+1) - (dn+1) \ln \frac{dn+1}{dn} + (dn+1) \, .$$

This implies $\lim_{n\to\infty} \frac{\Delta}{n} \sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn} \leq \Delta \left(\ln \frac{1}{d} - 1 + d \right)$. Rearranging terms gives the claim. ◀

The following lemma bounds the expected profit of the packing of $\mathcal{A}_S$, assuming ξ.

▶ **Lemma 5.4.** *It holds that* $\mathbf{E}\left[\mathcal{A}_S \mid \xi\right] \geq \left((1-d)(1+\Delta) - \Delta \ln \frac{1}{d}\right) \mathrm{OPT}_S$.

Proof. Let $\beta = (1-d)(1+\Delta) - \Delta \ln \frac{1}{d}$. By Lemma 5.3, the probability that an item i gets packed is $\mathbf{Pr}\left[E_i \mid \xi\right] \geq \alpha_i \beta$. Therefore,

$$\mathbf{E}\left[\mathcal{A}_S \mid \xi\right] = \sum_{i \in I_S} \mathbf{Pr}\left[E_i \mid \xi\right] v_i \geq \sum_{i \in I_S} \alpha_i \beta v_i \geq \beta \, \mathrm{OPT}_S \, . \qquad ◀$$

The conditioning on ξ can be resolved using Lemma 4.8. Thus we obtain the following lemma, which is the second pillar in the proof of Theorem 1.1 and concludes this section.

▶ **Lemma 5.5.** *With* $c = 0.42291$ *and* $d = 0.64570$*, we have* $\mathbf{E}\left[\mathcal{A}_S\right] \geq \frac{1}{6.65} \mathrm{OPT}_S$.

Proof. By Lemma 4.8, the probability for an empty knapsack after round dn is $\mathbf{Pr}\left[\xi\right] \geq \frac{c}{d}$. Thus, from Lemma 5.4 with $\Delta = \frac{1}{1-1/3} = \frac{3}{2}$, we obtain

$$\mathbf{E}\left[\mathcal{A}_S\right] = \mathbf{Pr}\left[\xi\right] \mathbf{E}\left[\mathcal{A}_S \mid \xi\right] = \frac{c}{d} \left(\frac{5}{2}(1-d) - \frac{3}{2} \ln \frac{1}{d} \right) \mathrm{OPT}_S \geq \frac{1}{6.65} \mathrm{OPT}_S \, . \qquad ◀$$

6 Extension to GAP

In this section we show that the sequential approach introduced in Section 3 can be easily adapted to GAP, yielding a (1/6.99)-competitive randomized algorithm. We first define the problem formally.

GAP. We are given a set of items $I = [n]$ and a set of resources $R = [m]$ of capacities $W_r \in \mathbb{Q}_{>0}$ for $r \in R$. If item $i \in I$ is assigned to resource $r \in R$, this raises profit (value) $v_{i,r} \in \mathbb{Q}_{\geq 0}$, but consumes $s_{i,r} \in \mathbb{Q}_{>0}$ of the resource's capacity. The goal is to assign each item to at most one resource such that the total profit is maximized and no resource exceeds its capacity. We call the tuple $(v_{i,r}, s_{i,r})$ an *option* of item i and w.l.o.g. assume that options for all resources exist. This can be ensured by introducing dummy options with $v_{i,r} = 0$. In the online version of the problem, in each round an item is revealed together with its set of options. The online algorithm must decide immediately and irrevocably, if the item is assigned. If so, it has to specify the resource according to one of its options.

Again, we construct restricted instances I_L and I_S according to the following definition, which generalizes Definition 2.1. Let $\delta \in (0, 1)$.

▶ **Definition 6.1.** *We call an option* $(v_{i,r}, s_{i,r})$ δ*-large if* $s_{i,r} > \delta W_r$ *and* δ*-small if* $s_{i,r} \leq \delta W_r$*. Whenever* δ *is clear from the context, we say an option is large or small for short. Based on a given instance* I *for GAP, we define two modified instances* I_L *and* I_S *which are obtained from* I *as follows.*

- I_L*: Replace each small option* $(v_{i,r}, s_{i,r})$ *by the large option* $(0, W_r)$*.*
- I_S*: Replace each large option* $(v_{i,r}, s_{i,r})$ *by the small option* $(0, \delta)$*.*

Thus, I_L only contains large options and I_S only contains small options. However, by construction no algorithm will assign an item according to a zero-profit option. We define OPT, OPT_L, and OPT_S accordingly. Note that the inequality $\mathrm{OPT} \leq \mathrm{OPT}_L + \mathrm{OPT}_S$ holds also for GAP.

The sequential framework of Algorithm 1 can be adapted in a straightforward manner by replacing terms like *packing* with *assignment to resource* r. Here, we set the threshold parameter to $\delta = 1/2$. In the following subsections, we specify algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ for $(1/2)$-large and $(1/2)$-small options, respectively.

6.1 Large Options

If each item consumes more than one half of a resource, no two items can be assigned to this resource. Thus, we obtain the following matching problem.

Edge-weighted bipartite matching problem. Given a bipartite graph $G = (L \cup R, E)$ and a weighting function $w\colon E \to \mathbb{Q}_{\geq 0}$, the goal is to find a bipartite matching $M \subseteq E$ such that $w(M) := \sum_{e \in M} w(e)$ is maximal. In the online version, the (offline) nodes from R and the number $n = |L|$ are known in advance, whereas the nodes from L are revealed online together with their incident edges. In the case of GAP, L is the set of items, R is the set of resources, and the weight of an edge $e = \{l, r\}$ is $w(e) = v_{l,r}$, i.e., the profit gained from assigning item l to resource r.

Under random arrival order, Kesselheim et al. [26] developed an optimal $(1/e)$-competitive algorithm for this problem. Adapting this algorithm to the sequential approach with parameters c and d leads to the following algorithm $\mathcal{A}_L$: After sampling the first cn nodes, in each round ℓ the algorithm computes a maximum edge-weighted matching $M^{(\ell)}$ for the graph revealed up to this round. Let $l \in L$ be the online vertex of round ℓ. If l is matched in $M^{(\ell)}$ to some node $r \in R$, we call $e^{(\ell)} = \{l, r\}$ the *tentative edge* of round ℓ. Now, if r is still unmatched and $\ell \leq dn$, the tentative edge is added to the matching.

A formal description of this algorithm is given in Appendix B.1. The proof of the approximation guarantee relies mainly on the following two lemmas; for completeness, we give the proofs from [26] in Appendix B.1. The first lemma shows that the expected weight of any tentative edge can be bounded from below.

▶ **Lemma 6.2** ([26]). *In any round ℓ, the tentative edge (if it exists) has expected weight* $\mathbf{E}\left[w(e^{(\ell)})\right] \geq \frac{1}{n}\,\mathrm{OPT}_L$.

However, we only gain the weight of the tentative edge $e^{(\ell)} = \{l, r\}$ if it can be added to the matching, i.e., if r has not been matched previously. The next lemma bounds the probability for this event from below.

▶ **Lemma 6.3** ([26]). *Let $\xi(r, \ell)$ be the event that the offline vertex $r \in R$ is unmatched after round ℓ. It holds that* $\mathbf{Pr}\left[\xi(r, \ell)\right] \geq \frac{cn}{\ell}$.

Using Lemmas 6.2 and 6.3, we can bound the competitive ratio of $\mathcal{A}_L$ in the following lemma. Note that we obtain the optimal algorithm from [26] for $c = 1/e$ and $d = 1$.

▶ **Lemma 6.4.** *For $n \to \infty$, it holds that* $\mathbf{E}\left[\mathcal{A}_L\right] \geq c \ln \frac{d}{c}\,\mathrm{OPT}_L$.

Proof. Let A_ℓ be the gain of the matching weight in round ℓ. As the tentative edge $e^{(\ell)} = \{l, r\}$ can only be added if r has not been matched in a previous round, we have $\mathbf{E}\left[A_\ell\right] = \mathbf{E}\left[w(e^{(\ell)})\right] \mathbf{Pr}\left[\xi(r, \ell)\right]$ for the event $\xi(r, \ell)$ from Lemma 6.3. Therefore, from

Lemmas 6.2 and 6.3 we have $\mathbf{E}[A_\ell] \geq \frac{1}{n}\operatorname{OPT}_L \frac{cn}{\ell} = \frac{c}{\ell}\operatorname{OPT}_L$. Summing over all rounds from $cn+1$ to dn yields

$$\mathbf{E}[\mathcal{A}_L] = \sum_{\ell=cn+1}^{dn} \mathbf{E}[A_\ell] \geq \left(c \sum_{\ell=cn+1}^{dn} \frac{1}{\ell}\right) \operatorname{OPT}_L \geq c \ln \frac{dn+1}{cn+1} \operatorname{OPT}_L \ .$$

Here, in the last step we used the fact $\sum_{\ell=cn+1}^{dn} \frac{1}{\ell} \geq \int_{cn+1}^{dn+1} \frac{1}{\ell}\,\mathrm{d}\ell = \ln \frac{dn+1}{cn+1}$. The claim follows by $\lim_{n\to\infty} \ln \frac{dn+1}{cn+1} = \ln \frac{d}{c}$. ◀

6.2 Small Options

For δ-small options we use the LP-based algorithm $\mathcal{A}_S$ from [27, Sec. 3.3]. On a high level, this algorithm works as follows: After a sampling phase of dn rounds, in each round ℓ the algorithm computes an optimal fractional solution for the instance revealed so far and uses the coefficients as probabilities for an integral assignment. In Appendix B.2 we prove the following lemma, where $\Delta = \frac{1}{1-\delta}$.

▶ **Lemma 6.5.** *For $n \to \infty$, it holds that $\mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left((1+\Delta)(1-d) - \Delta \ln \frac{1}{d}\right)\operatorname{OPT}_S$.*

Note that we obtain basically the same competitive ratio as in Lemma 5.4. Since Lemma 6.5 already addresses possible resource consumption due to assignments made by $\mathcal{A}_L$ in earlier rounds, the factor c/d arises (see Lemma 6.3).

6.3 Proof of Theorem 1.2

Finally, we prove our main theorem for GAP.

Proof of Theorem 1.2. We set the threshold between large and small options to $\delta = 1/2$ and consider Algorithm 1 with the algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ as defined previously. By Lemma 6.4, the expected gain of profit in rounds $cn+1, \ldots, dn$ is $\mathbf{E}[\mathcal{A}_L] \geq c \ln \frac{d}{c} \operatorname{OPT}_L$. Further, we gain $\mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left((1+\Delta)(1-d) - \Delta \ln \frac{1}{d}\right)\operatorname{OPT}_S$ with $\Delta = 2$ in the following rounds, according to Lemma 6.5. For parameters $c = 0.5261$ and $d = 0.6906$, we obtain $c \ln \frac{d}{c} \geq \frac{c}{d}\left(3(1-d) - 2\ln \frac{1}{d}\right)$ and thus, using $\operatorname{OPT}_L + \operatorname{OPT}_S \geq \operatorname{OPT}$,

$$\mathbf{E}[\mathcal{A}_L] + \mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left(3(1-d) - 2\ln \frac{1}{d}\right)(\operatorname{OPT}_L + \operatorname{OPT}_S) \geq \frac{1}{6.99}\operatorname{OPT} \ . \qquad ◀$$

References

1 Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A Dynamic Near-Optimal Algorithm for Online Linear Programming. *Operations Research*, 62(4):876–890, 2014.

2 Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. The Online Stochastic Generalized Assignment Problem. In *Proc. 16th International Workshop on Approximation, Randomization, and Combinatorial Optimization and 17th International Workshop on Randomization and Computation (APPROX/RANDOM)*, pages 11–25, 2013.

3 Susanne Albers and Leon Ladewig. New results for the k-secretary problem. Unpublished manuscript, 2018.

4 Moshe Babaioff, Jason Hartline, and Robert Kleinberg. Selling banner ads: Online algorithms with buyback. In *Fourth Workshop on Ad Auctions*, 2008.

5 Moshe Babaioff, Jason D. Hartline, and Robert D. Kleinberg. Selling ad campaigns: online algorithms with cancellations. In *Proc. 10th ACM Conference on Electronic Commerce (EC)*, pages 61–70, 2009.

6 Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. A Knapsack Secretary Problem with Applications. In *Proc. 10th International Workshop on Approximation, Randomization, and Combinatorial Optimization and 11th International Workshop on Randomization and Computation (APPROX/RANDOM)*, pages 16–28, 2007.

7 Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. Matroid Secretary Problems. *Journal of the ACM*, 65(6):35:1–35:26, 2018.

8 Bahman Bahmani, Aranyak Mehta, and Rajeev Motwani. A 1.43-Competitive Online Graph Edge Coloring Algorithm in the Random Order Arrival Model. In *Proc. 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 31–39, 2010.

9 Christian Borgs, Jennifer T. Chayes, Nicole Immorlica, Kamal Jain, Omid Etesami, and Mohammad Mahdian. Dynamics of bid optimization in online advertisement auctions. In *Proc. 16th International Conference on World Wide Web (WWW)*, pages 531–540, 2007.

10 Niv Buchbinder and Joseph Naor. Online Primal-Dual Algorithms for Covering and Packing. *Math. Oper. Res.*, 34(2):270–286, 2009.

11 Dirk G. Cattrysse and Luk N. Van Wassenhove. A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60(3):260–272, 1992.

12 T.-H. Hubert Chan, Fei Chen, and Shaofeng H.-C. Jiang. Revealing Optimal Thresholds for Generalized Secretary Problem via Continuous LP: Impacts on Online K-Item Auction and Bipartite K-Matching with Random Arrival Order. In *Proc. 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1169–1188, 2015.

13 Chandra Chekuri and Sanjeev Khanna. A Polynomial Time Approximation Scheme for the Multiple Knapsack Problem. *SIAM Journal on Computing (SICOMP)*, 35(3):713–728, 2005.

14 Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79, 2017.

15 Eugene B Dynkin. The optimum choice of the instant for stopping a Markov process. *Soviet Mathematics*, 4:627–629, 1963.

16 Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Clifford Stein. Online Stochastic Packing Applied to Display Ad Allocation. In *Proc. 18th Annual European Symposium on Algorithms (ESA)*, pages 182–194, 2010.

17 Jon Feldman, Nitish Korula, Vahab S. Mirrokni, S. Muthukrishnan, and Martin Pál. Online Ad Assignment with Free Disposal. In *Proc. 5th International Workshop Internet and Network Economics (WINE)*, pages 374–385, 2009.

18 P.R. Freeman. The secretary problem and its extensions: A review. *International Statistical Review/Revue Internationale de Statistique*, pages 189–206, 1983.

19 Waldo Gálvez, Fabrizio Grandoni, Sandy Heydrich, Salvatore Ingala, Arindam Khan, and Andreas Wiese. Approximating Geometric Knapsack via L-Packings. In *Proc. 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 260–271, 2017.

20 Oliver Göbel, Thomas Kesselheim, and Andreas Tönnis. Online Appointment Scheduling in the Random Order Model. In *Proc. 23rd Annual European Symposium on Algorithms (ESA)*, pages 680–692, 2015.

21 Xin Han, Yasushi Kawase, and Kazuhisa Makino. Online Unweighted Knapsack Problem with Removal Cost. *Algorithmica*, 70(1):76–91, 2014.

22 Xin Han, Yasushi Kawase, and Kazuhisa Makino. Randomized algorithms for online knapsack problems. *Theoretical Computer Science*, 562:395–405, 2015.

23 Kazuo Iwama and Shiro Taketomi. Removable Online Knapsack Problems. In *Proc. 29th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 293–305, 2002.

24 Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack problems*. Springer, 2004.

25 Claire Kenyon. Best-Fit Bin-Packing with Random Order. In *Proc. 7th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 359–364, 1996.

26 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An Optimal Online Algorithm for Weighted Bipartite Matching and Extensions to Combinatorial Auctions. In *Proc. 21st Annual European Symposium on Algorithms (ESA)*, pages 589–600, 2013.

27 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. Primal Beats Dual on Online Packing LPs in the Random-Order Model. *SIAM J. Comput.*, 47(5):1939–1964, 2018.

28 Samir Khuller, Stephen G. Mitchell, and Vijay V. Vazirani. On-Line Algorithms for Weighted Bipartite Matching and Stable Marriages. *Theoretical Computer Science*, 127(2):255–267, 1994.

29 Robert D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proc. 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 630–631, 2005.

30 Nitish Korula, Vahab S. Mirrokni, and Morteza Zadimoghaddam. Online Submodular Welfare Maximization: Greedy Beats 1/2 in Random Order. *SIAM J. Comput.*, 47(3):1056–1086, 2018.

31 Denis V Lindley. Dynamic programming and decision theory. *Applied Statistics*, pages 39–51, 1961.

32 George S. Lueker. Average-Case Analysis of Off-Line and On-Line Knapsack Problems. *J. Algorithms*, 29(2):277–305, 1998.

33 Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *Proc. 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 597–606, 2011.

34 Alberto Marchetti-Spaccamela and Carlo Vercellis. Stochastic on-line knapsack problems. *Mathematical Programming*, 68:73–104, 1995.

35 Silvano Martello and Paolo Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., New York, NY, USA, 1990.

36 Aranyak Mehta, Amin Saberi, Umesh V. Vazirani, and Vijay V. Vazirani. AdWords and generalized online matching. *Journal of the ACM*, 54(5):22, 2007.

37 Adam Meyerson. Online Facility Location. In *Proc. 42nd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 426–431, 2001.

38 Vahab S. Mirrokni, Shayan Oveis Gharan, and Morteza Zadimoghaddam. Simultaneous approximations for adversarial and stochastic online budgeted allocation. In *Proc. 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1690–1701, 2012.

39 Marco Molinaro and R. Ravi. The Geometry of Online Packing Linear Programs. *Math. Oper. Res.*, 39(1):46–59, 2014.

40 ML Nikolaev. On a generalization of the best choice problem. *Theory of Probability & Its Applications*, 22(1):187–190, 1977.

41 Temel Öncan. A Survey of the Generalized Assignment Problem and Its Applications. *Information Systems and Operational Research INFOR*, 45(3):123–141, 2007.

42 Mitsushi Tamaki. Recognizing both the maximum and the second maximum of a sequence. *Journal of Applied Probability*, 16(4):803–812, 1979.

43 Rahul Vaze. Online knapsack problem and budgeted truthful bipartite matching. In *Proc. IEEE Conference on Computer Communications (INFOCOM) 2017*, pages 1–9, 2017.

44 Rahul Vaze. Online Knapsack Problem Under Expected Capacity Constraint. In *Proc. IEEE Conference on Computer Communications (INFOCOM) 2018*, pages 2159–2167, 2018.

45 Yunhong Zhou, Deeparnab Chakrabarty, and Rajan M. Lukose. Budget Constrained Bidding in Keyword Auctions and Online Knapsack Problems. In *Proc. 4th International Workshop Internet and Network Economics (WINE)*, pages 566–576, 2008.

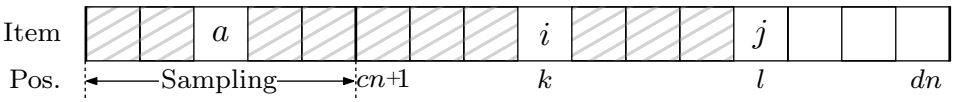


Figure 2 Input sequence considered in Lemma 4.3. The gray dashed slots represent items of rank greater than a.

A Missing Proofs for the Knapsack Result

Proof of Lemma 4.3. Let $i \in [n-1]$ and $j = i+1$. The proof follows the same structure as the proof of Lemma 4.2. Again, we construct the permutation by drawing the positions for items i, j, a first and afterwards all remaining items with position up to $\text{pos}(j)$. Fix positions $k = \text{pos}(i)$ and $l = \text{pos}(j)$. Again, $\text{pos}(a) \leq cn$ must hold by definition of a. The probability that a random permutation satisfies these three position constraints is $\beta := \frac{1}{n}\frac{1}{n-1}\frac{cn}{n-2}$. All remaining items up to position l must have rank greater than a (see Figure 2). Thus we need to draw $l-3$ items from a set of $n-3$ remaining items, from which $n-a$ have rank greater than a. This happens with probability $h(n-3, n-a, l-3)$. Using the law of total probability for $cn+1 \leq k < l \leq dn$ and $a \in \{j+1, \dots, n\}$, we obtain

$$\begin{aligned} p_{ij} &= \beta \sum_{k=cn+1}^{dn-1} \sum_{l=k+1}^{dn} \sum_{a=j+1}^{n} h(n-3, n-a, l-3) \\ &= \beta \sum_{k=cn+1}^{dn-1} \sum_{l=k+1}^{dn} \frac{1}{\binom{n-3}{l-3}} \sum_{a=j+1}^{n} \binom{n-a}{l-3} = \beta \sum_{k=cn+1}^{dn-1} \sum_{l=k+1}^{dn} \frac{\binom{n-j}{l-2}}{\binom{n-3}{l-3}}, \end{aligned}$$

where in the last step we used the equality $\sum_{a=j+1}^{n} \binom{n-a}{l-3} = \sum_{a=0}^{n-j-1} \binom{a}{l-3} = \binom{n-j}{l-2}$.

We next consider the asymptotic setting $n \to \infty$. For this purpose, we define $Q(l) = \binom{n-j}{l-2}/\binom{n-3}{l-3}$. For $(i,j) = (1,2)$ we have $Q(l) = \binom{n-2}{l-2}/\binom{n-3}{l-3} = \frac{n-2}{l-2}$. The sum $\sum_{l=k+1}^{dn} \frac{n-2}{l-2}$ converges to $n \ln \frac{dn}{k}$ for $n \to \infty$. Further, $\lim_{n\to\infty} \sum_{k=cn+1}^{dn-1} n \ln \frac{dn}{k} = n\left(F(dn) - F(cn)\right)$ for $F(x) := x \ln \frac{dn}{x} + x$. Hence,

$$\lim_{n\to\infty} p_{12} = \lim_{n\to\infty} \beta n \left(dn \ln \frac{dn}{dn} + dn - cn \ln \frac{dn}{cn} - cn \right) = c\left(d - c \ln \frac{d}{c} - c \right).$$

In the case $(i,j) = (2,3)$ it holds that $Q(l) = \binom{n-3}{l-2}/\binom{n-3}{l-3} = \frac{n-l}{l-2}$ and we have $\lim_{n\to\infty} \sum_{l=k+1}^{dn} \frac{n-l}{l-2} = n \ln \frac{dn}{k} - dn + k$. Let $F(x) := nx\left(\ln \frac{dn}{x} - d + 1\right) + \frac{x^2}{2}$. Again, by bounding the sum by the corresponding integral we obtain

$$\begin{aligned} &\lim_{n\to\infty} \sum_{k=cn+1}^{dn} n \ln \frac{dn}{k} - dn + k \\ &= F(dn) - F(cn) \\ &= dn^2 \left(\ln \frac{dn}{dn} - d + 1 \right) + \frac{d^2n^2}{2} - cn^2 \left(\ln \frac{dn}{cn} - d + 1 \right) - \frac{c^2n^2}{2} \\ &= n^2 \left(-d^2 + d + \frac{d^2}{2} - c \ln \frac{d}{c} + cd - c - \frac{c^2}{2} \right) \\ &= n^2 \left(d - c \ln \frac{d}{c} - c - \frac{d^2}{2} + cd - \frac{c^2}{2} \right). \end{aligned}$$

Multiplying the last term with $\lim_{n\to\infty} \beta = c/n^2$ gives the claim for p_{23}. ◀

Proof of Lemma 4.4. Suppose i is accepted first and j is accepted as the second item in the input sequence π. Consider the sequence π' obtained from π by swapping i with j. Since j and i are the first two elements beating the best sampling item in π', Algorithm 2 will select j and i on input π'. Hence, the number of permutations must be the same for both events, which implies the claim. ◀

Proof of Lemma 4.5. The argument is similar to the proof of Lemma 4.4. Consider any input sequence π where i is selected first and j second. We know that the best item a from sampling has profit $v_a < v_j < v_i$ and thus any item k with $i < k < j$ must occur after j. Let π' be the sequence obtained from π by swapping i with k. Now, i is behind k and j, thus Algorithm 2 will accept k and j. Again, this proves $p_{ij} = p_{kj}$ since the numbers of corresponding permutations are equal. ◀

The next lemma is used in the proof of Lemma 4.7 to show that for the given lists of parameters, we have $\beta_w \geq \gamma_w$.

▶ **Lemma A.1.** *Let $f(x) = 2\ln x - 6x + 2x^2 - \frac{x^3}{3}$. For parameters c,d with $f(c) \geq f(d)$ it holds that $\beta_w \geq \gamma_w$ where $2 \leq w \leq 5$.*

Proof. The function f is chosen in a way that $f(c) \geq f(d)$ is equivalent to $\beta_5 \geq \gamma_5$. This can be verified easily, using $\beta_5 = p_{\mathrm{D}} = p_1$, $\gamma_5 = p_{\mathrm{E}} + p_{\mathrm{F}} + p_{\mathrm{G}} = p_2 + p_3 + p_4$, and Lemma 4.2. Therefore, the claim for $w = 5$ holds by assumption. For $2 \leq w \leq 4$, the claims follow immediately from $f(c) \geq f(d)$ and the symmetry property of Lemma 4.4:

$$\beta_2 = p_{\mathrm{H}} = p_1 - p_{12} = p_1 - p_{21} \geq p_2 - p_{21} = p_{\mathrm{J}} = \gamma_2$$
$$\beta_3 = p_{\mathrm{I}} = p_1 - p_{13} = p_1 - p_{31} \geq p_2 + p_3 - p_{31} = p_{\mathrm{E}} + p_{\mathrm{L}} = \gamma_3$$
$$\beta_4 = p_{\mathrm{D}} + p_K = p_1 + p_2 - p_{23} \geq p_1 - p_{32} \geq p_3 - p_{32} = p_{\mathrm{M}} = \gamma_4\,. \qquad ◀$$

B Missing Proofs for the GAP Result

B.1 Large Options

Algorithm 4 Algorithm for edge-weighted bipartite matching from [26] (extended by our parameters c, d).

Input : Offline vertex set R, number of online vertices $n = |L|$,
parameters $c, d \in (0, 1)$ with $c < d$.
Output : Matching M.
Set $M = \emptyset$.
Let ℓ be the current round and l be the online vertex of round ℓ.
if $1 \leq \ell \leq cn$ **then**
  Sampling phase – do not add any edge.
if $cn + 1 \leq \ell \leq dn$ **then**
  Let $M^{(\ell)}$ be a maximum-weight matching for the graph in round ℓ.
  Let $e^{(\ell)} \in M^{(\ell)}$ be the edge incident to l.
  if $M \cup e^{(\ell)}$ *is a matching* **then**
    Add $e^{(\ell)}$ to M.
  if $\ell > dn$ **then**
    Do not add any edge.

Proof of Lemma 6.2. Let $e^{(\ell)}$ be the tentative edge of round ℓ and let $Q \subseteq L$ with $|Q| = \ell$ be the set of visible vertices from this round. Since each vertex from Q has the same probability of $1/\ell$ to arrive in round ℓ, we have

$$\mathbf{E}\left[w(e^{(\ell)})\right] = \sum_{e=\{l,r\}\in M^{(\ell)}} \mathbf{Pr}\left[l \text{ arrives in round } \ell\right] w(e) = \frac{1}{\ell} w(M^{(\ell)})\,. \tag{4}$$

Let $M^* = M^{(n)}$ be a maximum weight (offline) matching and $M^*_Q = \{e = \{l,r\} \in M^* \mid l \in Q\}$. We have $w(M^{(\ell)}) \geq w(M^*_Q)$, since $M^{(\ell)}$ is an optimal and M^*_Q a feasible matching for the graph revealed in round ℓ. As Q can be seen as uniformly drawn among all ℓ-element subsets, each vertex l has probability ℓ/n to be in Q. It follows

$$\mathbf{E}\left[w(M^{(\ell)})\right] \geq \mathbf{E}\left[w(M^*_Q)\right] = \sum_{e=\{l,r\}\in M^*} \mathbf{Pr}\left[l \in Q\right] w(e) = \frac{\ell}{n} w(M^*)\,. \tag{5}$$

Combining (4) and (5) concludes the proof. ◀

Proof of Lemma 6.3. In each round k, the vertex r can only be matched if it is incident to the tentative edge $e^{(k)} \in M^{(k)}$ of this round, i.e., $e^{(k)} = \{l,r\}$ where $l \in L$ is the online vertex of round k. As l can be seen as uniformly drawn among all k visible nodes (particularly, independent from the order of the previous $k-1$ items), l has probability $1/k$ to arrive in round k. Consequently, r is not matched in round k with probability $1 - 1/k$. This argument applies to all rounds $cn+1, \ldots, \ell$. Therefore,

$$\mathbf{Pr}\left[\xi(r,\ell)\right] \geq \prod_{k=cn+1}^{\ell} 1 - \frac{1}{k} = \prod_{k=cn+1}^{\ell} \frac{k-1}{k} = \frac{cn}{\ell}\,. \qquad ◀$$

B.2 Small Options

For δ-small options we use the LP-based algorithm from [27, Sec. 3.3] and analyze it within our algorithmic framework. In order to make this paper self-contained, we give a linear program for GAP (LP 1), the algorithm, and its corresponding proofs.

$$\begin{aligned}
\text{maximize} \quad & \sum_{\substack{i\in I_S \\ r\in R}} v_{i,r} x_{i,r} & & \\
\text{subject to} \quad & \sum_{i\in I_S} s_{i,r} x_{i,r} \leq W_r & & \forall r \in R \\
& \sum_{r\in R} x_{i,r} \leq 1 & & \forall i \in I_S \\
& x_{i,r} \in \{0,1\} & & \forall (i,r) \in I_S \times R
\end{aligned} \tag{LP 1}$$

Let $\mathcal{A}_S$ be Algorithm 5. After a sampling phase of dn rounds, in each round ℓ the algorithm computes an optimal solution $x^{(\ell)}$ of the relaxation of LP 1 for $I_S(\ell)$. Here, $I_S(\ell)$ denotes the instance of small options revealed so far. Now, the decision to which resource the current online item i is assigned, if at all, is made by randomized rounding using $x^{(\ell)}$: Resource $r \in R$ is chosen with probability $x^{(\ell)}_{i,r}$ and the item stays unassigned with probability $1 - \sum_{r\in R} x^{(\ell)}_{i,r}$. Note that it is only feasible to assign the item to the chosen resource if its remaining capacity is at least δW_r.

Algorithm 5 GAP algorithm for small options from [27, Sec. 3.3].

Input : Random order sequence of small options,
parameter $d \in (0,1)$.
Output : Integral GAP assignment.
Let ℓ be the current round and i be the online item of round ℓ.
if $1 \leq \ell \leq dn$ **then**
  Sampling phase – do not assign any item.
if $dn+1 \leq \ell \leq n$ **then**
  Let $x^{(\ell)}$ be an optimal fractional solution of LP 1 for $I_S(\ell)$.
  Choose a resource r (possibly none), where r has probability $x^{(\ell)}_{i,r}$.
  if *the remaining capacity of* r *is at least* δW_r **then**
    Assign i to r.

To analyze Algorithm 5, we consider the gain of profit in round $\ell \geq dn+1$, denoted by A_ℓ. For this purpose, let $i^{(\ell)}$ be the item of that round and $r^{(\ell)}$ the resource chosen by the algorithm. Now, it holds that $\mathbf{E}[A_\ell] = \mathbf{E}\left[v_{i^{(\ell)},r^{(\ell)}}\right] \mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)}\right]$, where in the first term, the expectation is over the item arriving in round ℓ and the resource chosen by the algorithm. The latter term only depends on the resource consumption of $r^{(\ell)}$ in earlier rounds. In the next two lemmas we give lower bounds for both terms.

▶ **Lemma B.1** ([27, Sec. 3.3]). *For any round $\ell \geq dn+1$, it holds that $\mathbf{E}\left[v_{i^{(\ell)},r^{(\ell)}}\right] \geq \frac{1}{n}\,\mathrm{OPT}_S$.*

Proof. The proof is similar to Lemma 6.2. As we consider a fixed round ℓ, we write i and r instead of $i^{(\ell)}$ and $r^{(\ell)}$ for ease of presentation. Further, we write $v(\alpha) := \sum_{j \in I_S} \sum_{s \in R} \alpha_{j,s} v_{j,s}$ for the profit of a fractional assignment α.

Fix any set Q of ℓ visible items in round ℓ. Let $x^{(n)}$ be an optimal (offline) solution to the relaxation of LP 1. Further, let $x^{(n)}|_Q$ denote the restriction of $x^{(n)}$ to the items in Q, i.e., $(x^{(n)}|_Q)_{j,s} = x^{(n)}_{j,s}$ if $j \in Q$ and $(x^{(n)}|_Q)_{j,s} = 0$ if $j \notin Q$. Since $x^{(n)}|_Q$ is a feasible and $x^{(\ell)}$ is an optimal solution for Q, we have $\mathbf{E}\left[v(x^{(\ell)})\right] \geq \mathbf{E}\left[v(x^{(n)}|_Q)\right]$. As in a random permutation each item has the same probability of ℓ/n to be in Q, it holds that

$$\mathbf{E}\left[v(x^{(\ell)})\right] \geq \mathbf{E}\left[v(x^{(n)}|_Q)\right] = \sum_{j \in I_S} \sum_{s \in R} \mathbf{Pr}[j \in Q]\, x^{(n)}_{j,s} v_{j,s} = \frac{\ell}{n} v(x^{(n)}) = \frac{\ell}{n}\,\mathrm{OPT}_S\,. \quad (6)$$

Similarly, each item from Q is the current online item i with probability $1/\ell$. The resource s, to which an item j gets assigned, is determined by randomized rounding using $x^{(\ell)}_{j,s}$. Therefore we get

$$\mathbf{E}[v_{i,r}] = \sum_{j \in Q} \sum_{s \in R} \mathbf{Pr}[j = i, s = r]\, v_{j,s} = \sum_{j \in Q} \sum_{s \in R} \frac{1}{\ell} x^{(\ell)}_{j,s} v_{j,s} = \frac{1}{\ell} v(x^{(\ell)})\,. \quad (7)$$

Combining (6) and (7) gives the claim. ◀

Hence, by the previous lemma the expected gain of profit in each round is a $(1/n)$-fraction of OPT_S, supposing the remaining resource capacity is large enough. The probability for the latter event is considered in the following lemma. Here, a crucial property is that we deal with δ-small options. Let $\Delta = \frac{1}{1-\delta}$.

▶ **Lemma B.2.** *For any round $\ell \geq dn+1$, we have $\mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)}\right] \geq \frac{c}{d}\left(1 - \Delta \ln \frac{\ell}{dn}\right)$.*

Proof. Let ξ be the event that no item is assigned to r after round dn. Note that ξ does not necessarily hold, since $\mathcal{A}_L$ might already have assigned items to r in earlier rounds. By Lemma 6.3, $\mathbf{Pr}[\xi] \geq \frac{c}{d}$. Therefore, it remains to show $\mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)} \mid \xi\right] \geq 1 - \Delta \ln \frac{\ell}{dn}$.

For this purpose, assume that ξ holds and let X denote the resource consumption of r after round $\ell - 1$. Further, let X_k be the resource consumption of r in round $k < \ell$. We have $X = \sum_{k=dn+1}^{\ell-1} X_k$. Let Q be the set of k visible items in round k. The set Q can be seen as uniformly drawn from all k-item subsets and any item $j \in Q$ is the current online item of round k with probability $1/k$. Now, the algorithm assigns any item j to resource r with probability $x_{j,r}^{(k)}$, thus

$$\mathbf{E}[X_k] = \sum_{j \in Q} \mathbf{Pr}[j \text{ occurs in round } k]\, s_{j,r} x_{j,r}^{(k)} = \frac{1}{k} \sum_{j \in Q} s_{j,r} x_{j,r}^{(k)} \leq \frac{W_r}{k}\,, \tag{8}$$

where the last inequality follows from the capacity constraint for resource r in LP 1. By linearity of expectation and inequality (8), the expected resource consumption up to round ℓ is thus

$$\mathbf{E}[X] = \sum_{k=dn+1}^{\ell-1} \mathbf{E}[X_k] \leq \sum_{k=dn+1}^{\ell-1} \frac{W_r}{k} \leq W_r \ln \frac{\ell}{dn}\,. \tag{9}$$

Now, since $i^{(\ell)}$ is δ-small, $X < (1-\delta)W_r$ implies $X + s_{i^{(\ell)},r^{(\ell)}} \leq W_r$ in which case the assignment is feasible. Using (9) and Markov's inequality, we obtain

$$\mathbf{Pr}[X < (1-\delta)W_r] = 1 - \mathbf{Pr}[X \geq (1-\delta)W_r] \geq 1 - \frac{\mathbf{E}[X]}{(1-\delta)W_r} \geq 1 - \Delta \ln \frac{\ell}{dn}\,. \qquad \blacktriangleleft$$

Now, the bound on the competitive ratio of $\mathcal{A}_S$ from Lemma 6.5 follows.

Proof of Lemma 6.5. We add the expected profits in single rounds using Lemmas B.1 and B.2.

$$\begin{aligned}
\mathbf{E}[\mathcal{A}_S] &= \sum_{\ell=dn+1}^{n} \mathbf{E}[A_\ell] = \sum_{\ell=dn+1}^{n} \mathbf{E}\left[v_{i^{(\ell)},r^{(\ell)}}\right] \mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)}\right] \\
&\geq \sum_{\ell=dn+1}^{n} \frac{1}{n} \operatorname{OPT}_S \frac{c}{d} \left(1 - \Delta \ln \frac{\ell}{dn}\right) = \frac{c}{dn} \left(\sum_{\ell=dn+1}^{n} 1 - \Delta \ln \frac{\ell}{dn}\right) \operatorname{OPT}_S \\
&= \frac{c}{dn} \left(n - dn - \Delta \sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn}\right) \operatorname{OPT}_S\,.
\end{aligned}$$

Since $\frac{\ell}{dn}$ is monotone increasing in ℓ, we have $\sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn} \leq \int_{dn+1}^{n+1} \ln \frac{\ell}{dn}\, \mathrm{d}\ell$ and this integral evaluates to $(n+1)\ln\frac{n+1}{dn+1} - (n+1) - (dn+1)\ln\frac{dn+1}{dn} + (dn+1)$. For $n \to \infty$, this approaches $n \ln \frac{1}{d} - n + dn$. Hence, we have $\lim_{n\to\infty} \mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left((1+\Delta)(1-d) - \Delta \ln \frac{1}{d}\right) \operatorname{OPT}_S$. $\blacktriangleleft$

Fast and Deterministic Approximations for k-Cut

Kent Quanrud
Department of Computer Science, University of Illinois at Urbana-Champaign, USA
http://www.kentquanrud.com
quanrud2@illinois.edu

Abstract

In an undirected graph, a k-cut is a set of edges whose removal breaks the graph into at least k connected components. The minimum weight k-cut can be computed in $n^{O(k)}$ time, but when k is treated as part of the input, computing the minimum weight k-cut is NP-Hard [18]. For poly(m, n, k)-time algorithms, the best possible approximation factor is essentially 2 under the small set expansion hypothesis [37]. Saran and Vazirani [46] showed that a $\left(2 - \frac{2}{k}\right)$-approximately minimum weight k-cut can be computed via $O(k)$ minimum cuts, which implies a $\tilde{O}(km)$ randomized running time via the nearly linear time randomized min-cut algorithm of Karger [27]. Nagamochi and Kamidoi [42] showed that a $\left(2 - \frac{2}{k}\right)$-approximately minimum weight k-cut can be computed deterministically in $O\left(mn + n^2 \log n\right)$ time. These results prompt two basic questions. The first concerns the role of randomization. Is there a deterministic algorithm for 2-approximate k-cuts matching the randomized running time of $\tilde{O}(km)$? The second question qualitatively compares minimum cut to 2-approximate minimum k-cut. Can 2-approximate k-cuts be computed as fast as the minimum cut – in $\tilde{O}(m)$ randomized time?

We give a deterministic approximation algorithm that computes $(2 + \epsilon)$-minimum k-cuts in $O\left(m \log^3 n/\epsilon^2\right)$ time, via a $(1 + \epsilon)$-approximation for an LP relaxation of k-cut.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Network optimization; Theory of computation → Linear programming; Theory of computation → Streaming, sublinear and near linear time algorithms; Theory of computation → Routing and network design problems

Keywords and phrases k-cut, multiplicative weight updates

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.23

Category APPROX

Related Version A full version of the paper is available at https://arxiv.org/abs/1807.07143.

Funding Work on this paper is partly supported by NSF grant CCF-1526799.

Acknowledgements The author thanks Chandra Chekuri for introducing him to the problem and providing helpful feedback, including pointers to the literature for rounding the LP. The author thanks Chao Xu for pointers to references. The author thanks the anonymous reviewers for their helpful comments.

1 Introduction

Let $G = (V, E)$ be an undirected graph with m edges and n vertices, with positive edge capacities given by $c : E \to \mathbb{R}_{>0}$. A *cut* is a set of edges $C \subseteq E$ whose removal leaves G disconnected. For $k \in \mathbb{N}$, a *k-cut* is a set of edges $C \subseteq E$ whose removal leaves G disconnected into at least k components. The capacity of a cut C is the sum capacity $\overline{c}(C) = \sum_{e \in C} c_e$ of edges in the cut. The *minimum k-cut* problem is to find a k-cut C of minimum capacity $\overline{c}(C)$.

The special case $k = 2$, which is to find the minimum cut, is particularly well-studied. The minimum cut can be computed in polynomial time by fixing a source s and computing the minimum s-t cut (via s-t max-flow) for all choices of t. Nagamochi and Ibaraki [39, 40, 41] and Hao and Orlin [22] improved the running time to $\tilde{O}(mn)$ which, at the time, was as fast


© Kent Quanrud;
licensed under Creative Commons License CC-BY

as computing a single maximum flow. A randomized edge contraction algorithm by Karger and Stein [28] finds the minimum cut with high probability in $\tilde{O}(n^2)$ time; this algorithm is now a staple of graduate level courses on randomized algorithms. Karger [27] gave a randomized algorithm based on the Tutte–Nash-Williams theorem [50, 44] that computes the minimum weight cut with high probability in $\tilde{O}(m)$ time. The best deterministic running time for minimum cut is currently $O(mn + n^2 \log n)$, by Stoer and Wagner [48]. Computing the minimum capacity cut deterministically in nearly linear time is a major open problem. Recently, Kawarabayashi and Thorup [30] made substantial progress on this problem with a deterministic nearly linear time algorithm for computing the minimum *cardinality* cut in an unweighted simple graph. This algorithm was simplified by Lo, Schmidt, and Thorup [35], and a faster algorithm was obtained by Henzinger, Rao, and Wang [24]. For capacitated graphs, a $(2+\epsilon)$-approximate minimum cut can be computed in $O\big((m \log n + n \log^2 n)/\epsilon\big)$ deterministic time by an algorithm of Matula [38] (as observed by Karger [26]).

The general case $k > 2$ is more peculiar. Goldschmidt and Hochbaum [18] showed that for any fixed k, finding the minimum k-cut is polynomial time solvable, but when k is part of the input, the problem is NP-Hard. The aforementioned randomized contraction algorithm of Karger and Stein [28] computes a minimum k-cut with high probability in $\tilde{O}(n^{2(k-1)})$ time. Thorup [49] gave a deterministic algorithm that also leverages the Tutte–Nash-Williams theorem [50, 44] and runs in $\tilde{O}(mn^{2k-2})$ time; this approach was recently refined to improve the running time to $\tilde{O}(mn^{2k-3})$ deterministic time [10] and $O(n^{(1.981+o(1))k})$ randomized time [21] (where the $o(1)$ goes to zero as k increases). There are slightly faster algorithms for particularly small values of k [34] and when the graph is unweighted [20]. As far as algorithms with running times that are polynomial in k are concerned, Saran and Vazirani [46] showed that a $\big(2 - \frac{2}{k}\big)$-approximate k-cut can be obtained by $O(k)$ minimum cut computations. By the aforementioned min-cut algorithms, this approach can be implemented in $\tilde{O}(km)$ randomized time, $\tilde{O}(kmn)$ time deterministically, and $\tilde{O}(km)$ time deterministically in unweighted graphs. Alternatively, Saran and Vazirani [46] showed that the same approximation factor can be obtained by computing a Gomory-Hu tree and taking the k lightest cuts. The Gomory-Hu tree can be computed in n maximum flow computations, and the maximum flow can be computed deterministically in $\tilde{O}\big(m \min\{m^{1/2}, n^{2/3}\} \log U\big)$ time for integer capacities between 1 and U [17]. This gives a $\tilde{O}\big(mn \min\{m^{1/2}, n^{2/3}\} \log U\big)$ deterministic time $(2 - 2/k)$-approximation for minimum k-cut, which is faster than $\tilde{O}(kmn)$ for sufficiently large k. (There are faster randomized algorithms for maximum flow [33, 36], but these still lead to slower randomized running times than $\tilde{O}(km)$ for k-cut.) An LP based 2-approximation was derived by Naor and Rabani [43], and a combinatorial 2-approximation was given by Ravi and Sinha [45] (see also [1]), but the running times are worse than those implied by Saran and Vazirani [46]. The best deterministic algorithm in the poly(m, n, k) regime is due to Nagamochi and Kamidoi [42], who compute $\big(2 - \frac{2}{k}\big)$-approximately minimum k-cuts in $O(mn + n^2 \log n)$ deterministic time. The constant factor of 2 is believed to be essentially the best possible. Manurangsi [37] showed that under the small set expansion hypothesis, for any fixed $\epsilon > 0$, one cannot compute a $(2-\epsilon)$-approximation for the minimum k-cut in poly(k, m, n) time unless $P = NP$.

The state of affairs for computing 2-approximate minimum k-cuts in poly(m, n, k)-time parallels the status of minimum cut. The fastest randomized algorithm is an order of magnitude faster than the fastest deterministic algorithm, while in the unweighted case the running times are essentially equal. A basic question is whether there exists a deterministic algorithm that computes a 2-approximation in $\tilde{O}(km)$ time, matching the randomized running time. As Saran and Vazirani's algorithm reduces k-cut to k minimum cuts, the gap

between the deterministic and randomized running times for k-cut is not only similar to, but a reflection of, the gap between the deterministic and randomized running times for minimum cut. An $\tilde{O}(m)$ deterministic algorithm for minimum cut would close the gap for 2-approximate k-cut as well.

A second question asks if computing a 2-approximate k-cut is qualitatively harder than computing the minimum cut. There is currently a large gap between the fastest algorithm for minimum cut and the fastest algorithm for 2-approximate minimum k-cut. Can one compute 2-approximate minimum k-cuts as fast as minimum cuts – in $\tilde{O}(m)$ randomized time? Removing the linear dependence on k would show that computing 2-approximate k-cuts is as easy as computing a minimum cut.

1.1 The main result

We make progress on both of these questions with a deterministic and nearly linear time $(2+\epsilon)$-approximation scheme for minimum k-cuts. To state the result formally, we first introduce an LP relaxation for the minimum k-cut due to Naor and Rabani [43].

$$\begin{aligned} \min \ & \sum_e c_e x_e \text{ over } x : E \to \mathbb{R} \\ \text{s.t.} \ & \sum_{e \in T} x_e \geq k-1 \text{ for all spanning trees } T, \\ & 0 \leq x_e \leq 1 \text{ for all edges } e. \end{aligned} \tag{L}$$

The feasible integral solutions of the LP (L) are precisely the k-cuts in G. The main contribution of this work is a nearly linear time approximation scheme for (L).

▶ **Theorem 1.** *In $O\big(m \log^3(n)/\epsilon^2\big)$ deterministic time, one can compute an $(1+\epsilon)$-multiplicative approximation to* (L)*.*

The integrality gap of (L) is known to be $(2-2/n)$ [43, 4]. Upon inspection, the rounding algorithm can be implemented in $O(m \log n)$ time, giving the following nearly linear time $(2+\epsilon)$-approximation scheme for k-cut.

▶ **Theorem 2.** *For sufficiently small $\epsilon > 0$, there is a deterministic algorithm that computes a k-cut with total capacity at most $(2+\epsilon)$ times the optimum value to* (L) *in $O\big(m \log^3(n)/\epsilon^2\big)$ time.*

The algorithm should be compared with the aforementioned algorithms of Saran and Vazirani [46], which computes a $\big(2-\frac{2}{k}\big)$-approximation to the minimum k-cut in $\tilde{O}(km)$ randomized time (with high probability), and of Nagamochi and Kamidoi [42], which computes a $\big(2-\frac{2}{k}\big)$-approximate minimum k-cut in $O\big(mn + n^2 \log n\big)$ deterministic time. At the cost of a $(1+\epsilon)$-multiplicative factor, we obtain a deterministic algorithm with nearly linear running time for *all values of k*. The approximation factor converges to 2, and we cannot expect to beat 2 under the small set expansion hypothesis [37]. Thus, Theorem 2 gives a tight, deterministic, and nearly linear time approximation scheme for k-cut. Theorem 2 leaves a little bit of room for improvement: the hope is for a deterministic algorithm that computes $(2-o(1))$-approximate minimum k-cuts in $\tilde{O}(m)$ time. Based on Theorem 2, we conjecture that such an algorithm exists.

1.2 Overview of the algorithm

We give a brief sketch of the algorithm, for the sake of informing subsequent discussion on related work in Section 1.3. A more complete description of the algorithm begins in earnest in Section 2.

The algorithm consists of a nearly linear time approximation scheme for the LP (L), and a nearly linear time rounding scheme. The approximation scheme for solving the LP extends techniques from recent work [6], applied to the dual of an *indirect reformulation* of (L). The rounding scheme is a simplification of the rounding scheme by Chekuri et al. [4] for the more general Steiner k-cut problem, which builds on the primal-dual framework of Goemans and Williamson [15].

The first step is to obtain a $(1+\epsilon)$-multiplicative approximation to the LP (L). Here a $(1+\epsilon)$-multiplicative approximation to the LP (L) is a feasible vector x of cost at most a $(1+\epsilon)$-multiplicative factor greater than the optimum value.

The LP (L) can be solved exactly by the ellipsoid method, with the separation oracle supplied by a minimum spanning tree (abbr. MST) computation, but the running time is a larger polynomial than desired. From the perspective of fast approximations, the LP (L) is difficult to handle because it is an exponentially large mixed packing and covering problem, with exponentially many covering constraints alongside upper bounds on each edge. Fast approximation algorithms for mixed packing and covering problems (e.g. [53, 8]) give bicriteria approximations that meet either the covering constraints or the packing constraints but not both. Even without consideration of the objective function, it is not known how to find feasible points to general mixed packing and covering problems in time faster than via exact LP solvers. Alternatively, one may consider the dual of (L), as follows. Let $\mathcal{T}$ denote the family of spanning trees in G.

$$\begin{aligned}
&\max\ (k-1)\sum_{T} y_T - \sum_{e\in E} z_e \text{ over } y:\mathcal{T}\to\mathbb{R} \text{ and } z: E\to\mathbb{R}\\
&\text{s.t. } \sum_{T\ni e} y_T \le c_e + z_e \text{ for all edges } e,\\
&\quad y_T \ge 0 \text{ for all spanning trees } T\in\mathcal{T},\\
&\quad z_e \ge 0 \text{ for all edges } e.
\end{aligned}$$

This program is not a positive linear program, and the edge potentials $z \in \mathbb{R}^E_{\ge 0}$ are difficult to handle by techniques such as [53, 6, 8]. It was not known, prior to this work, how to obtain any approximation to (L) (better than its integrality gap) with running time faster than the ellipsoid algorithm.

Critically, we consider the following larger LP instead of (L). Let $\mathcal{F}$ denote the family of all forests in G.

$$\begin{aligned}
&\min\ \sum_{e} c_e x_e \text{ over } x: E\to\mathbb{R}\\
&\text{s.t. } \sum_{e\in F} x_e \ge |F| + k - n \text{ for all forests } F\in\mathcal{F},\\
&\quad x_e \ge 0 \text{ for all edges } e\in E.
\end{aligned} \tag{C}$$

(C) is also an LP relaxation for k-cut. In fact, (C) is equivalent to (L), as one can verify directly (see Lemma 5 below). (C) is obtained from (L) by adding all the knapsack covering constraints [2], which makes the packing constraints ($x_e \le 1$ for each edge e) redundant.

Although the LP (C) adds exponentially many constraints to the original LP (L), (C) has the advantage of being a pure covering problem. Its dual is a pure packing problem, as follows.

$$\begin{aligned} \text{maximize} \quad & \sum_{F\in\mathcal{F}} (|F|+k-n)y_F \text{ over } y:\mathcal{F}\to\mathbb{R} \\ \text{s.t.} \quad & \sum_{F\ni e} y_F \le c_e \text{ for all edges } e\in E, \\ & y_F \ge 0 \text{ for all forests } F\in\mathcal{F}. \end{aligned} \tag{P}$$

The above LP packs forests into the capacitated graph G where the value of a forest F depends on the number of edges it contains, $|F|$. The objective value $|F|+k-n$ of a forest F is a lower bound on the number of edges F contributes to any k-cut. Clearly, we need only consider forests with at least $n-k+1$ edges.

To approximate the desired LP (C), we apply the MWU framework to the above LP (P), which generates $(1\pm\epsilon)$-multiplicative approximations to both (P) and its dual, (C). Implementing the MWU framework in nearly linear time is not immediate, despite precedent for similar problems. In the special case where $k=2$, the above LP (P) fractionally packs spanning trees into G. A nearly linear time approximation scheme for $k=2$ is given in previous work [6]. The general case with $k>2$ is more difficult for two reasons. First, the family of forests that we pack is larger than the family of spanning trees. Second, (P) is a weighted packing problem, where the coefficients in the objective depends on the number of edges in the forest. When $k=2$, we need only consider spanning trees with $n-1$ edges, so all the coefficients are 1 and the packing problem is unweighted. The heterogeneous coefficients in the objective create technical complications in the MWU framework, as the Lagrangian relaxation generated by the framework is no longer solved by a MST. In Section 2, we give an overview of the MWU framework and discuss the algorithmic complications in greater depth. In Section 3 and Section 4, we show how to extend the techniques of [6] with some new observations to overcome these challenges and approximate the LP (P) in nearly the same time as one can approximately pack spanning trees. Ultimately, we obtain the following deterministic algorithm for approximating the LP (P).

▶ **Theorem 3.** *In $O\big(m\log^3(n)/\epsilon^2\big)$ deterministic time, one can compute $(1\pm\epsilon)$-multiplicative approximations to* (P)*,* (C) *and* (L)*.*

The second step, after computing a fractional solution x to (L) with Theorem 3, is to round x to a discrete k-cut. The rounding step is essentially that of Chekuri et al. [4] for Steiner k-cuts. Their case is more general than ours; we simplify their rounding scheme, and pay greater attention to the running time. The rounding scheme is based on the elegant primal-dual MST algorithm of Goemans and Williamson [15].

▶ **Theorem 4.** *Given a feasible solution x to* (C)*, one can compute a k-cut C with cost at most $2(1-1/n)$ times the cost of x in $O(m\log n)$ time.*

Due to space constraints, Theorem 4 is deferred to Appendix A. Applying Theorem 4 to the output of Theorem 3 gives Theorem 2.

Computing the minimum k-cut via the LP (L) has additional benefits. First, computing a minimum k-cut with an approximation factor relative to the LP may be much stronger than the same approximation factor relative to the original problem, as LP's perform well in practice. Second, the solution to the LP gives a certificate of approximation ratio, as we can compare the rounded k-cut to the LP solution to infer an upper bound on the approximation ratio that may be smaller than 2.

Lastly, we note that some data structures can be simplified at the cost of randomization by using a randomized MWU framework instead [8]. These modifications are discussed at the end of Section 4.

1.3 Further related results and discussion

Fixed parameter tractability

The k-cut results reviewed above were focused on either exact polynomial-time algorithms for constant k or $(2-o(1))$-approximations in $\text{poly}(k,m,n)$ time, with a particular emphasis on the fastest algorithms in the $\text{poly}(k,m,n)$ time regime. There is also a body of literature concerning fixed parameter tractable algorithms for k-cut. Downey, Estivill-Castro, Fellows, Prieto-Rodriguez, and Rosamond [12] showed that k-cut is $W[1]$-hard in k even for simple unweighted graphs; $W[1]$-hardness implies that it is unlikely to obtain a running time of the form $f(k)\,\text{poly}(m,n)$ for any function f. On the other hand, Kawarabayashi and Thorup [29] showed that k-cut is fixed parameter tractable in the number of edges in the cut. More precisely, Kawarabayashi and Thorup [29] gave a deterministic algorithm that, for a given cardinality $s \in \mathbb{N}$, time, either finds a k-cut with at most s edges or reports that no such cut exists in $O\left(s^{s^{O(s)}}n^2\right)$ time. The running time was improved to $O\left(2^{O\left(s^2\log s\right)}n^4\log n\right)$ deterministic time and $\tilde{O}\left(2^{O(s)\log k}n^2\right)$ randomized time by Chitnis, Cygan, Hajiaghayi, Pilipczuk, and Pilipczuk [11].

Besides exact parameterized algorithms for k-cut, there is interest in approximation ratios between 1 and 2. Xiao, Cai, and Yao [51] showed that by adjusting the reduction of Saran and Vazirani [46] to use (exact) minimum ℓ-cuts – instead of minimum (2-)cuts, for any choice of $\ell \in \{2,\ldots,k-1\}$ – one can obtain $\left(2-\frac{\ell}{k}+O\left(\frac{\ell^2}{k^2}\right)\right)$-approximate minimum k-cuts in $n^{O(\ell)}$ time. For sufficiently small $\epsilon > 0$, by setting $\ell \approx \epsilon k$, this gives a $n^{O(\epsilon k)}$ time algorithm for $(2-\epsilon)$-approximate minimum cuts.

The hardness results of Downey, Estivill-Castro, Fellows, Prieto-Rodriguez, and Rosamond [12] and Manurangsi [37] do not rule out approximation algorithms with approximation ratio better than 2 and running times of the form $f(k)\,\text{poly}(m,n)$ (for any function f). Recently, Gupta, Lee, and Li [19] gave a FPT algorithm that, for a particular constant $c \in (0,1)$, computes a $(2-c)$-approximate k-cut in $2^{O\left(k^6\right)}\tilde{O}\left(n^4\right)$ time. This improves the running time of [51] for $\epsilon = c$ and k greater than some constant. Further improvements by Gupta et al. [20] achieved a deterministic 1.81 approximation in $2^{O\left(k^2\right)}n^{O(1)}$ time, and a randomized $(1+\epsilon)$-approximation (for any $\epsilon > 0$) in $(k/\epsilon)^{O(k)}n^{k+O(1)}$ time.

Knapsack covering constraints

We were surprised to discover that adding the knapsack covering constraints allowed for *faster* approximation algorithms. Knapsack covering constraints, proposed by Carr et al. [2] in the context of capacitated network design problems, generate a stronger LP whose solutions can be rounded to obtain better approximation factors [2, 31, 32, 3]. However, the larger LP can be much more complicated and is usually more difficult to solve. Recent work obtained a faster approximation scheme for approximating covering integer programs via knapsack covering constraints, but the dependency on ϵ is much worse, and the algorithm has a "weakly nearly linear" running time that suffers from a logarithmic dependency on the multiplicative range of input coefficients [9].

Fast approximations via LP's

Linear programs have long been used to obtain more accurate approximations to NP-Hard problems. Recently, we have explored the use of fast LP solvers to obtain *faster* approximations, including situations where polynomial time algorithms are known. In recent work [7], we used a linear time approximation to an LP relaxation for Metric TSP (obtained in [5]) to effectively sparsify the input and accelerate Christofides' algorithm. While nearly linear time approximations for complicated LP's are surprising in and of itself, perhaps the application to obtain faster approximations for combinatorial problems is more compelling. We think this result is an important data point for this approach.

2 Reviewing the MWU framework and identifying bottlenecks

In this section, let $\epsilon > 0$ be fixed. It suffices to assume that $\epsilon \geq 1/\operatorname{poly}(n)$, since below this point one can use the ellipsoid algorithm instead and still meet the desired running time. For ease of exposition, we seek only a $(1 + O(\epsilon))$-multiplicative approximation; a $(1 + \epsilon)$-multiplicative approximation with the same asymptotic running time follows by decreasing ϵ by a constant factor.

2.1 k-cuts as a (pure) covering problem

As discussed above, the first (and most decisive) step towards a fast, fractional approximation to k-cut is identifying the right LP. The standard LP (L) is difficult because it is a mixed packing and covering problem, and fast approximation algorithms for mixed packing and covering problems lead to bicriteria approximations that we do not know how to round. On the other hand, extending (L) with all the knapsack cover constraints makes the packing constraints $x_e \leq 1$ redundant (as shown below), leaving the pure covering problem, (C). The LPs (L) and (C) have essentially equivalent solutions in the following sense.

▶ **Lemma 5.** *Any feasible solution $x \in \mathbb{R}^E_{\geq 0}$ to (L) is a feasible solution to (C). For any feasible solution x to (C), the truncation $x' \in \mathbb{R}^n_{\geq 0}$ defined by $x'_e = \max\{x_e, 1\}$ is a feasible solution to both (L) and (C).*

Proof. Let x be a feasible solution to (L). We claim that x is feasible in (C). Indeed, let F be a forest, and extend F to a tree T. Then

$$\sum_{e \in F} x_e = \sum_{e \in T} x_e - \sum_{e \in T \setminus F} x_e \geq k - 1 - |T \setminus F| = k - 1 - (n - 1 - |F|) = |F| + k - n,$$

as desired.

Conversely, let $x \in \mathbb{R}^E_{\geq 0}$ be a feasible solution to (C), and let x' be the coordinatewise maximum of x and $\mathbb{1}$. Since $x' \leq \mathbb{1}$, and the covering constraints in (L) are a subset of the covering constraints in (C), if x' is feasible in (C) then it is also feasible in (L). To show that x' is feasible in (C), let F be a forest. Let $F' = \{e \in F : x_e > 1\}$ be the edges in F truncated by x' and let $F'' = F \setminus F'$ be the remaining edges. Since (a) $x'_e = 1$ for all $e \in F'$ and $x'_e = x_e$ for all $e \in F''$, and (b) x covers F'' in (C), we have

$$\sum_{e \in F} x'_e = \sum_{e \in F'} x'_e + \sum_{e \in F''} x'_e \overset{\text{(a)}}{=} |F'| + \sum_{e \in F''} x_e \overset{\text{(b)}}{\geq} |F'| + |F''| + k - n = |F| + k - n,$$

as desired. ◀

While having many more constraints than (L), (C) is a pure covering problem, for which finding a feasible point (faster than an exact LP solver) is at least plausible. The dual of (C) is the LP (P), which packs forests in the graph and weights each forest by the number of edges minus $(n-k)$. The coefficient of a forest in the objective of (P) can be interpreted as the number of edges that forest must contribute to any k-cut.

2.2 A brief sketch of width-independent MWU

We apply a *width-independent* version of the MWU framework to the packing LP (P), developed by Garg and Könemann [14] for multicommodity flow problems and generalized by Young [52]. We restrict ourselves to a sketch of the framework and refer to previous work for further details.

The width-independent MWU framework is a monotonic and iterative algorithm that starts with an empty solution $y = \mathbb{0}$ to the LP (P) and increases y along forests that solve certain Lagrangian relaxations to (P). Each Lagrangian relaxation is designed to steer y away from packing forests that have edges that are already tightly packed. For each edge e, the framework maintains a weight w_e that (approximately) exponentiates the load of edge e w/r/t the current forest packing y, as follows:

$$\ln(c_e w_e) \approx \frac{\log n}{\epsilon} \frac{\sum_{F \ni e} y_F}{c_e}. \tag{1}$$

The weight can be interpreted as follows. For an edge e, the value $\frac{\sum_{F \ni e} y_F}{c_e}$ is the amount of capacity used by the current packing y relative to the capacity of the edge e. We call $\frac{\sum_{F \ni e} y_F}{c_e}$ the (relative) *load* on edge e and is ≤ 1 if y is a feasible packing. The weight w_e is exponential in the load on the edge, where the exponential is amplified by the leading coefficient $\frac{\log n}{\epsilon}$. Initially, the empty solution $y = \emptyset$ induces zero load on any edge and we have $w_e = \frac{1}{c_e}$ for each edge e.

Each iteration, the framework solves the following Lagrangian relaxation of (P):

$$\text{maximize } \sum_{F \in \mathcal{F}} (|F| + k - n) z_F \text{ over } z : \mathcal{F} \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{e \in E} w_e \sum_{F \ni e} z_F \leq \sum_{e \in E} w_e c_e. \tag{R}$$

Given a $(1 + O(\epsilon))$-approximate solution z to the above, the framework adds δz to y for a carefully chosen value $\delta > 0$ (discussed in greater detail below). The next iteration encounters a different relaxation, where the edge weights w_e are increased to account for the loads increased by adding δz. Note that the edge weights w_e are monotonically increasing over the course of the algorithm.

At the end of the algorithm, standard analysis shows that the fractional forest packing y has objective value $(1 - O(\epsilon))\,\mathrm{OPT}$, and that $(1 - O(\epsilon))y$ satisfies all of the packing constraints. The error can be made one-sided by scaling y up or down. Moreover, it can be shown that at some point in the algorithm, an easily computable rescaling of w is a $(1 + O(\epsilon))$-relative approximation for the desired LP (C) (see for example [13, 14, 5]). Thus, although we may appear more interested in solving the dual LP (P), we are approximating the desired LP (C) as well.

The choice of δ differentiates this "width-independent" MWU from other MWU-type algorithms in the literature. The step size δ is chosen small enough that no weight increases by more than an $\exp(\epsilon)$-multiplicative factor, and large enough that some weight increases by (about) an $\exp(\epsilon)$-multiplicative factor. The analysis of the MWU framework reveals that $\langle w, c \rangle \leq n^{O(1/\epsilon)}$ at all times. In particular, each weight can increase by an $\exp(\epsilon)$-multiplicative factor at most $O\left(\frac{\ln n}{\epsilon^2}\right)$ times, so there are at most $O\left(\frac{n \ln n}{\epsilon^2}\right)$ iterations total.

2.3 Two bottlenecks

The MWU framework alternates between (a) solving the relaxation (R) induced by edge weights w_e and (b) updating the weights w_e for each edge in response to the solution to the relaxation. As the framework requires $O\left(\frac{m \log n}{\epsilon^2}\right)$ iterations, both parts must be implemented in polylogarithmic amortized time to reach the desired running time. A sublinear per-iteration running time seems unlikely by the following simple observations.

Consider first the complexity of simply expressing a solution. Any solution z to (R) is indexed by forests in G. A forest can have $\Omega(n)$ edges and requires $\Omega(n \log n)$ bits to specify. Writing down the index of just one forest in each of $O\left(\frac{m \log n}{\epsilon^2}\right)$ iterations takes $O\left(\frac{mn \log^2 n}{\epsilon^2}\right)$ time. The difficulty of even writing down a solution to (P) is not just a feature of the MWU framework. In general, there exists an optimal solution to (P) that is supported by at most m forests, as m is the rank of the implicit packing matrix. Writing down m forests also requires $\Omega(mn \log n)$ bits. Thus, either on a per-iteration basis in the MWU framework or w/r/t to the entire LP, the complexity of the output suggests a quadratic lower bound on the running time.

A second type of bottleneck arises from updating the weights. The weights w_e for each edge reflect the load induced by the packing y, per the formula (1). After computing a solution z to the relaxation (R), and updating $y \leftarrow y + \delta z$, we need to update the weights w_e to reflect the increased load from δz. In the worst case, δz packs into every edge, requiring us to update $O(m)$ individual weights. At the very least, δz should pack into the edges of at least one forest, and thus effect $\Omega(n)$ edges. Updating n edge weights in each iteration requires $O\left(\frac{mn \log n}{\epsilon^2}\right)$ time.

Even in hindsight, implementing either part – solving the relaxation or updating the weights – in isolation in sublinear time remains difficult. Our algorithm carefully plays both parts off each other, as co-routines, and amortizes against invariants revealed by the analysis of the MWU framework. The seemingly necessary dependence between parts is an important theme of this work and an ongoing theme from previous work [6, 5, 9].

3 Greedily finding forests to pack in $O\left(\log^2 n\right)$ amortized time

The MWU framework reduces (P) to a sequence of problems of the form (R). An important aspect of the Lagrangian approach is that satisfying the single packing constraint in (R) is much simpler than simultaneously satisfying all of the packing constraints in (P). With only 1 packing constraint, it suffices to (approximately) identify the best bang-for-buck forest F and taking as much as can fit in the packing constraint. The "bang-for-buck" ratio of a forest F is the ratio

$$\frac{|F| + k - n}{\sum_{e \in F} w_e},$$

where $|F|$ is the number of edges in F. Given a forest F (approximately) maximizing the above ratio, we set $z = \gamma e_F$ for γ as large as possible as fits in the single packing constraint. Note that, when $k = 2$, the optimal forest is the minimum weight spanning tree w/r/t w.

We first consider the simpler problem of maximizing the above ratio over forests F with exactly $|F| = \ell$ edges, for some $\ell > n - k$. Recall that the MST can be computed greedily by repeatedly adding the minimum weight edge that does not induce a cycle. Optimality of the greedy algorithm follows from the fact that spanning trees are the bases of a matroid called the *graphic matroid*. The forests of exactly ℓ edges are also the bases of a matroid; namely,

the restriction of the graphic matroid to forests of at most ℓ edges. In particular, the same greedy procedure computes the minimum weight forest of ℓ edges. Repeating the greedy algorithm for each choice of ℓ, one can solve (R) in $O(km \log n)$ time for each of $O\left(\frac{m \log n}{\epsilon^2}\right)$ iterations.

Stepping back, we want to compute the minimum weight forest with ℓ edges for a range of $k-1$ values of ℓ, and we can run the greedy algorithm for each choice of ℓ. We observe that the greedy algorithm is oblivious to the parameter ℓ, except for deciding when to stop. We can run the greedy algorithm *once* to build the MST, and then simulate the greedy algorithm for any value of ℓ by taking the first ℓ edges added to the MST.

▶ **Lemma 6.** *Let T be the minimum weight spanning tree w/r/t w. For any $\ell \in [n-1]$, the minimum weight forest w/r/t w with ℓ edges consists of the first ℓ minimum weight edges of T.*

Lemma 6 effectively reduces (R) to one MST computation, which takes $O(m \log n)$ time. Repeated over $O\left(\frac{m \log n}{\epsilon^2}\right)$ iterations, this leads to a $O\left(\frac{m^2 \log^2 n}{\epsilon^2}\right)$ running time. As observed previously [49, 6], the minimum weight spanning tree does not have to be rebuilt from scratch from one iteration to another, but rather adjusted dynamically as the weights change.

▶ **Lemma 7** (Holm, de Lichtenberg, and Thorup [25])**.** *In $O\left(\log^2 n\right)$ amortized time per increment to w, one can maintain the MST w/r/t w.*

The running time of Lemma 7 depends on the number of times the edge weights change, so we want to limit the number of weight updates exposed to Lemma 7. It is easy to see that solving (R) w/r/t a second set of weights $\tilde{w}$ that is a $(1 \pm \epsilon)$-multiplicative factor coordinate-wise approximation of w gives a solution that is a $(1 \pm \epsilon)$-multiplicative approximation to (R) w/r/t w. We maintain the MST w/r/t an approximation $\tilde{w}$ of w, and only propagate changes from w to $\tilde{w}$ when w is greater than $\tilde{w}$ by at least a $(1+\epsilon)$-multiplicative factor. As mentioned in Section 2, a weight w_e increases by a $(1+\epsilon)$-multiplicative factor at most $O\left(\frac{\log n}{\epsilon^2}\right)$ times. Applying Lemma 7 to the discretized weights $\tilde{w}$ and amortizing against the total growth of weights in the system gives us the following.

▶ **Lemma 8.** *In $O\left(\frac{m \log^3 n}{\epsilon^2}\right)$ total time, one can maintain the MST w/r/t a set of weights $\tilde{w}$ such that, for all $e \in E$, we have $\tilde{w}_e \in (1 \pm \epsilon) w_e$. Moreover, the MST makes at most $O\left(\frac{m \log n}{\epsilon^2}\right)$ edge updates total.*

Given such an MST T as above, and $\ell \in \{n-k+1, \ldots, n-1\}$ we need the ℓ minimum ($\tilde{w}$-)weight edges to form an (approximately) minimum weight forest F of ℓ edges. However, the data structure of Lemma 7 does not provide a list of edges in increasing order of weight. We maintain the edges in sorted order separately, where each time the dynamic MST replaces one edge with another, we make the same update in the sorted list. Clearly, such a list can be maintained in $O(\log n)$ time per update by dynamic trees. Our setting is simpler because the range of possible values of any weight $\tilde{w}_e$ is known in advance as follows. For $e \in E$, define

$$\mathcal{W}_e = \left\{ \frac{(1+\epsilon)^i}{c_e} : i \in \left\{ 0, 1, \ldots, O\left(\frac{\log n}{\epsilon^2}\right) \right\} \right\}.$$

Then $\tilde{w}_e \in \mathcal{W}_e$ for all $e \in E$ at all times. Define $\mathcal{L} = \{(e, \alpha) : \alpha \in \mathcal{W}_e\}$. The set $\mathcal{L}$ represents the set of all possible assignments of weights to edges that may arise. As mentioned above, $|\mathcal{L}| = O\left(\frac{m \log n}{\epsilon^2}\right)$. Let B be a balanced binary tree over $\mathcal{L}$, where $\mathcal{L}$ is sorted by increasing

order of the second coordinate $\tilde{w}_e$ (and ties are broken arbitrarily). The tree B has height $\log|\mathcal{L}| = O(\log m)$ and can be built in $O(|\mathcal{L}|) = O\left(\frac{m \log n}{\epsilon^2}\right)$ time[1].

We mark the leaves based on the edges in T. Every time the MST T adds an edge e of weight $\tilde{w}_e$, we mark the corresponding leaf $(e, \tilde{w}_e)$ as marked. When an edge e is deleted, we unmark the corresponding leaf. Lastly, when an edge $e \in T$ has its weight increased from α to α', we unmark the leaf (e, α) and mark the leaf (e, α'). Note that only edges in T have their weight changed.

For each subtree of T, we track aggregate information and maintain data structures over the set of all marked leaves in the subtree. For a node b in B, let $\mathcal{L}_b$ be the set of marked leaves in the subtree rooted at b. For each $b \in B$ we maintain two quantities: (a) the number of leaves marked in the subtree rooted at b, $|\mathcal{L}_b|$; and (b) the sum of edges weights of leaves marked in the subtree rooted at b, $W_b = \sum_{(e,\alpha)\in\mathcal{L}_b} \alpha$. Since the height of B is $O(\log n)$, both of these quantities can be maintained in $O(\log n)$ time per weight update.

▶ **Lemma 9.** *In $O\big(m \log(n)/\epsilon^2\big)$ time initially and $O(\log n)$ time per weight update, one can maintain a data structure that, given $\ell \in [n-1]$, returns in $O(\log n)$ time (a) the ℓ minimum weight edges of the MST (implicitly), and (b) the total weight of the first ℓ edges of the MST.*

With Lemma 7 and Lemma 9, we can now compute, for any $\ell \in [n-1]$, a $(1+\epsilon)$-approximation to the minimum weight forest of ℓ edges, along with the sum weight of the forest, both in logarithmic time. To find the best forest, then, we need only query the data structure for each of the $k-1$ integer values from $n-k+1$ to $n-1$. That is, excluding the time to maintain the data structures, we can now solve the relaxation (R) in $O(k \log n)$ time per iteration.

At this point, we still require $O(km \log n)$ time just to solve the Lagrangian relaxations (R) generated by the MWU framework. (There are other bottlenecks, such as updating the weights at each iteration, that we have not yet addressed.) To remove the factor of k in solving (R), we require one final observation.

▶ **Lemma 10.** *The minimum ratio subforest of T can be found by binary search.*

Proof. Enumerate the MST edges $e_1, \ldots, e_{n-1} \in T$ in increasing order of weight. For ease of notation, we denote $\tilde{w}_i = \tilde{w}_{e_i}$ for $i \in [n-1]$. We define a function $f : [k-1] \to \mathbb{R}_{>0}$ by

$$f(i) = \frac{i}{\sum_{j=1}^{n-k+i} \tilde{w}_j}.$$

For each i, $f(i)$ is the ratio achieved by the first $n-k+i$ edges of the MST. Our goal is to maximize $f(i)$ over $i \in [k-1]$. For any $i \in [k-2]$, we have

$$\begin{aligned} f(i+1) - f(i) &= \frac{i+1}{\sum_{j=1}^{n-k+i+1} \tilde{w}_j} - \frac{i}{\sum_{j=1}^{n-k+i} \tilde{w}_j} \\ &= \frac{(i+1)\sum_{j=1}^{n-k+i} \tilde{w}_j - i\sum_{j=1}^{n-k+i+1} \tilde{w}_j}{\left(\sum_{j=1}^{n-k+i+1} \tilde{w}_j\right)\left(\sum_{j=1}^{n-k+i} \tilde{w}_j\right)} = \frac{\sum_{j=1}^{n-k+i} \tilde{w}_j - i\tilde{w}_{n-k+i+1}}{\left(\sum_{j=1}^{n-k+i+1} \tilde{w}_j\right)\left(\sum_{j=1}^{n-k+i} \tilde{w}_j\right)}. \end{aligned}$$

Since the denominator $\left(\sum_{j=1}^{n-k+i+1} \tilde{w}_j\right)\left(\sum_{j=1}^{n-k+i} \tilde{w}_j\right)$ is positive, we have $f(i+1) \le f(i) \iff i\tilde{w}_{n-k+i+1} \ge \sum_{j=1}^{n-k+i} \tilde{w}_j$. If $i\tilde{w}_{n-k+i+1} \ge \sum_{j=1}^{n-k+i} \tilde{w}_j$ for all $i \in [k-2]$, then $f(1) < f(2) <$

[1] or even less time if we build B lazily, but constructing B is not a bottleneck

$\cdots < f(k-1)$, so $f(i)$ is maximized by $i = k-1$. Otherwise, let $i_0 \in [k-2]$ be the first value of i such that $f(i+1) \leq f(i)$. For $i_1 \geq i_0$, as (c) $\tilde{w}_{e_i}$ is increasing in i, (d) $f(i_0+1) \leq f(i_0)$, we have

$$\sum_{j=1}^{n-k+i_1} \tilde{w}_j - i_1 w_{n-k+i_1+1} = \sum_{j=1}^{n-k+i_0} \tilde{w}_j - i_0 \tilde{w}_{n-k+i_1+1} + \sum_{j=n-k+i_0+1}^{n-k+i_1} \tilde{w}_j - (i_1 - i_0)\tilde{w}_{n-k+i_1+1}$$
$$\overset{\text{(c)}}{\leq} \sum_{j=1}^{n-k+i_0} \tilde{w}_j - i_0 \tilde{w}_{n-k+i_0+1} \overset{\text{(d)}}{\leq} 0.$$

That is, $f(i_1+1) \leq f(i_1)$ for all $i_1 \geq i_0$. Thus f consists of one increasing subsequence followed by a decreasing subsequence, and its global maximum is the unique local maximum. ◀

By Lemma 10, the choice of ℓ can be found by calculating the ratio of $O(\log k)$ candidate forests. By Lemma 9, the ratio of a candidate forest can be computed in $O(\log n)$ time.

▶ **Lemma 11.** *Given the data structure of Lemma 9, one can compute a $(1+O(\epsilon))$-multiplicative approximation to* (R) *in $O(\log n \log k)$ time.*

The polylogarithmic running time in Lemma 11 is surprising when considering that solutions to (R) should require at least a linear number of bits, as discussed earlier in Section 2.3. In hindsight, a combination of additional structure provided by the MWU framework and the LP (P) allows us to apply data structures that effectively compress the forests and output each forest in polylogarithmic amortized time. Implicit compression of this sort also appears in previous work [6, 5, 9].

4 Packing greedy forests in $O\left(\log^2 n\right)$ amortized time

In Section 3, we showed how to solve (R) in polylogarithmic time per iteration. In this section, we address the second main bottleneck: updating the weights w after increasing y to $y + \delta z$ per the formula (1), where z is an approximate solution to the relaxation (R) and $\delta > 0$ is the largest possible value such that no weight increases by more than a $(1+\epsilon)$-multiplicative factor. As discussed in Section 2.3, this may be hard to do in polylogarithmic time when many of the edges $e \in E$ require updating.

A sublinear time weight update must depend heavily on the structure of the solutions generated to (R). In our case, each solution z to a relaxation (R) is of the form γe_F, where e_F is the indicator vector of a forest F and $\gamma > 0$ is a scalar as large as possible subject to the packing constraint in (R). We need to update the weights to reflect the loads induced by $\delta z = \delta\gamma e_F$, where δ is chosen large as possible so that no weight increases by more than an $\exp(\epsilon)$-multiplicative factor. With this choice of δ, the weight update simplifies to the following formula. Let w denote the set of weights before the updates and w' denote the set of weights after the updates. For a solution $z = \gamma e_F$, we have

$$w'_e = \begin{cases} w_e & \text{if } e \notin F, \\ \exp\left(\frac{\epsilon \min_{f \in F} c_f}{c_e}\right) & \text{if } e \in F. \end{cases} \tag{2}$$

The weight update formula above can be interpreted as follows. Because our solution is supported along a single forest F, the only edges whose loads are effected are those in the forest F. As load is relative to the capacity of an edge e, the increase of the logarithm the weight w_e of an edge $e \in F$ is inversely proportional to its capacity. By choice of δ, the

minimum capacity edge $\arg\min_{f \in F} c_f$ has its weight increased by an $\exp(\epsilon)$ multiplicative factor. The remaining edges with larger capacity each have the logarithm of their weight increased in proportion to the ratio of the bottleneck capacity to its own capacity.

Simplifying the weight update formula does not address the basic problem of updating the weights of every edge in a forest F, *without visiting every edge in* F. Here we require substantially more structure as to how the edges in F are selected. We observe that although there may be $\Omega(n)$ edges in F, we can always decompose F into a logarithmic number of "canonical subforests", as follows.

▶ **Lemma 12.** *One can maintain, in $O(\log n)$ time per update to the MST T, a collection of subforests $\mathcal{C}_T \subseteq \mathcal{F}$ such that:*

- **(i)** $|\mathcal{C}_T| = O(n \log n)$.
- **(ii)** *Each edge $e \in T$ is contained in $O(\log n)$ forests.*
- **(iii)** *For each $\ell \in [n-1]$, the forest F consisting of the ℓ minimum weight edges in T decomposes uniquely into the disjoint union of $O(\log n)$ forests in $\mathcal{C}_T$. The decomposition can be computed in $O(\log n)$ time.*

In fact, the collection of subforests is already maintained implicitly in Lemma 9. Recall, from Section 3, the balanced binary tree B over the leaf set $\mathcal{L}$, which consists of all possible discretized weight-to-edge assignments and is ordered in increasing order of weight. Leaves are marked according to the edges in the MST T, and each node is identified with the forest consisting of all marked leaves in the subtree rooted at the node. For each $\ell \in [n-1]$, the forest F_ℓ induced by the ℓ minimum weight edges in T is the set of marked leaves over an interval of $\mathcal{L}$. The interval decomposes into the disjoint union of leaves of $O(\log n)$ subtrees, which corresponds to decomposing F_ℓ into the disjoint union of marked leaves of $O(\log n)$ subtrees of B. That is, the forests of marked leaves induced by subtrees of B gives the "canonical forests" $\mathcal{C}_T$ that we seek.

The following technique of decomposing weight updates is critical to previous work [6, 5, 9]; we briefly discuss the high-level ideas and refer to previous work for complete details.

Decomposing the solution into a small number of known static sets is important because weight updates can be simulated over a *fixed set* efficiently. The data structure `lazy-inc`, defined in [6] and inspired by techniques by Young [53], simulates a weight update over a fixed set of weights in such a way that the time can be amortized against the logarithm of the increase in each of the weights. As discussed above, the total logarithmic increase in each of the weights is bounded from above. The data structure `lazy-inc` is dynamic, allowing insertion and deletion into the underlying set, in $O(\log n)$ time per insertion or deletion [5].

We define an instance of `lazy-inc` at each node in the balanced binary tree B. Whenever a leaf is marked as occupied, the corresponding edge is inserted into each of $O(\log n)$ instances of `lazy-inc` at the ancestors of the leaf; when a leaf is marked as unoccupied, it is removed from each of these instances as well. Each instance of `lazy-inc` can then simulate a weight update over the marked leaves at its nodes in $O(1)$ constant time per instance, plus a total $O(\log n)$ amortized time. More precisely, the additional time is amortized against the sum of increases in the logarithms of the weights, which (as discussed earlier) is bounded above by $O\big(m \log(n)/\epsilon^2\big)$.

We also track, for each canonical forest, the minimum capacity of any edge in the forest. The minimum capacity ultimately controls the rate at which all the other edges increase, per (2).

Given a forest F induced by the ℓ minimum weight edges of T, we decompose F into the disjoint union of $O(\log n)$ canonical subforests of T. For each subforest we have precomputed the minimum capacity, and an instance of `lazy-inc` that simulates weight updates on all

edges in the subforest. The minimum capacity over edges in F determines the rate of increase, and the increase is made to each instance of `lazy-inc` in $O(1)$ time per instance plus $O(\log n)$ amortized time over all instances.

▶ **Lemma 13.** *Given a forest F generated by Lemma 11, one can update the edge weights per* (2) *in $O\big(\log^2 n\big)$ amortized time per iteration.*

Note that Lemma 13 holds only for the forests output by Lemma 11. We can not decompose other forests in G, or even other subforests of T, into the disjoint union $O(\log n)$ subforests. Lemma 12 holds specifically for the forests induced by the ℓ minimum weight edges of T, for varying values of ℓ. This limitation highlights the importance of coupling the oracle and the weight update: the running time in Lemma 11 for solving (R) is amortized against the growth of the weights, and the weight updates in Lemma 13 leverage the specific structure by which solutions to (R) are generated.

We note that the `lazy-inc` data structures can be replaced by random sampling in the randomized MWU framework [8]. Here one still requires the decomposition into canonical subforests; an efficient threshold-based sampling is then conducted at each subforest.

5 Putting things together

In this section, we summarize the main points of the algorithm and account for the running time claimed in Theorem 3.

Proof of Theorem 3. By standard analysis (e.g., [6, Theorem 2.1]), the MWU framework returns a $(1 - O(\epsilon))$-multiplicative approximation to the packing LP (P) as long as we can approximate the relaxation (R) to within a $(1 - O(\epsilon))$-multiplicative factor. This slack allows us to maintain the weight w_e to within a $(1 \pm O(\epsilon))$-multiplicative factor of the "true weights" given (up to a leading constant) by (1). In particular, we only propagate a change to w_e when it has increased by a $(1 + \epsilon)$-multiplicative factor. Each weight w_e is monotonically increasing and its growth is bounded by a $m^{O\left(\frac{1}{\epsilon}\right)}$-multiplicative factor, so each weight w_e increases by a $(1 + \epsilon)$-multiplicative factor $O\left(\frac{\log m}{\epsilon^2}\right)$ times.

By Lemma 11, each instance of (R) can be solved in $O\big(\log^2 n\big)$ amortized time. Here the running time is amortized against the number of weight updates, as the solution can be updated dynamically in $O\big(\log^2 n\big)$ amortized time. By Lemma 13, the weight update w/r/t a solution generated by Lemma 11 can be implemented in $O\big(\log^2 n\big)$ amortized time. Here again the running time is amortized against the growth of the edge weights. Since there are $O\left(\frac{m \log n}{\epsilon^2}\right)$ total edge updates, this gives a total running time of $O\left(\frac{m \log^3 n}{\epsilon^2}\right)$. ◀

References

1 Francisco Barahona. On the k-cut problem. *Oper. Res. Lett.*, 26(3):99–105, 2000.

2 Robert D. Carr, Lisa Fleischer, Vitus J. Leung, and Cynthia A. Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, January 9-11, 2000, San Francisco, CA, USA.*, pages 106–115. ACM/SIAM, 2000.

3 Deeparnab Chakrabarty, Chandra Chekuri, Sanjeev Khanna, and Nitish Korula. Approximability of Capacitated Network Design. *Algorithmica*, 72(2):493–514, 2015. Preliminary version in IPCO 2011.

4 Chandra Chekuri, Sudipto Guha, and Joseph Naor. The Steiner k-Cut Problem. *SIAM J. Discrete Math.*, 20(1):261–271, 2006. Preliminary version in ICALP 2003.

5 Chandra Chekuri and Kent Quanrud. Approximating the Held-Karp Bound for Metric TSP in Nearly-Linear Time. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 789–800. IEEE Computer Society, 2017.

6 Chandra Chekuri and Kent Quanrud. Near-Linear Time Approximation Schemes for some Implicit Fractional Packing Problems. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 801–820. SIAM, 2017.

7 Chandra Chekuri and Kent Quanrud. Fast Approximations for Metric-TSP via Linear Programming. *CoRR*, abs/1802.01242, 2018. `arXiv:1802.01242`.

8 Chandra Chekuri and Kent Quanrud. Randomized MWU for Positive LPs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 358–377. SIAM, 2018.

9 Chandra Chekuri and Kent Quanrud. On Approximating (Sparse) Covering Integer Programs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1596–1615, 2019.

10 Chandra Chekuri, Kent Quanrud, and Chao Xu. LP relaxation and tree packing for minimum k-cuts. In *2nd Symposium on Simplicity in Algorithms, SOSA@SODA 2019, January 8-9, 2019 - San Diego, CA, USA*, pages 7:1–7:18, 2019.

11 Rajesh Chitnis, Marek Cygan, MohammadTaghi Hajiaghayi, Marcin Pilipczuk, and Michal Pilipczuk. Designing FPT Algorithms for Cut Problems Using Randomized Contractions. *SIAM J. Comput.*, 45(4):1171–1229, 2016. Preliminary version in FOCS 2012.

12 Rodney G. Downey, Vladimir Estivill-Castro, Michael R. Fellows, Elena Prieto-Rodriguez, and Frances A. Rosamond. Cutting Up is Hard to Do: the Parameterized Complexity of k-Cut and Related Problems. *Electr. Notes Theor. Comput. Sci.*, 78:209–222, 2003.

13 Naveen Garg and Jochen Könemann. Faster and Simpler Algorithms for Multicommodity Flow and Other Fractional Packing Problems. In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 300–309. IEEE Computer Society, 1998.

14 Naveen Garg and Jochen Könemann. Faster and Simpler Algorithms for Multicommodity Flow and Other Fractional Packing Problems. *SIAM J. Comput.*, 37(2):630–652, 2007. Preliminary version in FOCS 1998.

15 Michel X. Goemans and David P. Williamson. A General Approximation Technique for Constrained Forest Problems. *SIAM J. Comput.*, 24(2):296–317, 1995. Preliminary version in SODA 1992.

16 Michel X. Goemans and David P. Williamson. The primal-dual method for approximation algorithms and its applications to network design problems. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 144–191. PWS Publishing Company, Boston, MA, July 1996.

17 Andrew V. Goldberg and Satish Rao. Beyond the Flow Decomposition Barrier. *J. ACM*, 45(5):783–797, 1998. Preliminary version in FOCS 1997.

18 Olivier Goldschmidt and Dorit S. Hochbaum. A Polynomial Algorithm for the k-cut Problem for Fixed k. *Math. Oper. Res.*, 19(1):24–37, 1994. Preliminary version in FOCS 1988.

19 Anupam Gupta, Euiwoong Lee, and Jason Li. An FPT Algorithm Beating 2-Approximation for k-Cut. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2821–2837. SIAM, 2018.

20 Anupam Gupta, Euiwoong Lee, and Jason Li. Faster Exact and Approximate Algorithms for k-Cut. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*. IEEE Computer Society, 2018.

21 Anupam Gupta, Euiwoong Lee, and Jason Li. The number of minimum k-cuts: improving the Karger-Stein bound. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019.*, pages 229–240, 2019.

22 Jianxiu Hao and James B. Orlin. A Faster Algorithm for Finding the Minimum Cut in a Directed Graph. *J. Algorithms*, 17(3):424–446, 1994. Preliminary version in SODA 1992.

23 Dov Harel and Robert Endre Tarjan. Fast Algorithms for Finding Nearest Common Ancestors. *SIAM J. Comput.*, 13(2):338–355, 1984.

24 Monika Henzinger, Satish Rao, and Di Wang. Local Flow Partitioning for Faster Edge Connectivity. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1919–1938. SIAM, 2017.

25 Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *J. ACM*, 48(4):723–760, 2001. Preliminary version in STOC 1998.

26 David R. Karger. *Random Sampling in Graph Optimization Problems*. PhD thesis, Stanford University, Stanford, CA 94305, 1994.

27 David R. Karger. Minimum cuts in near-linear time. *J. ACM*, 47(1):46–76, 2000. Preliminary version in STOC 1996.

28 David R. Karger and Clifford Stein. A New Approach to the Minimum Cut Problem. *J. ACM*, 43(4):601–640, 1996. Preliminary version in STOC 1993.

29 Ken-ichi Kawarabayashi and Mikkel Thorup. The Minimum k-way Cut of Bounded Size is Fixed-Parameter Tractable. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 160–169. IEEE Computer Society, 2011.

30 Ken-ichi Kawarabayashi and Mikkel Thorup. Deterministic Global Minimum Cut of a Simple Graph in Near-Linear Time. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 665–674. ACM, 2015.

31 Stavros G. Kolliopoulos and Neal E. Young. Tight Approximation Results for General Covering Integer Programs. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 522–528. IEEE Computer Society, 2001.

32 Stavros G. Kolliopoulos and Neal E. Young. Approximation algorithms for covering/packing integer programs. *J. Comput. Syst. Sci.*, 71(4):495–505, 2005. Preliminary version in FOCS 2001.

33 Yin Tat Lee and Aaron Sidford. Path Finding Methods for Linear Programming: Solving Linear Programs in $\tilde{O}(\sqrt{\text{rank}})$ Iterations and Faster Algorithms for Maximum Flow. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 424–433. IEEE Computer Society, 2014.

34 Matthew S. Levine. Fast randomized algorithms for computing minimum $\{3, 4, 5, 6\}$-way cuts. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, January 9-11, 2000, San Francisco, CA, USA.*, pages 735–742, 2000.

35 On-Hei Solomon Lo, Jens M. Schmidt, and Mikkel Thorup. Contraction-Based Sparsification in Near-Linear Time. *CoRR*, abs/1810.03865, 2018. `arXiv:1810.03865`.

36 Aleksander Madry. Computing Maximum Flow with Augmenting Electrical Flows. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 593–602. IEEE Computer Society, 2016.

37 Pasin Manurangsi. Inapproximability of Maximum Edge Biclique, Maximum Balanced Biclique and Minimum k-Cut from the Small Set Expansion Hypothesis. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 79:1–79:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

38 David W. Matula. A Linear Time $2 + \epsilon$ Approximation Algorithm for Edge Connectivity. In *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA.*, pages 500–504, 1993.

39 Hiroshi Nagamochi and Toshihide Ibaraki. Computing Edge-Connectivity in Multiple and Capacitated Graphs. In *Algorithms, International Symposium SIGAL '90, Tokyo, Japan, August 16-18, 1990, Proceedings*, volume 450 of *Lecture Notes in Computer Science*, pages 12–20. Springer, 1990.

40 Hiroshi Nagamochi and Toshihide Ibaraki. A Linear-Time Algorithm for Finding a Sparse k-Connected Spanning Subgraph of a k-Connected Graph. *Algorithmica*, 7(5&6):583–596, 1992.

41 Hiroshi Nagamochi and Toshihide Ibaraki. Computing Edge-Connectivity in Multigraphs and Capacitated Graphs. *SIAM J. Discrete Math.*, 5(1):54–66, 1992.

42 Hiroshi Nagamochi and Yoko Kamidoi. Minimum cost subpartitions in graphs. *Inf. Process. Lett.*, 102(2-3):79–84, 2007.

43 Joseph Naor and Yuval Rabani. Tree packing and approximating k-cuts. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA.*, pages 26–27. ACM/SIAM, 2001.

44 C. St. J. A. Nash-Williams. Edge-disjoint spanning trees of finite graphs. *J. London Math. Soc.*, 36:445–450, 1961.

45 R. Ravi and Amitabh Sinha. Approximating k-cuts using network strengths as a Lagrangean relaxation. *European Journal of Operational Research*, 186(1):77–90, 2008. Preliminary version in SODA 2002.

46 Huzur Saran and Vijay V. Vazirani. Finding k Cuts within Twice the Optimal. *SIAM J. Comput.*, 24(1):101–108, 1995. Preliminary version in FOCS 1991.

47 Daniel Dominic Sleator and Robert Endre Tarjan. A Data Structure for Dynamic Trees. *J. Comput. Syst. Sci.*, 26(3):362–391, 1983. Preliminary version in STOC 1981.

48 Mechthild Stoer and Frank Wagner. A simple min-cut algorithm. *J. ACM*, 44(4):585–591, 1997. Preliminary version in ESA 1994.

49 Mikkel Thorup. Minimum k-way cuts via deterministic greedy tree packing. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 159–166. ACM, 2008.

50 W. T. Tutte. On the problem of decomposing a graph into n connected components. *J. London Math. Soc.*, 36:221–230, 1961.

51 Mingyu Xiao, Leizhen Cai, and Andrew Chi-Chih Yao. Tight Approximation Ratio of a General Greedy Splitting Algorithm for the Minimum k-Way Cut Problem. *Algorithmica*, 59(4):510–520, 2011.

52 Neal E. Young. Sequential and Parallel Algorithms for Mixed Packing and Covering. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 538–546. IEEE Computer Society, 2001.

53 Neal E. Young. Nearly Linear-Time Approximation Schemes for Mixed Packing/Covering and Facility-Location Linear Programs. *CoRR*, abs/1407.3015, 2014. `arXiv:1407.3015`.

A Rounding fractional forest packings to k-cuts

In this section, we show how to round a fractional solution x to (L) a k-cut of cost at most twice the cost of x. The rounding scheme is due to Chekuri et al. [4] for the more general problem of Steiner k-cuts. The rounding scheme extends the primal-dual framework of Goemans and Williamson [15, 16]. In hindsight, we realized the primal-dual framework is only required for the analysis, and that the algorithm itself is very simple.

We first give a conceptual description of the algorithm, called `greedy-cuts`. The conceptual description suffices for the sake of analyzing the approximation guarantee. Later, we give implementation details and demonstrate that it can be executed in $O(m \log n)$ time.

To describe the algorithm, we first introduce the following definitions.

▶ **Definition 14.** *Let F be a minimum weight spanning forest in a weighted, undirected graph, and order the edges of F in increasing order of weight (breaking ties arbitrarily). A* greedy component *of F is a connected component induced by a prefix of F. A* greedy cut *is a cut induced by a greedy component of F.*

Algorithm 1 A conceptual sketch of a deterministic rounding algorithm for k-cut.

`greedy-cuts`$(G = (V, E), c, x)$

```
// conceptual sketch
1. let E' = {e ∈ E : x_e ≥ (n−1)/(2n)},  E ← E \ E'
2. if E' is a k-cut then return E'
3. let F be a minimum weight spanning forest in E w/r/t x w/ ℓ components
4. return the union of E' and the k − ℓ minimum weight greedy cuts of F
```

The rounding algorithm is conceptually very simple and a pseudocode sketch is given in Algorithm 1. We first take all the edges with $x_e > 1/2 + o(1)$. If this is already a k-cut, then it is a 2-approximation because the corresponding indicator vector is $\leq (2 - o(1))x$. Otherwise, we compute the minimum spanning forest F in the remaining graph, where the weight of an edge is given by x. Letting ℓ be number of components of F, we compute the $k - \ell$ minimum weight greedy cuts w/r/t F. We output the union of E' and the $k - \ell$ greedy cuts.

Chekuri, Guha, and Naor [4] implicitly showed that this algorithm has an approximation factor of $2(1 - 1/n)$. Their analysis is for the more general Steiner k-cut problem, where we are given a set of terminal vertices T, and want to find the minimum weight set of edges whose removal divides the graph into at least k components each containing a terminal vertex $t \in T$. The algorithm and analysis is based on the primal-dual framework of Goemans and Williamson [15, 16]. For the minimum weight Steiner tree problem, the primal-dual framework returns a Steiner tree and a feasible fractional cut packing in the dual LP. The cost of the Steiner cut packing is within a $2(1 - o(1))$-multiplicative factor of the corresponding Steiner tree. Via LP duality, the Steiner tree and the cut packing mutually certify an approximation ratio of $2(1 - o(1))$. The cut packing certificate has other nice properties, and Chekuri, Guha, and Naor [4] show that the $k - 1$ minimum cuts in the support of the fractional cut packing give a 2-approximate Steiner k-cut.

For the (non-Steiner) k-cut problem, we want minimum cuts in the support of the fractional cut packing returned by the primal-dual framework applied to minimum spanning forests. To shorten the algorithm, we observe that (a) the primal-dual framework returns the minimum spanning forest, and (b) the cuts supported by the corresponding dual certificate are precisely the greedy cuts of the minimum spanning forest. Thus `greedy-cuts` essentially refactors the algorithm analyzed by Chekuri et al. [4].

▶ **Lemma 15** ([4])**.** `greedy-cuts` *returns a k-cut of weight at most $2\big(1 - \frac{1}{n}\big)\langle c, x\rangle$.*

The connection to Chekuri et al. [4] is not explicitly clear because Chekuri et al. [4] rounded a slightly more complicated LP. The complication arises from the difficulty of solving (L)

directly for Steiner k-cut (which can be simplified by knapsack covering constraints, in hindsight). Morally, however, their proof extends to our setting here. For the sake of completeness, a proof of Lemma 15 is included in Appendix B.

Algorithm 2 A detailed implementation of a deterministic rounding algorithm for k-cut.

`greedy-cuts`$(G = (V, E), c, x)$

1. `let` $E' = \left\{e \in E : x_e \geq \frac{n-1}{2n}\right\}$, $E \leftarrow E \setminus E'$
2. `if` E' `is a` k`-cut then return` E'
3. `let` F `be a minimum weight spanning forest in` E `w/r/t` x `w/` ℓ `components`

// *Arrange the greedily induced components as subtrees of a dynamic forest*

4. `for each` $v \in V$
   A. `make a singleton tree labeled by` v
5. `for each edge` $f = \{u, v\} \in F$ `in increasing order of` x_f
   A. `let` T_u `and` T_v `be the rooted trees containing` u `and` v`, respectively.`
   B. `make` T_u `and` T_v `children of a new vertex labeled by` f

// *Compute the weight of each cut induced by a greedy component.*

6. `let each node in the dynamic forest have value` 0
7. `for each edge` $e = \{u, v\} \in E$
   A. `add` x_e `to the value of every node on the` u`-to-root and` v`-to-root paths`
   B. `let` w `be the least common ancestor` u `and` v
   C. `subtract` $2x_e$ `from the value of every node on the` w `to root paths`
8. `let` $v_1, v_2, \ldots, v_{k-\ell}$ `be the` $k - \ell$ `minimum value nodes in the dynamic forest. For` $i \in [k - \ell]$`, let` C_i `be the components induced by the leaves in the subtree rooted by` v_i`.`
9. `return` $E' \cup \partial(C_1) \cup \cdots \cup \partial(C_{k-\ell})$

It remains to implement `greedy-cuts` in $O(m \log n)$ time. With the help of dynamic trees [47], this can be done in a straightforward fashion. We briefly describe the full implementation; pseudocode is given in Algorithm 2. Recall from the conceptual sketch above that `greedy-cuts` requires up to $k - 1$ minimum greedy cuts of a minimum spanning forest w/r/t x. To compute the value of these cuts, `greedy-cuts` first simulates the greedy algorithm by processing the edges in the spanning forest in increasing order of x. The greedy algorithm repeatedly adds an edge that bridges two greedy components. We assemble a auxiliary forest of dynamic trees where each leaf is a vertex, and each subtree corresponds to a greedy component induced by the vertices at the leaves of the subtree.

After building this dynamic forest, we compute the number of edges in each cut. We associate each node in the dynamic forest with the greedy component induced by its leaves, and given each node an initial value of 0. We process edges one at a time and add its weight to the value of every node corresponding to a greedy component cutting that edge. Now, an edge in the original graph is cut by a greedy component iff the corresponding subtree in the dynamic forest does not contain both its end points as leaves. We compute the least common ancestor of the endpoints in the dynamic forest in $O(\log n)$ time [23], and add the weight of edge to every node between the leaves and the common ancestor, excluding the common ancestor. Adding the weight to every node on a node-to-root path takes $O(\log n)$ time [47] in dynamic trees. After processing every edge, we simply read off the value of each greedy cut as the value of the corresponding node in the forest. Thus we have the following.

▶ **Lemma 16.** `greedy-cuts` *can be implemented in* $O(m \log n)$ *time.*

Together, Lemma 15 and Lemma 16 imply Theorem 4.

B Proofs for Appendix A

Chekuri et al. [4] gave a rounding scheme for the more general problem of Steiner k-cut and the analysis extends to the rounding schemes presented here. We provide a brief sketch for the sake of completeness as there are some slight technical gaps. The proof is simpler and more direct in our setting because we have a direct fractional solution to (L), while Chekuri et al. [4] dealt with a solution to a slightly more complicated LP. We note that our analysis also extends to Steiner cuts. We take as a starting point the existence of a dual certificate from the primal-dual framework.

▶ **Lemma 17** ([15, 16]). *Let F be a minimum spanning forest in a undirected graph $G = (V, E)$ weighted by $x \in \mathbb{R}^{E'}_{\geq 0}$. Let $\mathcal{C}$ be the family of greedy cuts induced by F. Then there exists $y \in \mathbb{R}^{\mathcal{C}}_{\geq 0}$ satisfying the following properties.*[2]

(a) *For each edge $e \in E$, $\sum_{C \in \mathcal{C} : C \ni e} y_C \leq x_e$.*
(b) *For each edge $e \in F$, $\sum_{C \in \mathcal{C} : C \ni e} y_C = x_e$.*
(c) $2\big(1 - \frac{1}{n}\big)\langle y, \mathbb{1}\rangle \geq \sum_{e \in F} x_e$.

We note that the dual variables y can be computed in $O(n)$ time (after computing the minimum spanning forest).

▶ **Lemma 18** ([4]). `greedy-cuts` *returns a k-cut with total cost $\leq 2\big(1 - \frac{1}{n}\big)\langle c, x\rangle$.*

Proof sketch. Let y be as in Lemma 17. We first make two observations about y. First, since $x_e \leq \frac{n}{2(n-1)}$ for all $e \in E'$, we have (by property (a) of Lemma 17) that $y_C \leq \frac{n}{2(n-1)}$ for all greedy cuts C. Second, by (e) property (c) of Lemma 17 and (f) the feasibility of x w/r/t the k-cut LP (L), we have

$$2\left(1 - \frac{1}{n}\right)\langle y, \mathbb{1}\rangle \overset{\text{(e)}}{\geq} \sum_{e \in T} x_e \overset{\text{(f)}}{\geq} k - 1.$$

Let $C_1, \ldots, C_{k-1}$ be the $k-1$ minimum greedy cuts. Now, by (g) rewriting the sum of the $k-1$ minimum greedy cuts as the solution of a minimization problem, (h) observing that $2\big(1 - \frac{1}{n}\big)y$ is a feasible solution to the minimization problem, (i) interchanging sums, and (j) property (a) of Lemma 17, we have

$$\sum_{i=1}^{k-1} \overline{c}(C_i) \overset{\text{(g)}}{=} \min\{\langle y', \overline{c}\rangle : \mathbb{0} \leq y' \leq \mathbb{1},\ \text{support}(y') \subseteq \text{support}(y),\ \text{and } \|y'\|_1 \geq k - 1\}$$
$$\overset{\text{(h)}}{\leq} 2\left(1 - \frac{1}{n}\right)\langle y, \overline{c}\rangle = \sum_{C} y_C \sum_{e \in C} c_e \overset{\text{(i)}}{=} \sum_{e \in E'} c_e \sum_{C \ni e} y_C \overset{\text{(j)}}{\leq} \sum_{e \in E'} c_e x_e,$$

as desired. ◀

[2] Here we do not require property (b), but we mention it anyway because it is important in other applications.

Global Cardinality Constraints Make Approximating Some Max-2-CSPs Harder

Per Austrin
KTH Royal Institute of Technology, Stockholm, Sweden
austrin@kth.se

Aleksa Stanković
KTH Royal Institute of Technology, Stockholm, Sweden
aleksas@kth.se

Abstract

Assuming the Unique Games Conjecture, we show that existing approximation algorithms for some Boolean Max-2-CSPs with cardinality constraints are optimal. In particular, we prove that Max-Cut with cardinality constraints is UG-hard to approximate within ≈ 0.858, and that Max-2-Sat with cardinality constraints is UG-hard to approximate within ≈ 0.929. In both cases, the previous best hardness results were the same as the hardness of the corresponding unconstrained Max-2-CSP (≈ 0.878 for Max-Cut, and ≈ 0.940 for Max-2-Sat).

The hardness for Max-2-Sat applies to monotone Max-2-Sat instances, meaning that we also obtain tight inapproximability for the Max-k-Vertex-Cover problem.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Mathematics of computing → Approximation algorithms; Theory of computation → Approximation algorithms analysis

Keywords and phrases Constraint satisfaction problems, global cardinality constraints, semidefinite programming, inapproximability, Unique Games Conjecture, Max-Cut, Max-2-Sat

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.24

Category APPROX

Related Version A full version of the paper is available at `https://arxiv.org/abs/1907.04165`.

Funding Research supported by the Approximability and Proof Complexity project funded by the Knut and Alice Wallenberg Foundation.

Acknowledgements The authors thank Johan Håstad for helpful suggestions and comments on the manuscript. We also thank anonymous reviewers for their helpful remarks.

1 Introduction

Constraint satisfaction problems (CSPs) are one of the most fundamental objects studied in complexity theory. An instance of a CSP has a set of variables taking values over a certain domain and a set of constraints on tuples of these variables as an input. Probably the best known CSP is 3-Sat, in which the constraints are clauses, each clause is a disjunction of at most three literals, and each literal is either a variable or negation of a variable. In the satisfiability version of CSP problems, we are interested whether there is an assignment to the variables which satisfies all the constraints. Hardness of deciding satisfiability of CSPs is well understood, due to the dichotomy theorem [32] of Schaefer which shows that each CSP with variables taking values in a Boolean domain is either in P or NP-complete, and due to the more recent results of Bulatov [8] and Zhuk [35] which settle this question on general domains.


© Per Austrin and Aleksa Stanković;
licensed under Creative Commons License CC-BY



Another well-studied version is the Max-CSP, which is the optimization version in which we are interested in maximizing the number of constraints satisfied. This type of problem is NP-hard in most cases and we typically settle with finding a good estimate of the optimal solution, for which we rely on approximation algorithms. A common example of a constraint satisfaction problem in this setting is Max-Cut, in which the input consists of a graph G, and the goal is to partition the vertices into two sets such that the number of edges between the two parts is maximized. Approximability of Max-CSPs has been a major research topic which inspired many influential breakthroughs. One of the first surprising results was an algorithm of Goemans and Williamson [13], which uses semidefinite programming (SDP) to approximate the optimal solution to within a constant of $\alpha_{GW} \approx 0.878$. The SDP approach is also useful in approximating many other well known Max-CSPs, such as Max-3-Sat [19] within a constant of $7/8$ and Max-2-Sat [23] within $\alpha_{LLZ} \approx 0.9401$.

On the hardness of approximation side, the first NP-hardness results are based on the celebrated PCP theorem [1, 2] which provided a strong starting point for studying inapproximability. For example, a direct corollary of the PCP theorem shows that the Max-3-Sat problem cannot be approximated within $1 - \delta$ for some universal constant $\delta > 0$. By using the PCP theorem and parallel repetition [31] as a starting point, Håstad [17] proved optimal inapproximability for Max-3-Sat by showing that it cannot be approximated better than $7/8 + \epsilon$ for any $\epsilon > 0$.

However, despite further works relying on the similar techniques which improved our understanding of inapproximability for several additional CSPs, the progress on closing the gap between the best algorithm and the best hardness was at a standstill for some fundamental problems such as Max-Cut, until the Unique Games Conjecture (UGC) was introduced by Khot [20]. In particular, by assuming the UGC, optimality of the α_{GW}-approximation algorithm for Max-Cut and the α_{LLZ}-approximation algorithm for Max-2-Sat was shown in [21, 26] and [3], respectively. The strength of semidefinite programming for approximating Max-CSPs was corroborated in a breakthrough result of Raghavendra [28], which showed that assuming the UGC, a certain SDP relaxation achieves optimal approximation ratios for all Max-CSPs.

Locality of the constraints was of crucial importance in studying CSPs and Max-CSPs since their inception. Therefore, it is not a surprise that typical techniques fail when we work with CSPs for which feasible assignments need to satisfy some additional global constraints, and these problems almost always become harder. For example, while the satisfiability of a 2-Sat instance can be checked by a straightforward algorithm, Guruswami and Lee recently showed [14] that when the satisfying assignment needs to have exactly half of its variables set to true, this problem becomes NP-hard. Hardness of deciding satisfiability of CSPs in which we prescribe how many variables are assigned to certain values is well understood due to the dichotomy theorem of Bulatov and Marx [9], which shows that these problems are either NP-hard or in P, and gives a simple classification. Another type of global constraint is studied by Brakensiek et al. [7], who consider hardness of deciding CSPs in presence of modular constraints, which restrict cardinality of values in an assignment modulo a natural number M.

In this paper we are interested in optimization variants of CSPs with global cardinality constraints, i.e., constraints which specify the number of occurrences of each value from the domain in the assignment. We refer to these problems as CC-Max-CSPs. It is not hard to see that these problems are at least as hard to approximate as their unconstrained counterparts. CC-Max-CSPs have been actively studied in the past. For example the Max-Bisection problem, i.e., Max-Cut where the two partitions need to be of the same size, has been of

a particular interest, with a series of papers [12], [34],[16], [11], [30] obtaining improved approximation algorithms, until the most recent result which achieves an approximation ratio of 0.8776 [4], which is only $\approx 10^{-3}$ below the UG-hardness bound α_{GW}. The state-of-the-art algorithm [30] for the more general CC-Max-Cut problem achieves an approximation ratio of $\alpha_{cut}^{cc} \approx 0.858$. Another related CC-Max-CSP actively studied is CC-Max-2-Sat, and its monotone variant (a version in which negated literals are not allowed) Max-k-VC[1]. The best algorithm [30] up to date for general CC-Max-2-Sat achieves an approximation ratio of α_{2sat}^{cc}, where $\alpha_{2sat}^{cc} \approx 0.929$, which improved on a series of increasingly stronger algorithms presented in [33], [6], and [18]. Manurangsi [25] showed that it is UG-hard to approximate Max-k-VC within a factor $\alpha_{AKS} \approx 0.944$ (note that this is slightly larger than the hardness of $\alpha_{LLZ} \approx 0.940$ for general Max-2-Sat).

Yet another well-studied CC-Max-CSP is the *Densest k-Subgraph* (Max-k-DS) problem, in which we are given a graph and the objective is to find a maximally dense induced subgraph on k vertices. Analogously to the Max-k-VC problem, Max-k-DS can be viewed as the monotone CC-Max-2-And problem. Max-k-DS is qualitatively very different from the previously discussed problems. It is not known to be approximable within a constant factor, and is in fact known to be hard to approximate to within almost polynomial factors assuming the Exponential Time Hypothesis [24], or to within any constant factor assuming the Small-Set Expansion Hypothesis [29].

Obtaining tight approximability results for CC-Max-CSPs presents an important research topic. Qualitatively, it is also interesting to determine whether adding a cardinality constraint to a non-trivial Max-CSP makes approximation strictly harder. For example, we know that CC-Max-2-Sat is as hard as Max-2-Sat, but it is still conceivable that they are equally hard. In particular, it would be interesting to answer the following question:

"Can CC-Max-2-Sat be approximated within α_{LLZ}?"

So far the only result in this direction comes from [4] which shows that the "bisection version" (where the cardinality constraint is that exactly half of the variables must be set to true) of CC-Max-2-Sat can be approximated within α_{LLZ}. However, the approach taken in that algorithm does not immediately extend to handle general cardinality constraints. A similar question arises for the CC-Max-Cut problem, but here even the basic Max-Bisection problem is not known to be approximable within the Max-Cut constant $\alpha_{GW} \approx 0.878$. As far as we are aware, prior to this paper, the only examples of cardinality-constrained Max-CSPs being harder than their unconstrained counterparts were examples where the unconstrained version is easy (e.g. unconstrained Max-k-VC is monotone Max-2-Sat, and unconstrained Max-k-DS is monotone Max-2-And, which are both trivial).

Our Results

In this paper, we answer the above question negatively, by giving improved UG-hardness results for CC-Max-Cut and Max-k-VC.

▶ **Theorem 1.** *For every $\varepsilon > 0$, CC-Max-Cut is UG-hard to approximate within $\beta_{cut}^{cc} + \varepsilon$, where $\beta_{cut}^{cc} \approx 0.858$.*

▶ **Theorem 2.** *For every $\varepsilon > 0$, Max-k-VC is UG-hard to approximate within $\beta_{vc}^{cc} + \varepsilon$, where $\beta_{vc}^{cc} \approx 0.929$.*

[1] Max-k-VC is an abbreviation for maximum k vertex cover, in which we are given a graph and the task is to select a subset of k vertices covering as many of the edges as possible.

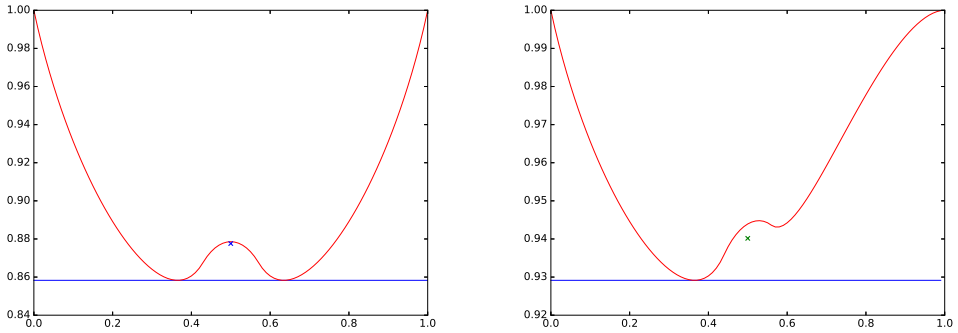


Figure 1 Hardness ratio (red) vs. approximation constant (blue) as a function of cardinality constraint q for CC-Max-Cut (left), as well as Max-k-VC (right). We also use $\times$ to highlight the approximation ratio known for Max-Bisection on the left plot, while $\times$ represents the optimal approximation constant for the CC-Max-2-Sat problem in the special case $q = 1/2$.

Note that since CC-Max-Cut and Max-k-VC are special cases of CC-Max-2-Lin and CC-Max-2-Sat respectively, the corresponding hardness results apply to the latter problems as well.

The constants β^{cc}_{vc} and β^{cc}_{cut} are calculated numerically and their estimated values match the constants α^{cc}_{2sat} and α^{cc}_{cut}, which are the approximation ratios for corresponding problems achieved by the algorithm of Raghavendra and Tan [30]. We provide even stronger evidence that these constants match each other, by showing that β^{cc}_{vc} and β^{cc}_{cut} are calculated as minima of the same functions used for calculating their counterparts α^{cc}_{vc} and α^{cc}_{cut}, but over a slightly more restricted domain.

Moreover, in Section 3 we give refined statements of Theorem 1 and Theorem 2 which describe inapproximability of these problems as a function of the cardinality constraint $q \in (0, 1)$, which specifies the fraction of variables that need to be set to true. For now, we provide a visualization of these results in Figure 1.

Overview of proof ideas

The main observation behind the hardness results is that the reduction used to prove hardness of approximation for the Independent Set and Vertex Cover problems in bounded degree graphs [5] gives very strong soundness guarantees. In particular it shows that in the "no" case of the reduction, all induced subgraphs of the graph contain many edges, which in turn gives useful upper bounds on the number of edges cut by a bipartition of a given size, or the number of edges covered by a subgraph. This is also how [25] obtained the previous hardness of ≈ 0.944 for Max-k-VC. Thus our results use essentially the same reduction as [5] (which is in turn similar to the reduction for Max-Cut [21]). Note however that even though the graph produced by that reduction has a small vertex cover in the "yes" case, using that small vertex cover is not necessarily the best solution for the Max-k-VC problem on the graph. In particular for $q < 1/2$, it makes more sense to instead use the large independent set as the Max-k-VC solution in the yes case (the intuition being that since it is independent, it covers many edges relative to its size).

Another difference is that we have somewhat greater flexibility in choosing the noise distribution of our "dictatorship test" (the key component of essentially all UG-hardness results) The reason is that for Independent Set/Vertex Cover, the reduction needs "perfect completeness", i.e., in the "yes" case it needs to produce graphs with large independent

sets/small vertex covers, whereas for e.g. Max-k-VC we are perfectly happy with graphs where there are sets of size k covering many, but not necessarily all, edges. This increased flexibility turns out to improve the hardness ratios for some range of the cardinality constraint q. For example, for the CC-Max-Cut problem with $q = 1/2$, this allows us to recover the α_{GW}-hardness for the Max-Bisection problem using the same reduction. However, at q further away from $1/2$, and in particular at the local minima in Figure 1, it turns out that this flexibility does not help. Thus in the global minimum at $q \approx 0.365$ for Max-k-VC, the reduction outputs a graph with a large independent set containing a q fraction of the vertices, and choosing that independent set is the optimal solution for the Max-k-VC instance. Similarly, at the local minimum with $q > 1/2$, the optimal solution to the Max-k-VC instance in the yes case is to pick an actual vertex cover of size q, and this point of the curve corresponds exactly to the hardness of 0.944 from [25].

Organization

This paper is organized as follows. In Section 2 we fix the notation, recall some well-known facts, and formally introduce the problems of interest. In Section 3 we give our improved inapproximability results. In Section 4 we give a brief overview of the algorithm of Raghavendra and Tan [30] in order to observe that the hardness ratios we get match the approximation ratios of the algorithm. Finally, in Section 5 we propose some possible directions for future research.

2 Preliminaries

2.1 Notational Conventions

In this paper we work with undirected (multi)graphs $G = (V, E)$. For a set $S \subseteq V$ of vertices we use S^c to denote its complement $S^c = V \setminus S$, and write $U \sqcup V$ for a disjoint union of sets U and V. The graphs are both edge and vertex weighted and the weights of vertices and edges are given by functions $w\colon V \to [0, 1]$, and $w\colon E \to [0, 1]$. For subsets $S \subseteq V$ and $K \subseteq E$ we interpret $w(S)$ and $w(K)$ as the sum of weights of vertices contained in S and edges in K, respectively. Furthermore, weights are normalized so that $w(V) = w(E) = 1$ and the weight of each vertex equals half the weight of all edges adjacent to it. Therefore, the weights of edges and vertices can be interpreted as probability distributions, and sampling a vertex with probability equal to its weight is the same as sampling an edge and then sampling one of its endpoints with probability $1/2$. For $S, T \subseteq V$, we write $w(S, T)$ for the total weight of edges from E which have one endpoint in S, and other in T. Note that, since we work with undirected graphs, the order of endpoints is not important, and therefore $w(S, T) = w(T, S)$. In other words, the weight of an edge $e = (u, v)$ contributes to $w(S, T)$ if either $(u, v) \in T \times S$ or $(u, v) \in S \times T$. We also have the identity

$$w(S, V) = w(S) + \frac{1}{2} w(S, S^c). \tag{1}$$

The set of all neighbours of a vertex v including v is denoted by $N(v)$, and the set of all neighbours of a set $S \subseteq V$ including S is denoted by $N(S)$. Let us also introduce the following definition.

▶ **Definition 3.** *A graph G is (q, ε)-dense if every subset $S \subseteq V$ with $w(S) = q$ satisfies $w(S, S) \geq \varepsilon$.*

We use $\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ to denote the density function of a standard normal random variable, and $\Phi(x) = \int_{-\infty}^{x} \phi(y)dy$ to denote its cumulative distribution function (CDF). We also work with bivariate normal random variables, and to that end introduce the following function.

▶ **Definition 4.** *Let $\rho \in [-1,1]$, and consider two jointly normal random variables X, Y with mean 0 and covariance matrix $\mathrm{Cov}(X,Y) = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$. We define $\Gamma_\rho \colon [0,1]^2 \to [0,1]$ as*

$$\Gamma_\rho(x,y) = \Pr\left[X \le \Phi^{-1}(x) \wedge Y \le \Phi^{-1}(y)\right].$$

We also write $\Gamma_\rho(x) = \Gamma_\rho(x,x)$. We have the following basic lemma (for a proof see Appendix A of [4]).

▶ **Lemma 5.** *For every $\rho \in [-1,1]$, and every $x, y \in [0,1]$, we have*

$$\Gamma_\rho(x,y) = \Gamma_\rho(1-x, 1-y) - 1 + x + y.$$

2.2 Problem Definitions

This paper is concerned with Max-Cut, Max-2-Lin, Max-2-Sat, and Max-k-VC problems with cardinality constraints. Let us give the definitions of these problems as integer optimization programs now. In these definitions instead of $\{0,1\}$ we represent Boolean domain as $\{-1,1\}$, and for that reason instead of cardinality constraint q we consider a *balance constraint* $r = 1 - 2q$.

▶ **Definition 6.** *An instance $\mathcal{F}$ of the cardinality constrained Max-2-Lin* (CC-Max-2-Lin) *problem with balance constraint $r \in (-1,1)$ over variables $X = \{x_1, \ldots, x_n\}$ taking values in $\{-1,1\}$ is given by the following integer optimization program*

$$\begin{aligned} \max \quad & \sum_{(i,j)=e_\ell \in E} \frac{1 + P_\ell x_i x_j}{2}, \\ s.t. \quad & \sum_{i \in V} x_i = nr, \end{aligned}$$

where $P_\ell \in \{-1,1\}$ and the term $(1+P_\ell x_i x_j)/2$ corresponds to the XOR constraint $x_i x_j = P_\ell$. In case $P_\ell = -1$ for all ℓ, the integer optimization program is an instance of CC-Max-Cut *problem.*

▶ **Definition 7.** *An instance $\mathcal{F}$ of the cardinality constrained Max-2-Sat* (CC-Max-2-Sat) *problem with balance constraint $r \in (-1,1)$ over variables $X = \{x_1, \ldots, x_n\}$ taking values in $\{-1,1\}$ is given by the following integer optimization program*

$$\begin{aligned} \max \quad & \sum_{(i,j)=e_\ell \in E} \frac{3 + P_\ell^1 x_i + P_\ell^2 x_j + P_\ell^3 x_i x_j}{4}, \\ s.t. \quad & \sum_{i \in V} x_i = nr, \end{aligned}$$

where $(P_\ell^1, P_\ell^2, P_\ell^3) \in \{(-1,-1,-1), (1,-1,1), (-1,1,1), (1,1,-1)\}$ corresponds to one of the four possible clauses

$$x_i \vee x_j, \quad \neg x_i \vee x_j, \quad x_i \vee \neg x_j, \quad \neg x_i \vee \neg x_j.$$

In case $(P_\ell^1, P_\ell^2, P_\ell^3) = (-1,-1,-1)$ for all ℓ, the integer optimization program is an instance of Max-k-VC *problem.*

The objective in the problems given by Definitions 6 and 7 is to find an assignment $z\colon X \to \{-1,1\}$ which satisfies a (hard) global cardinality constraint and maximizes the number of satisfied soft constraints represented by the objective function. For an assignment z that satisfies global constraint of an instance $\mathcal{F}$ we use $\mathrm{Val}_z(\mathcal{F})$ to denote the value of the objective function under the assignment z. Furthermore, we use

$$\mathrm{OptVal}(\mathcal{F}) = \max_{\substack{z\colon X\to\{-1,1\} \\ \sum_{x\in X} z(x)=rn}} \mathrm{Val}_z(\mathcal{F})$$

to denote the maximum value of the objective function over all assignments z satisfying the cardinality constraint.

The starting point of the hardness results in this paper is the Unique Games problems, which is defined as follows.

▶ **Definition 8.** *A* Unique Games instance $\Lambda = (\mathcal{U}, \mathcal{V}, \mathcal{E}, \Pi, [L])$ *consists of an unweighted bipartite multigraph* $(\mathcal{U} \sqcup \mathcal{V}, \mathcal{E})$*, a set* $\Pi = \{\pi_e\colon [L] \to [L] \mid e \in \mathcal{E} \text{ and } \pi_e \text{ is a bijection}\}$ *of permutation constraints, and a set* $[L]$ *of labels. The value of* Λ *under the assignment* $z\colon \mathcal{U} \sqcup \mathcal{V} \to [L]$ *is the fraction of edges satisfied, where an edge* $e = (u,v), u \in \mathcal{U}, v \in \mathcal{V}$ *is satisfied if* $\pi_e(z(u)) = z(v)$*. We write* $\mathrm{Val}_c(\Lambda)$ *for the value of* Λ *under* z*, and* $\mathrm{Opt}(\Lambda)$ *for the maximum possible value over all assignments* z*.*

The Unique Games Conjecture [20] can be formulated as follows ([22], Lemma 3.4).

▶ **Conjecture 9** (Unique Games Conjecture)**.** *For every constant* $\gamma > 0$ *there is a sufficiently large* $L \in \mathbb{N}$*, such that for a Unique Games instance* $\Lambda = (\mathcal{U}, \mathcal{V}, \mathcal{E}, \Pi, [L])$ *with a regular bipartite graph* $(\mathcal{U} \sqcup \mathcal{V}, \mathcal{E})$*, it is NP-hard to distinguish between*

- $\mathrm{Opt}(\Lambda) \geq 1 - \gamma$*,*
- $\mathrm{Opt}(\Lambda) \leq \gamma$*.*

2.3 Analysis of Boolean Functions

One of the ubiquitous tools in the hardness of approximation area is Fourier analysis of Boolean functions. We now recall some of the well-known facts which are used in the paper. For a more detailed study, we refer to [27].

For $q \in [0,1]$ and $n \in \mathbb{N}$ we write $\pi_q\colon \{0,1\} \to [0,1]$ for the probability distribution given by $\pi_q(1) = q, \pi_q(0) = 1-q$. We also write $\pi_q^{\otimes n}$ for the probability distribution on n-bit strings $x \in \{0,1\}^n$ where each bit is distributed according to π_q, independently. We use $L^2(\pi_q^{\otimes n})$ to denote the space of random variables $f\colon \{0,1\}^n \to \mathbb{R}$ over the probability space $\left(\{0,1\}^n, P(\{0,1\}^n), \pi_q^{\otimes n}\right)$, and interpret $\mathbf{E}[f]$ and $\mathbf{Var}[f]$ as expectation and variance of $f(X)$ when the X is drawn from $\pi_q^{\otimes n}$. Depending on context, the elements of $L^2(\pi_q^{\otimes n})$ will be interpreted as functions as well.

Let us now introduce some of the common objects used in the study of Boolean functions.

▶ **Definition 10.** *Consider a function* $f \in L^2(\pi_q^{\otimes n})$ *and* $i \in \{1,\dots,n\}$*. The* influence $\mathbf{Inf}_i[f]$ *of the* i*-th argument on* f *is defined as*

$$\mathbf{Inf}_i[f] = \mathbf{E}_{x\sim\pi_q^{\otimes n}}[\mathbf{Var}_{\tilde{x}_i\sim\pi_q}[f(x_1,\dots,x_{i-1},\tilde{x}_i,x_{i+1},\dots,x_n)]].$$

Minimal correlation between two q-biased bits is $\max(-q/(1-q), -(1-q)/q)$. For notational convenience, let us introduce the function κ which assigns to each value $q \in (0,1)$ an interval $I \subseteq (-1,0)$ as

$$\kappa(q) = \begin{cases} [-q/(1-q), 0), & \text{if } q < 1/2, \\ (-1,0), & \text{if } q = 1/2, \\ [-(1-q)/q, 0), & \text{if } q > 1/2. \end{cases}$$

▶ **Definition 11.** *For a fixed $x \in \{0,1\}, q \in (0,1)$ and $\rho \in \kappa(q)$ we write $y \sim N_\rho(x)$ to indicate that y is a ρ-correlated copy of x. In particular each bit y_i is equal to 1 with probability $q + \rho(1-q)$ if $x_i = 1$, and $y_i = 1$ with probability $q - \rho q$ when $x_i = 0$, independently.*

▶ **Definition 12.** *Consider $q \in (0,1)$ and $\rho \in \kappa(q)$. The* noise operator $T_\rho \colon L^2(\pi_q^{\otimes n}) \to L^2(\pi_q^{\otimes n})$ *is defined as*

$$T_\rho f(x) = \mathbf{E}_{y \sim N_\rho(x)}[f(y)].$$

The following lemma gives a useful bound on the number of influential variables of $T_\rho f$.

▶ **Lemma 13.** *Consider $q \in (0,1)$, a function $f \in L^2(\pi_q^{\otimes n})$, and $\rho \in \kappa(q)$. Then, for any $\tau > 0$ we have that*

$$|\{i \in [n] \mid \mathbf{Inf}_i[T_\rho f] \geq \tau\}| \leq \frac{\mathbf{Var}[f]}{\tau e \ln(1/|\rho|)}.$$

For a proof we refer to Lemma 3.4 of [15]. We also need to introduce the notion of noise stability, defined as follows.

▶ **Definition 14.** *Let $q \in (0,1), \rho \in \kappa(q)$ and $f \in L^2(\pi_q^{\otimes n})$. The* noise stability *of function f at ρ is defined as*

$$\mathbb{S}_\rho = \mathbf{E}[f \cdot T_\rho f].$$

Let us also recall the following variant of the "Majority is Stablest" theorem in the form that appeared in [5], and which follows from Theorem 3.1 in [10].

▶ **Theorem 15.** *Let $q \in (0,1)$ and $\rho \in \kappa(q)$. Then for any $\varepsilon > 0$, there exist $\tau > 0$ and $\delta > 0$ such that for every function $f \in L^2(\pi_q^{\otimes n})$, $f \colon \{-1,1\}^n \to [0,1]$ that satisfies*

$$\max_{i \in [n]} \mathbf{Inf}_i[T_{1-\delta} f] \leq \tau,$$

we have

$$\mathbb{S}_\rho(f) \geq \Gamma_\rho(\mathbf{E}[f]) - \varepsilon.$$

3 Hardness Reduction

In this section we give our main hardness reduction. As discussed in the introduction, it is a generalization of the reduction of Theorem III.1 from [5].

▶ **Theorem 16.** *For every $q \in (0,1), \varepsilon > 0$, and $\rho \in \kappa(q)$, there exists a $\gamma > 0$ and a reduction from Unique Games instances $\Lambda = (\mathcal{U}, \mathcal{V}, \mathcal{E}, \Pi, [L])$ to weighted multigraphs $G = (V, E)$ with the following properties:*

- Completeness: *If $\mathrm{Opt}(\Lambda) \geq 1 - \gamma$, then there is a set $S \subseteq V$ such that $w(S) = q$ and $w(S, S^c) \geq 2q(1-q)(1-\rho) - 2\gamma$.*
- Soundness: *If $\mathrm{Opt}(\Lambda) \leq \gamma$, then for every $r \in [0,1]$, G is $(r, \Gamma_\rho(r) - \varepsilon)$-dense.*

Moreover, the running time of the reduction is polynomial in $|\mathcal{U}|, |\mathcal{V}|, |\mathcal{E}|$, and exponential in L.

Proof. Let $\nu \colon \{0,1\}^2 \to [0,1]$ be the probability distribution over two ρ-correlated q-biased bits. In other words, letting $t = (q - q^2)(1-\rho)$, we have

$$\nu(0,0) = 1 - q - t, \quad \nu(0,1) = \nu(1,0) = t, \quad \nu(1,1) = q - t.$$

Let us now describe how the multigraph G can be constructed from Λ. We define the vertex set of G to be $V = \mathcal{V} \times \{0,1\}^L = \{(v,x) \mid v \in \mathcal{V}, x \in \{0,1\}^L\}$. In particular, for every vertex $v \in \mathcal{V}$ we create 2^L vertices of G, which we identify with L-bit strings in $\{0,1\}^L$. We also write v^x for a vertex (v,x) of the graph G. The weights of vertices in G are given by

$$w(v^x) = \frac{1}{|\mathcal{V}|}\pi_q^{\otimes L}(x). \tag{2}$$

The edges of G are constructed in the following way. For every $u \in \mathcal{U}$, and for every two $v_1, v_2 \in N(u)$, we create an edge between vertices v_1^x, v_2^y with weight

$$\frac{1}{|\mathcal{U}|D^2}\nu^{\otimes L}(x \circ \pi_{e_1}, y \circ \pi_{e_2}), \quad \text{where } e_1 = (u,v_1), \quad e_2 = (u,v_2).$$

Expressed formally, the edge set E is

$$E = \{(e_1^x, e_2^y) \mid e_1 = (u,v_1), e_2 = (u,v_2), u \in \mathcal{U}, v_1, v_2 \in \mathcal{V}, x, y \in \{0,1\}^L\}.$$

Since the marginal of the distribution ν over either the first or the second argument is a q-biased distribution on $\{0,1\}^L$, the weight of all edges adjacent to a vertex v^x equals two times the weight of the vertex v^x. Furthermore, it is trivial to check that $w(V) = w(E) = 1$. The number of vertices in G is $|\mathcal{V}|2^L$, and the number of edges is $|\mathcal{U}|D^2 2^L$, so the construction is indeed polynomial in $|\mathcal{U}|, |\mathcal{V}|$ and $|\mathcal{E}|$.

Let us now prove completeness and soundness.

Completeness: Since $\mathrm{Opt}(\Lambda) \geq 1-\gamma$, there is a labeling $z \colon \mathcal{U} \sqcup \mathcal{V} \to [L]$ such that $\mathrm{Val}_z(\Lambda) \geq 1-\gamma$. Consider a set S given by

$$S = \{v^x \in V \mid x_{z(v)} = 1\}.$$

The weight of the set S is obviously q. Let us consider a set consisting of pairs of edges in $\mathcal{E}$ which have a common vertex in $\mathcal{U}$, i.e. the set

$$\hat{E} = \{(e_1,e_2) \in \mathcal{E} \times \mathcal{E} \mid e_1 = (u,v_1), e_2 = (u,v_2), u \in \mathcal{U}, v_1, v_2 \in \mathcal{V}\},$$

and its subset $\hat{E}_{\mathrm{good}}$ consisting of edge pairs which are satisfied under the assignment z, or formally

$$\hat{E}_{\mathrm{good}} = \{(e_1,e_2) \in \hat{E} \mid e_1 = (u,v_1), e_2 = (u,v_2), z(u) = \pi_{e_1}^{-1}(z(v_1)) = \pi_{e_2}^{-1}(z(v_2))\},$$

Since at least fraction $1-\gamma$ of edges in $\mathcal{E}$ are satisfied under z, at least fraction $(1-\gamma)^2$ of edge pairs in $\hat{E}$ is satisfied under z, i.e. $|\hat{E}_{\mathrm{good}}| \geq (1-\gamma)^2|\hat{E}|$. For every $(e_1,e_2) \in \hat{E}_{\mathrm{good}}, e_1 = (u,v_1), e_2 = (u,v_2)$, the edges between S and S^c created through the pair of edges (e_1,e_2) have the total weight of

$$\begin{aligned}\frac{1}{|\mathcal{U}|D^2}\Pr_{(x,y)\sim\nu^{\otimes L}}\left[(x \circ \pi_{e_1^{-1}})_{z(v_1)} \neq (y \circ \pi_{e_2^{-1}})_{z(v_2)}\right] &= \frac{1}{|\mathcal{U}|D^2}\Pr_{(x,y)\sim\nu^{\otimes L}}\left[x_{z(u)} \neq y_{z(u)}\right] \\ &= \frac{1}{|\mathcal{U}|D^2}\left(\nu(0,1)+\nu(1,0)\right) = \frac{1}{|\mathcal{U}|D^2}2t.\end{aligned}$$

Therefore, we have $w(S,S^c) \geq 2t(1-\gamma)^2 \geq 2q(1-q)(1-\rho) - 2\gamma$.

Soundness: Let us assume by contradiction that G is not $(r, \Gamma_\rho(r)-\varepsilon)$-dense, and therefore that there is a set $S \subseteq V$ of weight $w(S) = r$ for which $w(S,S) < \Gamma_\rho(r) - \varepsilon$. For each $v \in \mathcal{V}$, let us define a function $S_v \in L^2(\pi_q^{\otimes L})$ to be the indicator function of S restricted

to the vertex v. In particular, we have that $S_v(x) = 1$ if and only if $v^x \in S$. Furthermore, for all $u \in \mathcal{U}$ let us define $S_u \in L^2(\pi_q^{\otimes L})$ as

$$S_u(x) = \mathop{\mathbf{E}}_{\substack{e=(u,v),\\ v\in N(u)}} [S_v(x \circ \pi_e^{-1})].$$

We have that

$$\begin{aligned} w(S,S) &= \mathop{\mathbf{E}}_{\substack{u\in\mathcal{U},\\ e_1=(u,v_1),e_2=(u,v_2)\\ v_1,v_2\in N(u)}} \left[\mathop{\mathbf{E}}_{(x,y)\sim\nu^{\otimes L}} [S_{v_1}(x \circ \pi_{e_1}^{-1}) S_{v_2}(y \circ \pi_{e_2}^{-1})] \right] \\ &= \mathop{\mathbf{E}}_{\substack{u\in\mathcal{U},\\ (x,y)\sim\nu^{\otimes L}}} \left[\mathop{\mathbf{E}}_{\substack{e_1=(u,v_1),e_2=(u,v_2)\\ v_1,v_2\in N(u)}} [S_{v_1}(x \circ \pi_{e_1}^{-1}) S_{v_2}(y \circ \pi_{e_2}^{-1})] \right] \\ &= \mathop{\mathbf{E}}_{u\in\mathcal{U}} \left[\mathop{\mathbf{E}}_{(x,y)\sim\nu^{\otimes L}} [S_u(x) S_u(y)] \right] = \mathop{\mathbf{E}}_{u\in\mathcal{U}} \left[\mathop{\mathbf{E}}_{x\sim\pi_q^{\otimes L}} [S_u(x) T_\rho S_u(x)] \right] = \mathop{\mathbf{E}}_{u\in\mathcal{U}} [\mathbb{S}_\rho(S_u)]. \end{aligned}$$

Let us define $\mu_u = \mathbf{E}_{x\sim\pi_q^{\otimes L}}[S_u(x)]$, and remark that due to regularity of Λ we have $\mathbf{E}_{u\in U}[S_u] = r$. We claim that there is a set $\mathcal{U}' \subseteq \mathcal{U}, |\mathcal{U}'| \geq \varepsilon|\mathcal{U}|/2$ such that for every $u \in \mathcal{U}'$ we have $\mathbb{S}_\rho(S_u) < \Gamma_\rho(\mu_u) - \varepsilon/2$. Otherwise, we reach a contradiction by noticing that

$$\begin{aligned} \Gamma_\rho(r) - \varepsilon > w(S,S) = \mathop{\mathbf{E}}_{u\in\mathcal{U}} [\mathbb{S}_\rho(S_u)] &\geq (1-\varepsilon/2) \left(\mathop{\mathbf{E}}_{u\in U} [\Gamma_\rho(\mu_u)] - \varepsilon/2 \right) \\ &\geq \mathop{\mathbf{E}}_{u\in U} [\Gamma_\rho(\mu_u)] - \varepsilon \geq \Gamma_\rho(r) - \varepsilon, \end{aligned}$$

where in the last inequality we used the fact that Γ_ρ is convex.

By Theorem 15 there is $\tau > 0$ and $\delta > 0$ such that for every $u \in \mathcal{U}'$ there is a significant coordinate $i \in [L]$ for which $\mathbf{Inf}_i[T_{1-\delta}S_u] \geq \tau$. For each $u \in \mathcal{U}'$ and for its significant coordinate i, by using the fact that $\mathbf{Inf}_i$ is convex and Markov's inequality we conclude that for at least $\tau/2$ of $v \in N(u)$ we have

$$\mathbf{Inf}_{\pi_e(i)}[T_{1-\delta}S_v] \geq \tau/2, \quad e = (u,v).$$

For each $v \in \mathcal{V}$ let $[L]_v \subseteq [L]$ denote a set of labels defined by

$$[L]_v = \{i \in [L] \mid \mathbf{Inf}_i[T_{1-\delta}S_v] \geq \tau/2\}.$$

By Lemma 13 we have that $|[L]_v| \leq \frac{2}{\tau e \ln(1/(1-\delta))}$. Let us now pick an assignment $z : \mathcal{U} \sqcup \mathcal{V} \to [L]$ of Λ using the following randomized procedure. For each $v \in \mathcal{V}$, pick $i \in [L]_v$ randomly, and set $z(v) = i$. If $[L]_v = \emptyset$, we pick $i \in [L]$ randomly. Then, for each $u \in \mathcal{U}$, we set $z(u) = i$ for the i that maximizes the number of edges satisfied. From the previous discussion we conclude that this labeling satisfies $\Omega\left(\varepsilon\tau^4 \ln^2(1/(1-\delta))\right)$ of constraints of Λ in expectation. But since this constant does not depend on γ this would be a contradiction if we started with a sufficiently small γ. ◀

3.1 Hardness for CC-Max-Cut

Now that we have proven Theorem 16, it is straightforward to prove the following theorem which gives a hardness result of CC-Max-Cut.

▶ **Theorem 17.** *For any $q \in (0,1)$ and $\rho \in \kappa(q)$ it is UG-hard to approximate CC-Max-Cut with cardinality constraint q within $\beta^{cc}_{cut}(q,\rho)+\varepsilon$ where $\varepsilon > 0$ is arbitrary small and $\beta^{cc}_{cut}(q,\rho)$ is given by*

$$\beta^{cc}_{cut}(q,\rho) = \frac{1-\Gamma_\rho(q)-\Gamma_\rho(1-q)}{2(q-q^2)(1-\rho)}.$$

Proof. By Theorem 16 there exists a family of multigraphs $G=(V,E)$ for which it is UG-hard to decide between the following two statements:

- There is a set $S \subseteq V, w(S) = q$, such that $w(S,S^c) \geq 2q(1-q)(1-\rho) - 2\gamma$.
- For any $r \in [0,1]$ and every set $T \subseteq V, w(T) = r$ we have $w(T,T) \geq \Gamma_\rho(r) - \varepsilon$.

The second statement implies that for any $S \subseteq V, w(S) = q$, we have $w(S,S^c) = w(V,V) - w(S,S) - w(S^c,S^c) \leq 1 - \Gamma_\rho(q) - \Gamma_\rho(1-q) + 2\varepsilon$. Therefore, by setting γ sufficiently small this shows UG-hardness of approximating CC-Max-Cut with cardinality constraint q within

$$\frac{1-\Gamma_\rho(1-q)-\Gamma_\rho(q)}{2q(1-q)(1-\rho)} + 2\varepsilon,$$

where $\varepsilon > 0$ is arbitrarily small. This reduction yields a weighted graph, which can be easily converted into an unweighted multigraph, using e.g. a simple reduction from Step 1 of Theorem 4.1. in [5]. ◀

3.2 Hardness for Max-k-VC

Next we give the hardness result for Max-k-VC.

▶ **Theorem 18.** *Consider $q \in (0,1)$ and let $\rho \in \kappa(q)$. Then, it is UG-hard to approximate Max-k-VC with cardinality constraint q within $\beta^{cc}_{vc}(q,\rho)+\varepsilon$ where $\varepsilon > 0$ is arbitrary small and $\beta^{cc}_{vc}(q,\rho)$ is given by*

$$\beta^{cc}_{vc}(q,\rho) = \frac{1-\Gamma_\rho(1-q)}{q(1+(1-q)(1-\rho))}.$$

Proof. As we have shown in Theorem 16, there is a family of multigraphs $G=(V,E)$ for which it is UG-hard to decide between the following two statements:

- There is a set $S \subseteq V, w(S) = q$, such that $w(S,S^c) \geq 2q(1-q)(1-\rho) - 2\gamma$.
- For any $r \in [0,1]$ and every set $T \subseteq V, w(T) = r$ we have $w(T,T) \geq \Gamma_\rho(r) - \varepsilon$.

By (1), the first item implies that $w(S,V) = q(1+q(1-q)(1-\rho)) - \gamma$. The second statement implies that for any $S \subseteq V, w(S) = q$, we have $w(S,V) = w(V,V) - w(S^c,S^c) \leq 1 - \Gamma_\rho(1-q) + \varepsilon$. Therefore, by letting $\gamma \to 0$ this shows UG-hardness of approximating Max-k-VC with cardinality constraint q within

$$\frac{1-\Gamma_\rho(1-q)}{q(1+(1-q)(1-\rho))} + \varepsilon,$$

where $\varepsilon > 0$ is arbitrarily small. As in the CC-Max-Cut case, this reduction yields a weighted graph, which can be converted into an unweighted multigraph by using the reduction from [5]. ◀

3.3 Hardness as a Function of the Cardinality Contraint

As we have concluded in Theorems 17 and 18, it is UG-hard to approximate CC-Max-Cut and Max-k-VC with cardinality constraint $q \in (0,1)$ to within

$$\beta^{cc}_{cut}(q) = \inf_{\rho \in \kappa(q)} \beta^{cc}_{cut}(q,\rho), \quad \beta^{cc}_{vc}(q) = \inf_{\rho \in \kappa(q)} \beta^{cc}_{vc}(q,\rho),$$

respectively. For a fixed q it is not clear for which ρ the functions $\beta^{cc}_{cut}(q,\cdot)$ and $\beta^{cc}_{vc}(q,\cdot)$ are minimized. For the plots of the inapproximability curves in Figure 1, the optimization over ρ was done numerically. Interestingly, numerical calculations show that the worst-case value of the cardinality constraint $q < 1/2$ (the value of q at which the hardness ratio meets the approximation ratio) is the same for Max-k-VC and CC-Max-Cut, and in particular its value is $q^* \approx 0.365$. The value of the correlation parameter ρ for which this worst-case hardness is achieved is extremal, i.e., $\rho = -q^*/(1-q^*) \approx -0.575$. However, the local minima at $q > 1/2$ in the two curves do not occur at the same value of q. For CC-Max-Cut the curve is symmetric around $1/2$ and the minimum occurs at $1 - q^* \approx 0.635$, but for the less symmetric Max-k-VC problem it occurs at ≈ 0.574.

Furthermore, for all $q \le q^*$ and also for $q > 1/2$ greater than the respective local minimum, the ρ minimizing both $\beta^{cc}_{cut}(q,\rho)$ and $\beta^{cc}_{vc}(q,\rho)$ is the minimum value of $\kappa(q)$. On the other hand, when q is close to $1/2$, the best choice of ρ does not equal $\min \kappa(q)$. For example, when $q = 1/2$, the hardness we obtain for CC-Max-Cut is the same as for the Max-Cut problem, attained using the value $\rho \approx -0.689$.

4 Approximation Algorithm

In this section we recall the algorithm of Raghavendra and Tan [30], somewhat reformulated in order to obtain explicit expressions for the approximation ratios that match the hardness results we obtain. We keep the exposition at a high level and skip over certain technical details, and refer the reader interested in the details to [30] or the follow-up work [4].

In order to find a good approximation for NP-hard integer optimization problems given in Definitions 6 and 7 we use semidefinite programming (SDP) relaxations. In particular, we extend the domain of variables $\{x_i\}_{i=1}^n$ from $\{0,1\}$ to vectors on an n-sphere, which we denote by $v_i \in S^n$. We also introduce a vector $v_0 \in S^n$ which represents the value false (corresponding value is 1 in the integer program). Then, we replace x_i by the scalar product $\langle v_0, v_i\rangle$ and $x_i x_j$ with $\langle v_i, v_j\rangle$. For example, the semidefinite relaxation of the CC-Max-Cut program is given as

$$\begin{aligned} \max \sum_{(i,j)=e_\ell \in E} & \frac{1-\langle v_i, v_j\rangle}{2}, \\ \text{s.t.} \sum_{i\in V} & \langle v_i, v_0\rangle = rn. \end{aligned}$$

Furthermore, since $|x_i - x_j| \le |x_i - x_k| + |x_k - x_j|$, we also demand from the vectors v_i to satisfy the triangle inequalities $\|v_i - v_j\|_2^2 \le \|v_i - v_0\|_2^2 + \|v_0 - v_j\|_2^2$. In order to relax the notation we define $\mu_i = \langle v_0, v_i\rangle$, $\rho_{ij} = \langle v_i, v_j\rangle$, and write triangle inequalities as

$$\begin{aligned} \mu_i + \mu_j + \rho_{ij} &\ge -1, & \mu_i - \mu_j - \rho_{ij} &\ge -1, \\ -\mu_i + \mu_j - \rho_{ij} &\ge -1, & -\mu_i - \mu_j + \rho_{ij} &\ge -1. \end{aligned}$$

The triples (μ_1, μ_2, ρ) satisfying triangle inequalities will be called *configurations*. We denote the set of all configurations as $\mathbf{Conf} \subseteq [-1,1]^3$. We can solve a semidefinite program up to desired accuracy in polynomial time. Then, the main challenge is finding a *rounding algorithm* which translates the vectors $\{v_i\}_{i=0}^n$ back to $\{-1,1\}$ so that they satisfy the balance constraint, and such that the rounding does not incur a big loss in the objective value. Raghavendra and Tan used a randomized rounding procedure, which rounds vectors

$\{v_i\}_{i=0}^n$ to ± 1 integers $\{y_i\}_{i=1}^n$ in the following way. First, let us define $w_i = v_i - \mu_i v_0$, and let[2] $\overline{w}_i = w_i/\|w_i\|$. Then, we draw a vector g from the Gaussian distribution $\mathcal{N}(0, I^{n+1})$ and set the values of $\overline{y}_i$ as

$$\overline{y}_i = \begin{cases} 1 & \text{if } \langle g, \overline{w}_i \rangle \geq \Phi^{-1}\left(\frac{1-\mu_i}{2}\right), \\ -1 & \text{otherwise.} \end{cases}$$

It is trivial to check that $\mathbf{E}[\overline{y}_i] = \mu_i$, so we have $\mathbf{E}\left[\sum_{i=1}^n \overline{y}_i\right] = rn$, and therefore the solution $\{\overline{y}_i\}_{i=1}^n$ satisfies the balance constraint in expectation. Furthermore, as shown in [30], using additional levels of the Lasserre hierarchy we can guarantee that with probability $1-\delta$ the sampled solution $\{\overline{y}_i\}_{i=1}^n$ is $O(\delta)$-far away from satisfying the balance constraint, where $\delta > 0$ can be chosen arbitrarily small. Therefore, we can change the values of at most $O(\delta)n$ variables $\overline{y}_i$ to get a solution y_i exactly satisfying the balance constraint, while losing only an additional small factor $O(\delta)$ in the objective value. Thus, it is sufficient to show that the objective value of the $\overline{y}_i$'s is large.

Consider now the SDP relaxation for any of the integer programs $\mathcal{F}$ given in either Definition 6 or Definition 7, and let $\mathrm{SDPVal}(\mathcal{F})$ be the optimal value of the SDP relaxation for the instance $\mathcal{F}$. We have that $\mathrm{SDPVal}(\mathcal{F}) \geq \mathrm{OptVal}(\mathcal{F})$. Finally, let us define $\mathrm{RndVal}(\mathcal{F})$ to be the expectation of the value of the objective function after randomized rounding procedure. The analysis of the approximation ratio for the algorithm boils down to proving $\mathrm{RndVal}(\mathcal{F}) \geq \alpha\,\mathrm{SDPVal}(\mathcal{F})$, where α is a constant that depends on the problem of interest. The way to calculate α is to look at the loss incurred by rounding at each constraint. Let us now show how this can be done for the CC-Max-Cut problem.

The expected value of each constraint $\frac{1-x_i x_j}{2}$ after rounding the SDP solution of CC-Max-Cut problem is $\frac{1-\mathbf{E}[\overline{y}_i \overline{y}_j]}{2}$, and therefore at each constraint the loss factor incurred by rounding is given as

$$\frac{1-\mathbf{E}\left[\overline{y}_i \overline{y}_j\right]}{2} \cdot \frac{1}{(1-\langle v_i, v_j\rangle)/2}.$$

Thus, in order to calculate the approximation ratio, we need to bound this expression from below. Let us first note that

$$\mathbf{E}[\overline{y}_1 \overline{y}_2] = 4\Gamma_{\overline{\rho}}\left(\frac{1-\mu_1}{2}, \frac{1-\mu_2}{2}\right) + \mu_1 + \mu_2 - 1,$$

where $\overline{\rho}$ is given as

$$\overline{\rho} = \frac{\rho - \mu_1\mu_2}{\sqrt{1-\mu_1^2}\sqrt{1-\mu_2^2}}.$$

Then, the approximation ratio is lower bounded by the quantity α_{cut}^{cc} defined as the solution of the optimization problem

$$\alpha_{cut}^{cc} = \min_{(\mu_1,\mu_2,\rho)\in\mathbf{Conf}} \frac{2 - 4\Gamma_{\overline{\rho}}\left(\frac{1-\mu_1}{2}, \frac{1-\mu_2}{2}\right) - \mu_1 - \mu_2}{1-\rho}.$$

[2] We assume that $\|w_i\| \neq 0$, since we can introduce a small perturbation to the values v_i without affecting the objective value too much.

Computing α_{cut}^{cc} is a hard global optimization problem, and therefore we resort to numerical computations to estimate it (we remark that the same approach is taken for a similar function in [23] and [3]). Extensive numerical experiments show that the minimum is attained at $\mu_1 = \mu_2 = \mu$, while the ρ is on the boundary of the polytope **Conf**, $\rho = -1 + 2|\mu|$. More precisely, the minimum is attained at $\mu \approx 0.27$, and $\rho \approx -0.575$, and it has a value of approximately 0.858.

Assuming that the minimum is attained at the configuration of the form $(\mu, \mu, -1+2\mu), \mu > 0$, constant α_{cut}^{cc} can be found as the minimum of a function

$$\frac{1 - 2\Gamma_{\overline{\rho}}\left(\frac{1-\mu}{2}, \frac{1-\mu}{2}\right) - \mu}{1-\mu},$$

where $\mu \in (0,1)$. If we introduce $q = (1-\mu)/2$, we can reexpress this function as

$$\alpha_{cut}^{cc}(q) = \frac{2q - 2\Gamma_{\overline{\rho}}(q)}{2q} = \frac{1 - \Gamma_{\overline{\rho}}(q) - \Gamma_{\overline{\rho}}(1-q)}{2q}, \quad q \in (0, 1/2),$$

where in the last equality we used Lemma 5. Furthermore, $\overline{\rho} = -q/(1-q)$. Similar analysis for CC-Max-2-Lin shows that the approximation ratio is the minimal value of the same function.

Straightforward calculations show that $\beta_{cut}^{cc}(q, -q/(1-q))$ from Theorem 17 equals the value of $\alpha_{cut}^{cc}(q)$. Therefore, under the (mild) assumption that worst-case configurations indeed take the special form as explained above, our hardness result is sharp and the algorithm for CC-Max-Cut of Raghavendra and Tan is optimal on general instances of CC-Max-Cut / CC-Max-2-Lin.

In completely analogous way, we can conclude that the approximation ratio for CC-Max-2-Sat and Max-k-VC problems can be calculated as the minimum of the following function

$$\alpha_{2sat}^{cc}(q) = \frac{1 - \Gamma_{\overline{\rho}}(1-q)}{2q}, \quad q \in (0, 1/2),$$

where $\overline{\rho} = -q/(1-q)$. Numerical experiments show that $\alpha_{2sat}^{cc} \approx 0.929$, and that the minimum is attained at $q \approx 0.365$.

Again we have that the corresponding hardness expression from Theorem 18 satisfies $\beta_{vc}^{cc}(q, -q/(1-q)) = \alpha_{2sat}^{cc}(q)$, implying (under the assumption on worst-case configurations) that the algorithm for CC-Max-2-Sat of Raghavendra and Tan is optimal.

5 Conclusion and Some Open Questions

We studied some of the cardinality constrained 2-CSPs, and assuming the Unique Games Conjecture derived hardness results which show that approximation ratios achieved by the algorithm described in [30] are optimal for CC-Max-2-Sat (and its special case Max-k-VC) and CC-Max-2-Lin (and its special case CC-Max-Cut). It would be interesting to derive UG-hardness for related CC-Max-CSPs of arity 2, most interestingly for the Max-k-DS problem. While super-constant hardness for Max-k-DS is currently known under the closely related Small-Set Expansion Hypothesis [29], it is not yet known whether the UGC implies hardness of Max-k-DS.

We also think it would be valuable to study whether we can achieve better approximation ratios or derive stronger hardness results for CC-Max-2-CSP with fixed values of the cardinality constraint q. Can the hardness curves of Theorem 17 and Theorem 18 depicted in Figure 1 be matched algorithmically for every q?

Another interesting research direction would be to come up with hardness results for some other well-know Max-CSPs like Max-3-Sat, or even more ambitiously to extend the results of Raghavendra [28] and obtain tight hardness for all cardinality-constrained Max-CSPs.

References

1. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and Hardness of Approximation Problems. In *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*, pages 14–23, 1992. doi:10.1109/SFCS.1992.267823.
2. Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs; A New Characterization of NP. In *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*, pages 2–13, 1992. doi:10.1109/SFCS.1992.267824.
3. Per Austrin. Balanced max 2-sat might not be the hardest. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 189–197, 2007. doi:10.1145/1250790.1250818.
4. Per Austrin, Siavosh Benabbas, and Konstantinos Georgiou. Better Balance by Being Biased: A 0.8776-Approximation for Max Bisection. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 277–294, 2013. doi:10.1137/1.9781611973105.21.
5. Per Austrin, Subhash Khot, and Muli Safra. Inapproximability of Vertex Cover and Independent Set in Bounded Degree Graphs. *Theory of Computing*, 7(1):27–43, 2011. doi:10.4086/toc.2011.v007a003.
6. Markus Bläser and Bodo Manthey. Improved Approximation Algorithms for Max-2SAT with Cardinality Constraint. In *Algorithms and Computation, 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, November 21-23, 2002, Proceedings*, pages 187–198, 2002. doi:10.1007/3-540-36136-7_17.
7. Joshua Brakensiek, Sivakanth Gopi, and Venkatesan Guruswami. CSPs with Global Modular Constraints: Algorithms and Hardness via Polynomial Representations. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:13, 2019. URL: https://eccc.weizmann.ac.il/report/2019/013.
8. Andrei A. Bulatov. A Dichotomy Theorem for Nonuniform CSPs. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.
9. Andrei A. Bulatov and Dániel Marx. The complexity of global cardinality constraints. *Logical Methods in Computer Science*, 6(4), 2010. doi:10.2168/LMCS-6(4:4)2010.
10. Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional Hardness for Approximate Coloring. *SIAM J. Comput.*, 39(3):843–873, 2009. doi:10.1137/07068062X.
11. Uriel Feige and Michael Langberg. The RPR^2 rounding technique for semidefinite programs. *J. Algorithms*, 60(1):1–23, 2006. doi:10.1016/j.jalgor.2004.11.003.
12. Alan M. Frieze and Mark Jerrum. Improved Approximation Algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica*, 18(1):67–81, 1997. doi:10.1007/BF02523688.
13. Michel X. Goemans and David P. Williamson. .879-approximation algorithms for MAX CUT and MAX 2SAT. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 422–431, 1994. doi:10.1145/195058.195216.
14. Venkatesan Guruswami and Euiwoong Lee. Complexity of Approximating CSP with Balance / Hard Constraints. *Theory Comput. Syst.*, 59(1):76–98, 2016. doi:10.1007/s00224-015-9638-0.
15. Venkatesan Guruswami, Rajsekar Manokaran, and Prasad Raghavendra. Beating the Random Ordering is Hard: Inapproximability of Maximum Acyclic Subgraph. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 573–582, 2008. doi:10.1109/FOCS.2008.51.

16 Eran Halperin and Uri Zwick. A unified framework for obtaining improved approximation algorithms for maximum graph bisection problems. *Random Struct. Algorithms*, 20(3):382–402, 2002. doi:10.1002/rsa.10035.

17 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

18 Thomas Hofmeister. An Approximation Algorithm for MAX-2-SAT with Cardinality Constraint. In *Algorithms - ESA 2003, 11th Annual European Symposium, Budapest, Hungary, September 16-19, 2003, Proceedings*, pages 301–312, 2003. doi:10.1007/978-3-540-39658-1_29.

19 Howard J. Karloff and Uri Zwick. A 7/8-Approximation Algorithm for MAX 3SAT? In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 406–415, 1997. doi:10.1109/SFCS.1997.646129.

20 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 767–775, 2002. doi:10.1145/509907.510017.

21 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal Inapproximability Results for MAX-CUT and Other 2-Variable CSPs? *SIAM J. Comput.*, 37(1):319–357, 2007. doi:10.1137/S0097539705447372.

22 Subhash Khot and Oded Regev. Vertex Cover Might be Hard to Approximate to within 2-ε. In *18th Annual IEEE Conference on Computational Complexity (Complexity 2003), 7-10 July 2003, Aarhus, Denmark*, page 379, 2003. doi:10.1109/CCC.2003.1214437.

23 Michael Lewin, Dror Livnat, and Uri Zwick. Improved Rounding Techniques for the MAX 2-SAT and MAX DI-CUT Problems. In *Integer Programming and Combinatorial Optimization, 9th International IPCO Conference, Cambridge, MA, USA, May 27-29, 2002, Proceedings*, pages 67–82, 2002. doi:10.1007/3-540-47867-1_6.

24 Pasin Manurangsi. Almost-polynomial ratio ETH-hardness of approximating densest k-subgraph. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 954–961, 2017. doi:10.1145/3055399.3055412.

25 Pasin Manurangsi. A Note on Max k-Vertex Cover: Faster FPT-AS, Smaller Approximate Kernel and Improved Approximation. In *2nd Symposium on Simplicity in Algorithms, SOSA@SODA 2019, January 8-9, 2019 - San Diego, CA, USA*, pages 15:1–15:21, 2019. doi:10.4230/OASIcs.SOSA.2019.15.

26 Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. *Ann. of Math. (2)*, 171(1):295–341, 2010. doi:10.4007/annals.2010.171.295.

27 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. URL: http://www.cambridge.org/de/academic/subjects/computer-science/algorithmics-complexity-computer-algebra-and-computational-g/analysis-boolean-functions.

28 Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 245–254, 2008. doi:10.1145/1374376.1374414.

29 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 755–764. ACM, 2010. doi:10.1145/1806689.1806792.

30 Prasad Raghavendra and Ning Tan. Approximating CSPs with global cardinality constraints using SDP hierarchies. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 373–387, 2012. URL: http://portal.acm.org/citation.cfm?id=2095149&CFID=63838676&CFTOKEN=79617016, doi:10.1137/1.9781611973099.33.

31 Ran Raz. A Parallel Repetition Theorem. *SIAM J. Comput.*, 27(3):763–803, 1998. doi:10.1137/S0097539795280895.

32 Thomas J. Schaefer. The Complexity of Satisfiability Problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, STOC '78, pages 216–226, New York, NY, USA, 1978. ACM. doi:10.1145/800133.804350.

33 Maxim Sviridenko. Best Possible Approximation Algorithm for MAX SAT with Cardinality Constraint. *Algorithmica*, 30(3):398–405, 2001. doi:10.1007/s00453-001-0019-5.

34 Yinyu Ye. A .699-approximation algorithm for Max-Bisection. *Math. Program.*, 90(1):101–111, 2001. doi:10.1007/PL00011415.

35 Dmitriy Zhuk. A Proof of CSP Dichotomy Conjecture. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 331–342, 2017. doi:10.1109/FOCS.2017.38.

Robust Appointment Scheduling with Heterogeneous Costs

Andreas S. Schulz
Technische Universität München, Germany
andreas.s.schulz@tum.de

Rajan Udwani
Columbia University, New York, NY, USA
rudwani@alum.mit.edu

Abstract

Designing simple appointment systems that under uncertainty in service times, try to achieve both high utilization of expensive medical equipment and personnel as well as short waiting time for patients, has long been an interesting and challenging problem in health care. We consider a robust version of the appointment scheduling problem, introduced by Mittal et al. (2014), with the goal of finding simple and easy-to-use algorithms. Previous work focused on the special case where per-unit costs due to under-utilization of equipment/personnel are homogeneous i.e., costs are linear and identical. We consider the heterogeneous case and devise an LP that has a simple closed-form solution. This solution yields the first constant-factor approximation for the problem. We also find special cases beyond homogeneous costs where the LP leads to closed form optimal schedules. Our approach and results extend more generally to convex piece-wise linear costs.

For the case where the order of patients is changeable, we focus on linear costs and show that the problem is strongly NP-hard when the under-utilization costs are heterogeneous. For changeable order with homogeneous under-utilization costs, it was previously shown that an EPTAS exists. We instead find an extremely simple, ratio-based ordering that is 1.0604 approximate.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Discrete optimization; Theory of computation → Scheduling algorithms

Keywords and phrases Appointment scheduling, approximation algorithms, robust optimization

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.25

Category APPROX

Funding *Andreas S. Schulz*: Alexander von Humboldt Foundation (with funds from BMBF)
Rajan Udwani: ONR Grant N00014-17-1-2194, NSF Grant CMMI 1636046

Acknowledgements The authors would like to thank James B. Orlin for helpful discussions.

1 Introduction

Consider the problem of scheduling appointments in service operations where customers are served sequentially by a single server. Service times of customers are uncertain, and we wish to assign time slots for serving the customers in advance. An important practical setting where this problem arises everyday is in health care services, where there are numerous instances that require efficient scheduling of appointments, such as scheduling outpatient appointments in primary care and specialty clinics, scheduling surgeries for operating rooms, or appointments for MRI scans. Often in these settings the order in which patients undergo the procedure is known in advance. Then a day before the procedures, a hospital manager determines planned start times and how much time to allot to each procedure. If the manager allots too small an interval to a procedure, it could easily go overtime and delay the next procedure. The inconvenience and costs resulting from such a delay are referred to as the


© Andreas S. Schulz and Rajan Udwani;
licensed under Creative Commons License CC-BY



overage cost for that procedure. On the contrary, if the manager assigns a very large interval, then the procedure will likely end much sooner. In this case the hospital incurs an *underage* cost as the equipment and personnel may be left idle until the scheduled start of the next procedure. We would like to design an appointment schedule that can achieve the desired trade-off between overage and underage costs.

In an influential survey on outpatient scheduling, Cayirli and Veral [6] concluded that the "biggest challenge for future research will be to develop easy-to-use heuristics". Traditionally, most models for the problem are stochastic in nature and assume distributional knowledge of the uncertain service times. While such models are very powerful, estimating distributions accurately can require large amounts of data. There are many settings where such data is in fact available but, in applications related to health care appointment scheduling there is evidence (ref. [17, 18, 8]) that the amount of data available by surgery types, let alone by surgery types and surgeons, is extremely limited. Moreover, computing an objective function that involves finding the expectation of a high dimensional non-linear function can be computationally burdensome. Further, Mak et al. [18] and Mittal et al. [20] point out that methods proposed for solving stochastic models often use sophisticated subroutines, such as submodular function minimization and Monte-Carlo techniques, that may not always be practical. Robust optimization (e.g., [4, 5, 3]) offers an alternative framework to address uncertainty that avoids distributional assumptions. Instead, it uses deterministic uncertainty sets and performs a worst-case analysis w.r.t. to the uncertainty. This addresses the problems arising out of insufficient data and often these models are more tractable.[1] Indeed, for the case of appointment scheduling, Mittal et al. [20] introduced the following robust formulation for the problem.

Referring to the procedures/services in any given context simply as "jobs", in the robust appointment scheduling problem we are given n jobs with uncertain service times $\{t_i\}$. Assume that the time t_i for job i can be anything in the range $[p_i - \hat{\delta}_i, p_i + \delta_i]$. The range is known to us but the value of t_i can be chosen adversarially. Our task is to propose an appointment start time for each job. A job will be available to process after this start time and jobs will be served in order of increasing index i. The order in which jobs are processed is fixed a priori. Due to uncertainty in service times, a job may have to wait to be processed after its appointed start time. The overage cost incurred per unit of wait is given by $o_i \geq 0$. Similarly, $u_i \geq 0$ is the per-unit underage cost associated with job i. Given a set of appointment start times, an adversary chooses the worst possible instantiation of service durations t_i i.e., one that maximizes the overall overage/underage cost. We seek start times to minimize this worst case total cost.

Mittal et al. [20] found closed-form optimal solutions for the special case of homogeneous underage costs i.e., $u_i = u$ for all jobs i. We study more general cases[2], where a direct generalization of their solution can be arbitrarily bad. Our central goal here is to find simple, easy to implement, and theoretically well founded algorithms. In the following we summarize our contributions and discuss closely related work.

[1] Sometimes at the expense of being too conservative.

[2] Arguably, just as processing times and uncertainty vary across procedures, so can personnel and equipment. Thus, in general the per-unit underage costs will vary across procedures. Moreover, it is also reasonable to assume that the per-unit cost changes with the amount of underage/overage. For instance, keeping patients waiting for another unit of time becomes less and less desirable with every unit of delay. This prompts the study of piece-wise linear overage and underage costs.

Robust Appointment Scheduling (RAS). We explore the structure of optimal appointment schedules and find several properties that all optimal schedules must satisfy. Somewhat surprisingly, we find that in every optimal appointment schedule, the case of all jobs underaged as well as the case of all jobs overaged are always worst-cases for the adversary's problem. Casting these in the form of linear (in)equalities gives an LP, resulting in a 2-approximation. Further simplification yields a closed-form solution to the LP that could be implemented even on a spreadsheet. More generally, when the per-unit costs are allowed to change with the amount of overage/underage and the cost functions are piece-wise linear and convex (and non-decreasing), we are able to generalize these properties to obtain a 2-approximation for the problem via solving a min-cost flow problem on a graph with convex piece-wise linear arc costs.

Previously, Mittal et al. [20] examined the special case of identical underage costs and, quite remarkably, found a closed-form optimal solution in that case. While their analysis was tailored for the special case, our different approach allows us to characterize exactly when such a result holds, and we discover more general conditions under which there exists a closed-form optimal solution to the problem. We also show a similar extension for the case of convex piece-wise linear costs, where we give a simple iterative algorithm that gives an optimal allocation under additional assumptions on the cost functions.

Robust Appointment Scheduling and Ordering (RASO). While the order of patients is often fixed in advance, there are instances where patient order is flexible and simultaneously part of the decision making [6]. When underage costs are identical, Mittal et al. [20] established a key connection between RASO and the theory of scheduling jobs on a single machine to minimize the sum of a weighted nonlinear (concave) cost function of the completion times [19, 12, 25]. Focusing on the special case, we exploit this connection further to give an extremely practical ratio based ordering policy, inspired by Smith's rule [24]. We call it the Customized-Smith rule (short C-Smith) and show that it is 1.0604-approximate. Further, we find an algorithm that is as good as C-Smith on every instance, but is also locally optimal w.r.t. pairwise swaps of neighboring jobs. Previously, Mittal et al. [20] showed that using the EPTAS for the related min-sum scheduling problem given by Stiller and Wiese [25], yields an EPTAS for RASO. The runtime of the EPTAS scales as $O(2^{1/\epsilon^7}n)$, and its implementation complexity makes it hard to use in practice. Mittal et al. [20] offered Smith's rule as a practical alternative, with approximation ratio 1.137 due to a result in Höhn and Jacobs [12]. Finally, for the general case (arbitrary underage costs), we show the problem is strongly NP hard. We also briefly discuss the inefficacy of *list ordering* heuristics for the problem and give a heuristic with a matching lower bound.

Other related work. The existing literature on appointment scheduling is quite diverse and, in addition to the robust model we discussed above, includes numerous results on stochastic optimization models, queueing models, as well as distributionally robust optimization models. We only give a very brief review here and refer the reader to [6] for a broad overview of the challenges in scheduling appointment systems in health care, and to [18, 20] for more comprehensive surveys on past work. Starting with more recent work, Jiang et al. [14] consider a distributionally robust model and propose a MINLP formulation that can handle random no-shows. Mak et al. [18] consider a distributionally robust model given marginal moments for job durations, and formulate the problem as a tractable conic program. For the case of flexible job order, they show that under certain assumptions a simple variance based ordering can be optimal. Prior to [18], Kong et al. [16] first considered a distributionally

robust formulation but with cross-moments as opposed to marginal moments. They formulate the problem as a copostive cone program and propose a tractable SDP relaxation. Wang [26] modeled the problem using a queueing model where the processing times of the jobs are i.i.d. exponential and new jobs may be released over time. Wang [27] generalized the model to allow for different mean processing times for jobs. In a different direction, the problem was modeled as a two-stage stochastic linear program in Denton and Gupta [8]. For this problem, Robinson and Chen [23] compute near-optimal solutions using a Monte-Carlo integration technique. Kaandorp and Koole [15] considered a local search algorithm and proved that it converges to an optimal solution. Another stochastic model was introduced by Green et al. [11]. They considered the problem of outpatient appointment scheduling with emergency services and modeled it as a dynamic stochastic control problem. In highly influential work, Begen and Queyranne [2] considered a discrete stochastic model (where job durations are integer random variables with finite support). They showed that the problem reduces to an instance of submodular function minimization, under certain assumptions on the per-unit costs. Begen et al. [1] extended the idea, proving a similar result for a data-driven discrete stochastic model. Ge et al. [10] further extended the result of Begen and Queyranne [2], to the setting of convex piece-wise linear per-unit costs.

Assumptions on underage costs. Previous work on the stochastic and distributionally robust models discussed above, often assumes an upper bound on the variation in underage costs. Formally, consider jobs that are indexed in the order they are scheduled. So we have jobs $i \in \{1, \ldots, n\}$,with per unit costs u_i, o_i. Now for instance, the result by Begen and Queyranne [2], assumes that there exists values $\alpha_i \leq o_i$ such that $u_{i+1} \leq u_i + \alpha_i - \alpha_{i+1}$ for every $i \in [n-1]$. Descending underage costs for example, satisfy this condition. More generally, [14, 18, 16, 10] assume that $u_{i+1} \leq o_i + u_i$ for every $i \in [n-1]$ (or that $u_i = u$ for all i). In effect, these assumptions are satisfied as long as the underage costs don't *increase* too sharply.

In contrast, we give a closed form optimal allocation for instances where the u_i values do not *decrease* too sharply. For example, our results yield a closed form optimal solution for the case of non-decreasing underage costs. This includes the case $u_{i+1} \geq u_i + o_i \, \forall i \in [n-1]$, converse of the condition assumed in some previous work. If the per-unit underage costs are decreasing, we show that as long as the per-unit values are "large enough", our LP still leads to a closed form optimal allocation. In practical terms, it is quite possible that the underage costs could be increasing or decreasing, preventing a direct comparison of our assumptions with past work. In fact, for the RASO problem where one has the flexibility to choose the order of jobs, an optimal order can have arbitrarily varying underage costs. From a technical viewpoint, our results complement previous work, indicating that perhaps the robust model is more tractable in instances where other models are not, and vice versa.

Overview of the paper. In Section 2, we introduce notation and formally state the problem. We also make certain observations that simplify the problem w.l.o.g.. Then in Section 3, we explore the appointment scheduling problem under fixed order. We first give our LP based approximation for general underage costs in Section 3.1. And later in Section 3.2, we tighten our LP formulation to obtain optimal closed form solutions under additional assumptions on the costs. In Section 4, we consider the problem of jointly finding an optimal order as well as appointment schedule (RASO). We show that the problem is NP hard in general and discuss the limitations of a classes of simple heuristics in Section 4.1. In Section 4.2, we focus on the ordering and scheduling problem in the special case of homogeneous underage costs. Here we discuss some disadvantages of existing results and propose two new heuristics to tackle some of the issues. Finally, we conclude with some open problems in Section 5.

2 Notation & Preliminaries

We start with a description of RAS, and define additional notation required for RASO in Section 4. Recall, we have n jobs to be served in order of increasing index $i \in \{1, \ldots, n\}$. Service times are uncertain and modeled via a box uncertainty set: job i takes time t_i in the range $[p_i - \hat{\delta}_i, p_i + \delta_i]$. Here we assume that $0 \leq \hat{\delta}_i \leq p_i$ as well as $\delta_i \geq 0$, for all jobs i. We denote by $o_i \geq 0$ and $u_i \geq 0$ the per-unit overage cost and per-unit underage cost of job i, respectively. More generally, we represent overage and underage costs as *non-decreasing* functions $o_i(\cdot), u_i(\cdot)$ respectively. The case of constant per-unit costs is then given by $o_i(x) = o_i \cdot x$ and $u_i(x) = u_i \cdot x$. We would like to appoint job start times, $\{A_i\}_i$, such that every job arrives at its appointed time and is served as soon as possible after. Given $\{A_i\}_i$, consider an arbitrary instance of service times $\{t_i\}_i$, and let $\{C_i\}_i$ denote the completion times of jobs.[3] If job i is delayed and ends after the appointed start time for job $i+1$, we incur overage cost $o_i(C_i - A_{i+1})$. Similarly, if job i ends before the appointed start time for job $i+1$, we incur underage cost $u_i(A_{i+1} - C_i)$. Therefore, the cost due to job i is $\max\{o_i(C_i - A_{i+1}), u_i(A_{i+1} - C_i)\}$. The RAS problem can now be stated as follows,

$$\min_{\{A_i\}_i} \underbrace{\left(\max_{t_i \in [p_i - \hat{\delta}_i, p_i + \delta_i] \; \forall i \in [n]} \sum_{i=1}^{n} \max\{o_i(C_i - A_{i+1}), u_i(A_{i+1} - C_i)\} \right)}_{\textit{Adversary's problem, given appointment times } A_i}. \tag{1}$$

The adversary's problem in (1), finds a *worst possible profile/instance of service times* $T = (t_i) = (t_1, \ldots, t_n)$, maximizing the cost, given the schedule $\{A_i\}_i$. Let $c(T, A, i) = \max\{o_i(C_i(T) - A_{i+1}), u_i(A_{i+1} - C_i(T))\}$ denote the cost of job i given allocation $\{A_i\}_i$ and time profile T. When the allocation is clear from the context, we use the shorthand $c(T, i)$. Let $c(T) = \sum_i c(T, i)$ denote the total cost due to profile T. It is not difficult to see that an optimal allocation needs to allocate at least $p_i - \hat{\delta}_i$ time for job i, so we let $p_i - \hat{\delta}_i = 0$ for every job i, w.l.o.g. (also in Lemma 6 of [20]). To simplify notation henceforth, we let service times be in the range $[0, \Delta_i]$, where $\Delta_i = \hat{\delta}_i + \delta_i$. Now, given appointment schedule $\{A_i\}_i$, consider equivalent variables x_i that represent the duration allocated for a job. We have, $x_i = A_{i+1} - A_i$, which is the difference between the start times of job $i+1$ and job i. Equivalently, $A_i = \sum_{j=1}^{i-1} x_j$ for $i \geq 2$ and $A_1 = 0$, w.l.o.g. We call $\{x_i\}_i$, and sometimes by abuse of terminology $\{A_i\}_i$, the *allocation*. Note that in our model, job n is cost free since there is no appointment succeeding it ($A_{n+1} = \infty$). W.l.o.g. we may assume that job n also suffers from overage and underage based on its assigned end time (assign a dummy job to succeed job n). Note, if $u_i(\cdot) = 0$, we allot a very large time slot for job i and jobs on different sides of the slot become independent. Therefore, we let $u_i(\cdot) \neq 0$ for all i, w.l.o.g.. Also, if $o_n(\cdot) = 0$ we can assume that $x_n = 0$ and in fact ignore job n, therefore we also let $o_n(\cdot) \neq 0$, w.l.o.g.. To coalesce, the assumptions we have made w.l.o.g., so far:

(i) Service time for job i lies in $[0, \Delta_i]$; (ii) for every i, $u_i(\cdot) \neq 0$ ($u_i > 0$ in the constant per-unit case); and (iii) $o_n(\cdot) \neq 0$ ($o_n > 0$ in the constant per-unit case).

Finally, we say job i is *underaged* if it ends on or before time A_{i+1}, and *overaged* otherwise. However, if $\Delta_i = 0$ and job i starts/ends at A_{i+1}, we consider it to be both overaged and underaged (for technical reasons). Further, if job i takes time Δ_i to be served, we say it runs for *maximum time*. Note that if a job i is underaged in some worst-case $T = (t_i)$, then w.l.o.g., $t_i = 0$, as the underaged costs $u_i(\cdot)$ are non-decreasing functions for every i (similar

[3] $C_i = \max\{C_{i-1}, A_i\} + t_i$ for all $i \geq 1$ with $C_0 = A_1 = 0$, w.l.o.g.

to Lemma 11 of [20]). Observe also that if a job i is underaged in some worst-case time profile, and it has zero cost, then i must end at A_{i+1} and take zero time. Otherwise, we have a strictly worse case by underaging i (simply reduce the time taken by i by some small $\epsilon > 0$).

Finally, let $S_i = \{i, i+1, \ldots, n\} = \{j | j \geq i\}$ be the subset of the last $n - i + 1$ jobs in the schedule. Let x_i^S denote the optimal time allocation for job i when considering only jobs in a subset S that contains i. When $S = [n]$ we often use shorthand x_i. The next section discusses a result from previous work and offers some intuition for our treatment of the general case that follows subsequently in Section 3.

2.1 Closed Form Optimal Solution of Mittal et al. [20]

Consider a single job, with per unit costs u, o and maximum time Δ. If this job is alloted time duration x, then the worst case cost is given by $\max\{u(x), o(\Delta - x)\}$, minimized at $x = \Delta\frac{o}{o+u}$ (recall, we assume that a dummy job always follows the last job, so the last job incurs overage cost for delays). Mittal et al. [20] showed that for constant per-unit underage costs, $u_i = u$ for every job $i \in [n]$, this formula generalizes and the optimal allocation is given simply by, $x_i = \Delta_i \frac{o(i)}{o(i)+u}$. Here $o(i)$ is the sum $\sum_{j=i}^{n} o_j$, of the per unit overage costs of the jobs succeeding job i. So each job is alloted a fraction of its maximum service time Δ_i, with a smaller fraction for larger values of u (to prevent large underage costs). Further, earlier jobs are alloted a larger fraction of their maximum time to prevent a large cascade of delays for jobs serviced later on.

Now, consider a natural generalization of this formula for heterogeneous underage costs given by, $x_i = \Delta_i \frac{o(i)}{o(i)+u_i}$. Unfortunately, this can be a suboptimal allocation even for two jobs with reasonable parameter values, and in general an arbitrarily bad approximation. For example, consider two jobs that are almost identical except that job 2 (which is scheduled later) has a small underage cost. Specifically, let $\Delta_1 = \Delta_2 = 1$ and suppose per unit costs $o_1 = o_2 = 1$ and $u_1 = M \geq 2$, but $u_2 = 1$. The allocation given by the formula sets x_1 to $\frac{2}{M+2}$ and x_2 to $\frac{1}{2}$. The worst case cost of this allocation is attained when both jobs are underaged or both overaged (more on this later), and equals $\frac{2M}{M+2} + \frac{1}{2}$. For large M, this value approaches $\frac{5}{2}$. Now instead, consider the following allocation, $y_1 = 0$ and $y_2 = \Delta_2 \frac{o_2}{o_2+u_2} + \Delta_1 \frac{o(1)}{o_2+u_2} = \frac{1}{2} + 1$ (we show later in Section 3.2, that this is in fact an optimal allocation). It is easily checked that the worst case cost is $\frac{3}{2}$ for this allocation. More generally, if we set o_1 and u_2 to a small value ϵ, the first allocation becomes arbitrarily bad. Intuitively, this demonstrates that if jobs that are later in the order have small underage costs, it is beneficial to allocate larger time to these later jobs to buffer for delays from earlier jobs. When this is not the case (such as for homogeneous underage costs), it is better to instead buffer by allocating larger fractions to jobs that are earlier in the order.

3 Robust Appointment Scheduling

3.1 Heterogeneous Per-unit Underage Costs

In this section, we focus on the case of constant per-unit costs o_i, u_i. The key behind our results lies in finding useful properties satisfied by optimal allocations. Towards that end, it will be instrumental to understand the worst cases (solutions to the adversaries' problem) for optimal allocations. Recall the adversary's objective: given $\{A_i\}_i$, maximize cost $c(T, A)$ over all possible time profiles T allowed by the uncertainty set. It turns out that for an arbitrary allocation, there could be a unique worst-case where some jobs are overaged while

others are underaged, and for some jobs the service time t_i is neither 0 nor Δ_i [4]. It turns out though, that for any given allocation the adversary's problem can be reduced to an instance of finding longest paths on a directed acyclic graph with $n+1$ nodes [21], and therefore can be solved in polynomial time. However, using this fact to find an optimal allocation does not obviously lead to a tractable problem. Instead, here we shall find properties in the form of linear inequalities that every optimal allocation satisfies. Using these, we formulate an LP relaxation for the problem and show that an optimal solution to the LP is 2-approximate.

As a natural next step, we then look for more structure to further tighten the formulation. Surprisingly, we find that the adversary's problem given an *optimal allocation* for problem (1) is easily solved; in every optimal allocation, $t_i = \Delta_i$ for every i (all jobs taking maximum time) and $t_i = 0$ for all i (all jobs taking zero time) are worst-cases. Combining this with other structural insights, in Section 3.2 we propose a strengthened formulation that leads to closed form optimal solutions under some assumptions on underage costs. In the lemmas that follow we introduce two properties of optimal allocations, leading to the first LP formulation.

▶ **Lemma 1.** *For every optimal allocation* $\{x_i\}_i$, $\sum_{j=1}^{k} x_j \leq \sum_{j=1}^{k} \Delta_j$ *for all* $k = 1, \ldots, n$.

Proof. Given an optimal allocation $\{x_i\}_i$, let $\delta_k = \sum_{j=1}^{k} x_j - \sum_{j=1}^{k} \Delta_j$, for $k \in [n]$. Suppose, $\delta_k > 0$ for some k, and let k_0 be the smallest such k. Given this, notice that even if the first k_0 jobs take maximum time, job k_0 can never be overaged. Therefore, decreasing x_{k_0} decreases the underage cost of job k_0 ($u_{k_0} > 0$) and thus, also the worst-case cost. So the allocation $\{x'_i\}_i$ where, $x'_{k_0} = x_{k_0} - \delta_{k_0}$ and $x'_i = x_i$ for all other i is clearly a better allocation, contradiction. This lemma also follows as a direct corollary of Lemma 5 stated later on. ◀

▶ **Lemma 2.** *Given an allocation* $\{x_i\}_i$ *where* $\sum_{j=1}^{i} x_j \leq \sum_{j=1}^{i} \Delta_j$ *for all* i*, and a time profile* $T = (t_i)$*. The cost* $c(T)$ *of profile* T *is at most* $\sum_{i=1}^{n} o(i)(\Delta_i - x_i) + \sum_{i=1}^{n} u_i x_i$.

Proof. For any given time profile T, job i is either underaged with $c(T,i) \leq u_i x_i$ or overaged with cost at most $o_i \sum_{j \leq i}(\Delta_j - x_j)$. The latter follows from the fact that $C_i \leq \sum_{j \leq i} \Delta_j$ since $\sum_{j=1}^{k} x_j \leq \sum_{j=1}^{k} \Delta_j$ for all $k \in [i]$. Therefore, $\sum_{i=1}^{n} c(T,i) \leq \sum_{i=1}^{n} o_i \sum_{j=1}^{i}(\Delta_j - x_j) + \sum_{i=1}^{n} u_i x_i$. Rearranging the sum we have, $\sum_{i=1}^{n} o_i \sum_{j=1}^{i}(\Delta_j - x_j) = \sum_{j=1}^{n}(\Delta_j - x_j) o(j)$. ◀

Using the above results, the following LP now gives us an extremely simple approximation for the general problem that could be easily implemented on most systems.

$$\textbf{LP-1:} \quad \min \quad \sum_{j=1}^{n} (u_j - o(j)) y_j$$
$$\text{s.t.} \quad \sum_{j=1}^{k} (y_j - \Delta_j) \leq 0 \quad \forall k \in [n]; \tag{2}$$
$$y_j \geq 0 \quad \forall j \in [n].$$

▶ **Theorem 3.** *LP-1 is a (tight) 2-approximation, and the following is an optimal solution to LP-1. Define* $m_i = \arg\min_{j \geq i}(u_j - o(j))$*, then for every* i*,*

$$y_i = \begin{cases} 0 & \text{if } u_i - o(i) \geq 0 \text{ or } i \neq m_i \\ \sum_{j | m_j = i} \Delta_j & \text{otherwise.} \end{cases}$$

[4] This is in contrast to Lemma 9 in [20] for the special case of $u_i = u \, \forall i$.

Proof. The proof of optimality for the proposed solution is easy to verify. To see the guarantee let us re-write the LP-1 objective as,

$$\min \sum_j u_j y_j + \sum_j o(j)(\Delta_j - y_j).$$

From constraints (2) and Lemma 2, this is an upper bound on the worst-case cost of any feasible solution to LP-1. Let $\{y_i^*\}$ denote an optimal solution to the LP. Lemma 1 implies that every optimal allocation $\{x_i\}$ is a feasible solution to LP-1. Further, $\sum_j u_j x_j$ denotes the cost when all jobs take zero time and $\sum_j o(j)(\Delta_j - x_j)$ is the cost of overaging all jobs with each job taking maximum time (Δ_i). Therefore, the worst-case cost of an optimal allocation, denoted OPT, is at least $\max\{\sum_j u_j x_j, \sum_j o(j)(\Delta_j - x_j)\}$ (we show later that these two costs are both in fact, equal to OPT). Therefore,

$$\sum_j u_j y_j^* + \sum_j o(j)(\Delta_j - y_j^*) \leq \sum_j u_j x_j + \sum_j o(j)(\Delta_j - x_j) \leq 2\ OPT.$$

For a tight instance, consider two jobs $\{1, 2\}$ with $\Delta_1 = \epsilon \to 0$, $u_1 = 1/\epsilon, o_1 = 1/\epsilon - (1+\epsilon)$ and $\Delta_2 = u_2 = o_2 = 1$. Therefore, $\frac{u_2}{u_2+o_2} = 0.5 \lessapprox \frac{u_1}{u_1+o(1)}$. Consider the allocation $x_2 = 0.5$ and $x_1 = \epsilon\frac{o(1)}{u_1+o(1)} \approx \epsilon/2$. It is easy to see that the worst-case of the allocation is when both jobs are underaged (overaged) and hence the cost of this allocation is $u_1x_1 + u_2x_2 \approx 1$. Now, consider the solution $y_2 = 0.5 + \frac{\Delta_1 o(1)}{u_2+o_2} \approx 1$ and $y_1 = 0$. This is an optimal solution to the LP and the worst-case occurs when job 2 is underaged and job 1 is overaged. The worst-case cost of this allocation is $o_1\Delta_1 + u_2(y_2 - \Delta_1) \approx 2$. ◀

Beyond Constant Per-unit Costs

Suppose instead of scalar costs o_i, u_i, we have non-decreasing, piece-wise linear and convex cost functions $o_i(\cdot), u_i(\cdot)$. Then it is easy to check that Lemma 1 still holds. Similar to Lemma 2, we have that given an allocation $\{x_i\}_i$, satisfying Lemma 1, the cost $c(T)$ for any profile T is at most,

$$\sum_i u_i(x_i) + \sum_i o_i\Big(\sum_{j\leq i}(\Delta_j - x_j)\Big).$$

Now the problem of minimizing this objective subject to the linear constraints in LP-1, is a min cost flow problem with arc costs given by the overage and underage cost functions. More specifically, consider a directed graph with $n+1$ nodes and $2n$ edges, where there is a directed edge from node i to node $n+1$ with cost $u_i(\cdot)$ and an edge $i \to i+1$ with cost $o_i(\cdot)$, for every $i \in [n]$. Finally, there is another edge from n to $n+1$ with cost $o_n(\cdot)$. Finally, each node i has a supply of Δ_i and node $n+1$ is a sink with demand $\sum_i \Delta_i$. Given a feasible flow in this graph, the flow on edge $(i \to n+1)$ gives the time allocation for job i, and vice versa. Now, if the costs are piece-wise linear and convex, we have from the algorithm by Pinto and Shamir [22] for min flows with convex piece-wise linear costs, that the problem can be solved in polynomial (in n and the maximum number of pieces in the cost functions) time. The optimal solution to this problem is a 2-approximation, and the analysis closely resembles the proof of Theorem 3 (details deferred to full version).

3.2 Optimal Solution to RAS for Special Cases

Let us now investigate additional properties with the goal of strengthening our LP. We start by proving our claim from earlier – all jobs underaged (taking 0 time) and all jobs overaged (taking maximum time) are worst-cases for optimal allocations. We break down the proof

into smaller parts. The first lemma is very useful and appears often in proofs of other lemmas. The key insight behind the lemma is simple – when all jobs overaged is a worst-case, if the first job is forced to start late, then the case of all jobs overaged suffers maximum increase in cost, and is thus still a worst case.

▶ **Lemma 4.** *Given an arbitrary allocation $\{A_i\}_i$, where $t_i = \Delta_i$ for all i is a worst-case. Recall that w.l.o.g., $A_1 = 0$ and consider a modified problem for the adversary, where the first job is always forced to start at a later time t_0 instead of time $A_1 = 0$. Then, $T = (t_i)$ is also a worst-case for the modified problem.*

Proof. Let $T = (\Delta_i)$ denote the profile for all jobs taking maximum time. When job 1 starts at time 0, denote the cost of job i by $c(0, T, i)$. Since T is a worst-case by assumption, we have $c(0, T) \geq c(0, Z)$ for every profile Z. For the modified setting where job 1 starts at time $t > 0$, we claim that $c(t, Z, i) \leq c(0, Z, i) + o_i t$ for every Z. To see this, suppose job 1 starts at 0 and consider two cases: (i) i overaged in Z and (ii) i underaged in Z. In case (i), i will still be overaged when job 1 starts at time t and the completion time of i can increase by at most t. In case (ii), if i is still underaged when job 1 starts at time t, we are done. Else, i becomes overaged but the maximum overage cost is $o_i t$. Now for profile T, since all jobs are overaged when job 1 starts at time 0, $c(t, T, i) = c(0, T, i) + o_i t$. Therefore, $c(t, Z) \leq \sum_i (c(0, Z, i) + o_i t) \leq \sum_i c(0, T, i) + \sum_i o_i t = c(t, T)$. ◀

The next lemma says that we cannot have an optimal allocation where a certain job is underaged (overaged) in all the worst-case time profiles. Observe that if job i is always underaged, simply reducing the allocation x_i would give a strictly better allocation, contradicting optimality. Indeed, Lemmas 12 and 15 in [20] argue exactly this. However, that argument fails if there exists a worst-case where i is underaged with zero cost (occurs when $i - 1$ ends at A_{i+1} and $t_i = 0$, which we defined as a case of underage in Section 2). This gives rise to a subtle issue that demands a more involved argument (similarly for the case of overage). We postpone the formal proof to the full version.

▶ **Lemma 5.** *Given an optimal allocation, consider an arbitrary job i. There exists a worst-case $T = (t_i)$ where i is underaged and $t_i = 0$, as well as a worst-case where i is overaged (with some t_i that is not necessarily Δ_i).*

▶ **Lemma 6.** *In every optimal allocation, the case of all jobs underaged, i.e., $t_i = 0$ for every job i, is a worst-case.*

Proof. Let $\{A_i\}_i$ denote an optimal allocation. Lemma 5 implies there is a worst-case where job 1 takes zero time. We proceed via induction, assuming there is a worst-case $T = (t_i)$ where $t_i = 0$ for $i \in [k] = \{1, \ldots, k\}$, i.e., all jobs in $[k]$ are underaged. We will show there exists a worst-case where all jobs in $[k + 1]$ are underaged.

Suppose that $k + 1$ is overaged in T (otherwise we are done). Lemma 5 implies there is a worst-case, denoted $T' = (t'_i)$, where job $k + 1$ is underaged and takes zero time. Let C_{k+1} denote the completion time of job $k + 1$ in T. Clearly, $C_{k+1} > A_{k+2}$ and $k + 1$ starts at A_{k+1} in T. Similarly, let C'_{k+1} denote the start/completion time of job $k + 1$ in T'. Then, $C'_{k+1} \leq A_{k+2}$. Now consider a new profile $Z = (z_i)$, formed by a combination of T and T'. We let $z_i = t'_i$ for $i \in [k]$ and $z_i = t_i$ for $i \in \{k+2, \ldots, n\}$. Since the completion times of job k are identical in T' and Z, we set $z_{k+1} = C_{k+1} - C'_{k+1} \leq C_{k+1} - A_{k+1} = t_{k+1}$. Therefore, job $k+1$ ends at time C_{k+1} in Z. Now, observe that $\sum_{i\in[n]} c(Z, i) = \sum_{i=1}^{k} c(T', i) + \sum_{i=k+1}^{n} c(T, i)$. Since T is a worst-case, we also have $\sum_{i=1}^{k} c(T', i) + \sum_{i=k+1}^{n} c(T, i) \leq \sum_{i=1}^{n} c(T, i)$. Therefore, $\sum_{i=1}^{k} c(T', i) \leq \sum_{i=1}^{k} c(T, i)$. Now, consider another hybrid case Q, where jobs 1 to k are all

underaged and take zero time as in T and jobs $k+2$ to n are as in T'. Job $k+1$ starts/ends at A_{k+1} in Q and hence $c(T', k+1) \leq c(Q, k+1)$. Combining everything, $\sum_i c(T', i) \leq \sum_{i=1}^{k} c(T, i) + \sum_{k+1}^{n} c(T', i) \leq \sum_{i=1}^{k} c(T, i) + c(Q, k+1) + \sum_{k+2}^{n} c(T', i) = \sum_i c(Q, i)$. Hence, Q is a worst-case with jobs 1 to $k+1$ all underaged. ◀

▶ **Lemma 7.** *In every optimal allocation, the case of all jobs overaged is a worst-case. Therefore, $t_i = \Delta_i$ for all i is a worst-case.*

Proof. Let $\{A_i\}_i$ denote an optimal allocation. Observe that if there exists a worst-case where jobs $S_k = \{k, \ldots, n\}$ are all overaged for some k, then there exists a worst-case where all jobs in S_k are overaged and take maximum time. We proceed by induction and show that if there is a worst-case where every job in S_k is overaged, then there exists a worst-case where every job in S_{k-1} is overaged. For $k = n$, by Lemma 5 there is a worst-case where job n is overaged. Assume there exists a worst-case where jobs k to n are overaged, denoted as $T = (t_i)$.

Suppose that job $k-1$ is underaged in profile T (otherwise we are done). Recall that $u_{k-1} > 0$ and $t_{k-1} = 0$, since k is underaged in T. Then, since T is a worst-case profile; restricted to the subset S_k, the profile $\{t_i = \Delta_i\}_{i \geq k}$ is a worst-case with all jobs overaged, for allocation $\{A_i\}_{i \geq k}$. Now, by Lemma 5 for the set of all jobs $[n]$, there exists a worst-case where job $k-1$ is overaged, denoted $T' = (t'_i)$. Then, consider profile $Z = (z_i)$ with $z_i = t'_i$ for jobs in $S_1 - S_{k-1}$ and $z_i = \Delta_i$ for $i \in S_{k-1}$. Z is a worst-case profile since it matches the cost in T' for jobs in $S_1 - S_{k-1}$, and due to Lemma 4 the total costs for jobs in S_{k-1} can only be larger than the same total in T'. ◀

The following equation characterizes the worst-case cost of every optimal allocation $\{x_i\}_i$,

$$\sum_{i=1}^{n} o(i)(\Delta_i - x_i) = \sum_{i=1}^{n} u_i x_i. \tag{3}$$

We can already modify LP-1 by adding the equality above. However, this property alone is not sufficient to offer improved results even for identical underage costs. To that end, we introduce the following technical property.

▶ **Lemma 8.** *Given an optimal allocation $\{x_i\}_{i \in [n]}$,*

$$\sum_{j=k}^{n} (u_j + o(j)) y_j \geq \sum_{j=k}^{n} o(j)\Delta_j \quad \textit{for all } k \in [n]. \tag{4}$$

To get some intuition behind the lemma, consider the objective of minimizing $\sum_i u_i y_i$ for non-negative y_i subject to the equation (3). A greedy solution that sets $y_i = \frac{\sum_j \Delta_j o(j)}{u_i + o(i)}$ for $i = \arg\min_{j \in [n]} \frac{u_j}{u_j + o(j)}$ is optimal. Here the job with the minimum ratio $\frac{u_i}{u_i + o(i)}$, bears the entire burden of the equality (3). Lemma 8 says that this can only occur when $i = n$ and more generally, places a lower bound on the contribution to equality (3) from values y_k to y_n, for all k. The proof of the lemma is rather technical and is postponed to the full version, along with two accompanying helper lemmas. Consider now the strengthened formulation,

$$\textbf{LP-2:} \quad \min \quad \sum_{j=1}^{n} u_j y_j$$

$$\text{s.t.} \quad \sum_{j=1}^{n} (u_j + o(j)) y_j = \sum_{j=1}^{n} o(j)\Delta_j \tag{5}$$

$$\sum_{j=k}^{n} (u_j + o(j)) y_j \geq \sum_{j=k}^{n} o(j)\Delta_j \quad \forall k \in [n] \tag{6}$$

$$\sum_{j=1}^{k} (y_j - \Delta_j) \leq 0 \quad \forall k \in [n]$$

$$y_j \geq 0 \quad \forall j \in [n]$$

Clearly, every optimal allocation is a feasible solution for the LP. However, an optimal solution to the LP need not have all jobs underaged (or overaged) as worst-case, and hence need not be an optimal allocation. This is demonstrated by the example used in Theorem 3, where it is easily checked that the additional constraints given by (5) and (6) do not improve the worst-case approximation bound from earlier. However, we show that under additional assumptions LP-2 yields optimal allocations.

The recipe behind proving this is as follows: (i) Using the assumptions, show that there is actually a closed-form optimal solution to the LP. (ii) Show that for this solution, all jobs underaged and all jobs overaged are worst-cases. In particular, using this recipe for the special case of homogeneous underage costs $u_i = u$ for every i, we find that $x_i = \frac{o(i)\Delta_i}{u+o(i)}$ for all $i \in [n]$ is an optimal solution to LP-2 and an optimal allocation. More generally, we have the following.

▶ **Theorem 9.** *If $u_i \leq u_{i+1}\frac{o(i)}{o(i+1)}$ for all $i \in [n-1]$, then the allocation given by $x_i = \frac{o(i)\Delta_i}{u+o(i)}$, $i \in [n]$, is an optimal allocation.*

This generalizes and offers a different perspective on Theorem 5 in [20] (since $u_i = u$ for every i, implies $u_i \leq u_{i+1}\frac{o(i)}{o(i+1)}$ for all $i \in [n-1]$). As a direct implication of the above, we have a closed form optimal allocation if if the underage costs are non-decreasing i.e.,

$$u_i \leq u_{i+1} \quad \forall i \in [n-1].$$

This includes for instance, the case of increasing underage costs where $u_{i+1} \geq u_i + o_i$. As we discussed earlier in Section 1, this complements the assumptions made in previous work, where the common assumption is that the underage costs are not increasing too drastically and in particular that, $u_{i+1} \leq u_i + o_i$ for all $i \in [n-1]$.

Next, we claim that even if $u_i > u_{i+1}\frac{o(i)}{o(i+1)}$ for some i, there may still exist an LP optimal solution that is also an optimal allocation. This generalizes Theorem 9. A direct corollary of the result is that if underage costs are "large enough", we have a closed form optimal solution even if the costs otherwise vary arbitrarily. More specifically, if the underage costs are such that for ever pair of jobs $i-l$ and i with $u_{i-l} > u_i$, we have,

$$u_i \geq \sum_{j=1}^{l} o_{i-j}.$$

Then LP-2 leads to a closed form optimal allocation.

▶ **Theorem 10.** *Given n jobs with parameters such that for $i \in [n]$, whenever $i \neq m_i := \min\left(\arg\min_{j \geq i} \frac{u_j}{u_j + o(j)}\right)$, we have $u_{m_i} \geq o(i) - o(m_i)$, then the following is both an optimal solution to the LP and an optimal allocation,*

$$x_k = \frac{\sum_{i|k=m_i} \Delta_i o(i)}{u_k + o(k)} \quad \text{for all } k \in [n].$$

Optimal Allocation Beyond Constant Per-unit Costs

For non-decreasing, piece-wise linear and convex costs $o_i(\cdot), u_i(\cdot)$, let $\overline{o}_i, \underline{o}_i$ denote the largest and smallest slope for $o_i(\cdot)$ and, $\overline{u}_i, \underline{u}_i$ the largest and smallest slope for $u_i(\cdot)$. By generalizing insights from the case of constant per-unit costs, we show that Algorithm 1 finds the optimal allocation when $\underline{u}_{i+1} \geq \overline{u}_i$, for all $i \in [n-1]$. This condition is satisfied for instance if $u_i(x) = u \cdot x$ for all $i \in [n]$, and $o_i(\cdot)$ is an arbitrary non decreasing, convex piece-wise linear function. Similar to the case of constant per-unit costs, our assumption complements assumptions made in previous work on the stochastic setting of the problem. For instance, in the stochastic setting Ge et al. [10], gave a polynomial time algorithm for non decreasing, piece-wise linear and convex costs when $\overline{u}_{i+1} \leq \underline{u}_i + \underline{o}_i$, for all $i \in [n-1]$.

Algorithm 1 Allocation for Non-Linear Costs.

for $i = n$ to 1 **do**
  $\hat{o}_i(x) := \sum_{j \geq i} \left(o_j(\Delta_i - x + \sum_{k|i<k\leq j}(\Delta_k - x_k)) - o_j(\sum_{k|i<k\leq j}(\Delta_k - x_k)) \right)$
  $x_i = \operatorname{argmin}_{x \geq 0} \max\{u_i(x), o_i(\Delta_i - x) + \hat{o}_i(x)\}$
Output: $\{x_i\}_i$

▶ Remark. Given two sets of jobs $S_{k-1} = \{k-1, \ldots, n\}$ and $S_k = \{k, \ldots, n\}$. Let the output of Algorithm 1 over set S_{k-1} be $\{x_i^{k-1}\}$, and over set S_k be $\{x_i^k\}$. Then we have, $x_i^{k-1} = x_i^k$ for all $i \in S_k$. Also, note that the algorithm takes n iterations, each involves a minimization of the maximum of two convex piece-wise linear functions (as $u_i(x)$ and $o_i(\Delta_i - x) + \hat{o}_i(x)$ are both piece-wise linear and convex in x). Therefore, each step involves finding the minimum of a convex piece-wise linear function. Hence, the algorithm runs in polynomial time (in n and the number of pieces in the cost functions).

4 Robust Appointment Scheduling and Ordering

So far, we assumed that jobs are given to us in fixed order and our task was to find an optimal appointment schedule. Focusing on the case of constant per-unit costs, we now consider the joint problem of finding an ordering and an appointment schedule such that the resulting cost is minimized. More formally, consider a permutation π over the set $[n]$, that determines the order of appointments. So given an ordering π, job i is the $\pi(i)$-th appointment in the schedule. We let A_i, C_i denote the start time and the completion time of the i-th appointment (or the $\pi^{-1}(i)$-th job). The joint scheduling and ordering problem can now be stated as,

$$\min_{\pi:[n]\to[n],\{A_i\}_i} \quad \max_{t_i \in [0,\Delta_i]\,\forall i} \sum_{i=1}^{n} \max\{o_{\pi^{-1}(i)}(C_i - A_{i+1}), u_{\pi^{-1}(i)}(A_{i+1} - C_i)\}. \tag{7}$$

Recall that under homogeneous underage costs $u_i = u \, \forall i$, we have a closed form solution for the problem for fixed permutation π. Letting $o(i) = \sum_{j|\pi(j)\geq\pi(i)} o_j$, the objective in this special case can be more simply stated as,

$$\min_{\pi:[n]\to[n]} \sum_{i=1}^{n} \frac{\Delta_i o(i) u}{o(i)+u}.$$

Let us call this problem RASO-H for brevity. Mittal et al. [20] showed that RASO-H reduces to an instance of min-sum scheduling with concave costs $1||\sum w_j f(C_j)$. Here given jobs j with processing time p_j, weight w_j, and a concave function $f(\cdot)$ over completion times C_j, the goal is to find an ordering that achieves the following,

$$\min_{\{C_j\}_j} \sum_{j=1}^{n} w_j f(C_j). \tag{8}$$

Indeed, letting $w_j = \Delta_j$, $p_j = o_j$ and $f(C_j) = \frac{uC_j}{C_j+u}$, we see that any ordering π for RASO-H is equivalent to an order π' in, $1||\sum w_j f(C_j)$. Here $\pi'(j) = n - \pi(j) + 1$ (i.e., orders are reversed as we move between the two problems). This reduces RASO-H to an instance of $1||\sum w_j f(C_j)$, while preserving the objective value. Therefore, algorithms for $1||\sum w_j f(C_j)$ can be used directly for RASO-H without any loss in guarantee. Using a result in [12], the following rule,

$$\text{Smith's rule: Schedule jobs in the order of descending ratios } \tfrac{w_j}{p_j} \text{ (or ascending ratios } \tfrac{\Delta_j}{o_j}),$$

is 1.137 approximate for RASO-H. In fact, there is an EPTAS for RASO-H due to the EPTAS for $1||\sum w_j f(C_j)$ [25]. Howevever, the hardness of RASO with a single underage cost, and more generally of $1||\sum w_j f(C_j)$, remains an intriguing open problem in scheduling theory [25, 19, 12, 20].

In the upcoming section, we consider the general problem for which no results were previously known. We give evidence indicating that the problem becomes much harder without the homogeneous underage costs assumption.

4.1 RASO with General Underage Costs

We show that RASO is strongly NP-hard when there are at least two different underage costs via a reduction from the strongly NP hard problem of min-sum scheduling on identical parallel machines, $P||\sum w_j C_j$ (problem SS13 in Garey and Johnson [9]). Further, we also show that no list ordering rule (such as Smith's rule) can be better than $O(n)$ approximate for the problem, ruling out the existence of "simple" approximation heuristics. We defer the details and proofs to the full version.

To develop approximation heuristics for RASO, unlike RASO-H, we cannot rely on a closed-form solution for the optimal allocation problem to simplify the problem. However, we do have a closed-form solution with cost guaranteed to be within a constant factor of the optimal due to Theorem 3. This does not immediately lead to a tractable problem, and taking a different approach we instead consider the closed form allocation given by the formula $x_i = \frac{\Delta_i o(i)}{u_i + o(i)}$. Given order π, let $o(i) = \sum_{j|\pi(j)\geq\pi(i)} o_j$. For this allocation rule the ordering problem simplifies to,

$$\min_{\pi:[n]\to[n]} \sum_{j=1}^{n} \frac{\Delta_j o(j) u_j}{o(j)+u_j}. \tag{9}$$

This is equivalent to a min-sum scheduling problem of the form,

$$\min_{\{C_j\}_j} \sum_j f_j(C_j).$$

Here C_j is the completion time of job j which has processing time $p_j = o_j$. Functions $f_j(C_j) = \Delta_j u_j \frac{C_j}{C_j+u_j}$ are concave, but now we have a different function for each job. We show (details in full version) that given an α approximation for $1||\sum_j f_j(C_j)$, there is a $4\alpha n$ approximation for RASO. The additional factor of $4n$ arises from the fact that (9) does not represent the true objective (7). Recall that the closed form allocation formula $x_j = \Delta_j \frac{o(j)}{o(j)+u_j}$, is not only suboptimal but can be arbitrarily worse than the optimal allocation for certain orders. So it is perhaps surprising that using this allocation we can still achieve some approximation bound. Note that for the min-sum scheduling problem, $1||\sum f_j(C_j)$, [7] gives a $4+\epsilon$ approximation for arbitrary f_j. Using this algorithm, we have a $(16+\epsilon)n$ approximation for RASO. We also show that no heuristic that orders jobs based on a simple ratio can beat $\Omega(n)$. Here we use the term "simple ratio" to refer to any real valued function evaluated independently for each job, using only the job parameters (o_i, Δ_i, u_i). A list ordering heuristic for this function then simply orders jobs in ascending or descending order of the function values.

4.2 Homogeneous Underage Costs

In this section, we develop some easy to implement heuristics with improved approximations for RASO-H. As we mentioned earlier, the computational complexity of RASO-H and more generally, $1||\sum w_j f(C_j)$ with concave cost function f, remains open but, there is a scaling based EPTAS for RASO-H due to the EPTAS for $1||\sum w_j f(C_j)$. The EPTAS is non-trivial and not easy to implement in practice. Practical heuristics such as Smith's rule, can be suboptimal for RASO-H even with just two jobs. This motivates us to consider the following heuristic,

Customized-Smith's rule (C-Smith): Schedule jobs in ascending order of $\frac{\Delta_i}{o_i(o_i+u)}$.

This straightforward heuristic is optimal for two jobs by design, and has an approximation guarantee of β, where $1.06036 < \beta < 1.06043$. While the order output by C-Smith is optimal for two jobs, we could more generally seek an order that is optimal w.r.t. exchanging the order of any two consecutive jobs. Let us call such an order *locally optimal*. The schedule output by both Smith and C-Smith is not locally optimal. In fact, no list-ordering heuristic can be locally optimal for all instances of RASO-H. To address this we introduce Algorithm 2, which outputs a locally optimal order. While not as simple as C-Smith, it is still fairly easy to implement – at each step, it computes a new set of ratios for the remaining jobs and picks the one with the best ratio, removing it from further consideration. However, a naive implementation has runtime $O(n^2)$, in contrast to $O(n \log n)$ for list ordering heuristics.

We show that Algorithm 2 outputs a solution that is at least as good as C-Smith on every instance. Thus, it is also β approximate. There exist instances for which the algorithm is not optimal. However, we leave finding the exact guarantee of the algorithm as an open problem.

▶ **Theorem 11.** *For every instance of RASO-H, Algorithm 2 outputs an order that is at least as good as the order given by C-Smith.*

The proof is deferred to the full version. It remains to show the β approximation for C-Smith. For this analysis it will be much more convenient both for clarity as well as notation, to focus on analyzing C-Smith for the scheduling problem $1||\sum_j w_j f(C_j)$, where

Algorithm 2 Locally Optimal Algorithm.

Initialize: $S = \emptyset, o(S) = 0$
for $i = 1$ to n **do**
  Find,

$$j = \underset{k \in [n] \setminus S}{\operatorname{argmax}} \frac{\Delta_k}{o_k(o_k + o(S) + 1)}.$$

  In case of a tie, pick the job with largest overage cost.

  $\pi(j) = n - i + 1, S = S \cup \{j\}, o(S) = o(S) + o_j$
Output: $\pi(.)$

$f(C_j) = \frac{C_j u}{C_j + u}$ and $w_j = \Delta_j, p_j = o_j$. Note that we can let $u = 1$ w.l.o.g. Also recall, there is a cost preserving bijection between orders for $1||\sum_j w_j f(C_j)$ and RASO-H – reversing the order when moving from one to the other. Therefore, we will show the following equivalent theorem.

▶ **Theorem 12.** *For some constant $\beta \in [1.06036, 06043]$. scheduling jobs in descending order of $\frac{w_j}{p_j(p_j+1)}$ is exactly β approximate for the scheduling problem $1||\sum_j w_j \frac{C_j}{C_j+1}$.*

To prove the above theorem, we establish several intermediate results that characterize and simplify the worst-case instance for C-Smith. First, we show that there is a worst-case instance for C-Smith where all jobs are tied and in fact have ratio 1. This is shown via a generalization of Lemma 3.5 in [12] (Lemma 21 in [25]). Then, we show that in the worst-case C-Smith orders jobs in ascending order of processing times and the optimal order is the exact reverse. Interestingly, this is the opposite of Lemma 3.6 in [12] (Proposition 20 in [25]) for Smith's rule, where the optimal order is ascending in p_i and the worst order is descending. Given these properties, we can formulate a non-convex optimization problem in infinitely many variables, the optimal value of which is the approximation ratio, and every optimal solution is a worst-case instance. To then get lower and upper bounds on the optimum value, we utilize properties specific to our objective $f(x) = \frac{x}{x+1}$ to approximate the problem (which has infinitely many variables) with a family of optimization problems, that while still non-convex, have a finite number of variables. More complex objectives from the family give a tighter upper bound on true approximation ratio, but the number of variables involved increases. We find the global optimum to a problem in the family with five variables, and this closely matches our lower bound. We solve such a non-convex problem to global optimality by establishing upper and lower bounds on variables and using linear cuts, both of which allow us to then effectively use a nonlinear globally optimal MINLP solver, Couenne [13] (details in full version).

5 Conclusion & Open Problems

We considered the robust appointment scheduling problem with general underage costs. For the appointment scheduling problem with fixed jobs order, we found a simple LP that gives a 2-approximation for the problem under constant per-unit costs. Then we further refined this LP, resulting in a closed form solution for optimal allocations in special cases (generalizing previous results in the robust model, and complementing similar results in other models). We also showed that our results and approach extend more generally to convex piece-wise

linear costs. When seeking an optimal allocation for the general case using our approach, more complications arise and it is not clear if one can still construct a linear (or convex) program such that an optimal solution to the program is also an optimal allocation. We leave finding an optimal solution for the general case (or showing hardness), as an open problem.

In the second setting, we considered the problem of jointly finding the optimal order and allocation given that order for the case of constant per-unit costs. For the case of heterogeneous underage costs, we show that the problem is strongly NP hard and no list ordering policy can do better than $\Omega(n)$ to approximate the optimal value. We also gave a heuristic that achieves this bound. Finding a better approximation for this setting remains another interesting open problem. For the case of homogeneous underage costs, we designed two simple and practical heuristics that are guaranteed to be with ≈ 1.06 of the optimal.

References

1 M. A Begen, R. Levi, and M. Queyranne. A sampling-based approach to appointment scheduling. *Operations Research*, 60(3):675–681, 2012.

2 M. A Begen and M. Queyranne. Appointment scheduling with discrete random durations. *Mathematics of Operations Research*, 36(2):240–257, 2011.

3 A. Ben-Tal and A. Nemirovski. Robust optimization–methodology and applications. *Mathematical Programming*, 92(3):453–480, 2002.

4 D. Bertsimas, D. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

5 D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

6 T. Cayirli and E. Veral. Outpatient scheduling in health care: a review of literature. *Production and Operations Management*, 12(4):519–549, 2003.

7 M. Cheung, J. Mestre, D. B. Shmoys, and J. Verschae. A Primal-Dual Approximation Algorithm for Min-Sum Single-Machine Scheduling Problems. *SIAM Journal on Discrete Mathematics*, 31(2):825–838, 2017.

8 B. Denton and D. Gupta. A sequential bounding approach for optimal appointment scheduling. *IIE Transactions*, 35(11):1003–1016, 2003.

9 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

10 D. Ge, G. Wan, Z. Wang, and J. Zhang. A note on appointment scheduling with piecewise linear cost functions. *Mathematics of Operations Research*, 39(4):1244–1251, 2013.

11 L. V. Green, S. Savin, and B. Wang. Managing patient service in a diagnostic medical facility. *Operations Research*, 54(1):11–25, 2006.

12 W. Höhn and T. Jacobs. On the performance of Smith's rule in single-machine scheduling with nonlinear cost. *ACM Transactions on Algorithms*, 11(4):25, 2015.

13 IBM and Carnegie Mellon University. Couenne, an exact solver for nonconvex MINLPs. `https://projects.coin-or.org/Couenne/`, 2006.

14 R. Jiang, S. Shen, and Y. Zhang. Integer programming approaches for appointment scheduling with random no-shows and service durations. *Operations Research*, 65(6):1638–1656, 2017.

15 G. C. Kaandorp and G. Koole. Optimal outpatient appointment scheduling. *Health Care Management Science*, 10(3):217–229, 2007.

16 Q. Kong, C. Lee, C. Teo, and Z. Zheng. Scheduling arrivals to a stochastic service delivery system using copositive cones. *Operations Research*, 61(3):711–726, 2013.

17 A. Macario. Is it possible to predict how long a surgery will last? *Medscape Anesthesiology*, 108(3):681–685, 2010.

18 H. Mak, Y. Rong, and J. Zhang. Appointment scheduling with limited distributional information. *Management Science*, 61(2):316–334, 2014.

19 N. Megow and J. Verschae. Dual techniques for scheduling on a machine with varying speed. In *Automata, Languages, and Programming - 40th International Colloquium (ICALP)*, pages 745–756, 2013.

20 S. Mittal, A. S. Schulz, and S. Stiller. Robust appointment scheduling. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

21 J. Orlin. Personal communication, 2018.

22 Y. Pinto and R. Shamir. Efficient algorithms for minimum-cost flow problems with piecewise-linear convex costs. *Algorithmica*, 11(3):256–277, 1994.

23 L. W. Robinson and R. R. Chen. Scheduling doctors' appointments: optimal and empirically-based heuristic policies. *IIE Transactions*, 35(3):295–307, 2003.

24 W. E. Smith. Various optimizers for single-stage production. *Naval Research Logistics*, 3(1-2):59–66, 1956.

25 S. Stiller and A. Wiese. Increasing Speed Scheduling and Flow Scheduling. In *Algorithms and Computation - 21st International Symposium (ISAAC)*, pages 279–290, 2010.

26 P. P. Wang. Static and dynamic scheduling of customer arrivals to a single-server system. *Naval Research Logistics*, 40(3):345–360, 1993.

27 P. P. Wang. Sequencing and scheduling n customers for a stochastic server. *European Journal of Operational Research*, 119(3):729–738, 1999.

Collapsing Superstring Conjecture

Alexander Golovnev
Harvard University, Cambridge, MA, USA

Alexander S. Kulikov
Steklov Institute of Mathematics at St. Petersburg, Russian Academy of Sciences, Russia

Alexander Logunov
St. Petersburg State University, Russia

Ivan Mihajlin
University of California, San Diego, CA, USA

Maksim Nikolaev
St. Petersburg State University, Russia

Abstract

In the Shortest Common Superstring (SCS) problem, one is given a collection of strings, and needs to find a shortest string containing each of them as a substring. SCS admits $2\frac{11}{23}$-approximation in polynomial time (Mucha, SODA'13). While this algorithm and its analysis are technically involved, the 30 years old Greedy Conjecture claims that the trivial and efficient Greedy Algorithm gives a 2-approximation for SCS.

We develop a graph-theoretic framework for studying approximation algorithms for SCS. The framework is reminiscent of the classical 2-approximation for Traveling Salesman: take two copies of an optimal solution, apply a trivial edge-collapsing procedure, and get an approximate solution. In this framework, we observe two surprising properties of SCS solutions, and we conjecture that they hold for all input instances. The first conjecture, that we call Collapsing Superstring conjecture, claims that there is an elementary way to transform any solution repeated twice into the same graph G. This conjecture would give an elementary 2-approximate algorithm for SCS. The second conjecture claims that not only the resulting graph G is the same for all solutions, but that G can be computed by an elementary greedy procedure called Greedy Hierarchical Algorithm.

While the second conjecture clearly implies the first one, perhaps surprisingly we prove their equivalence. We support these equivalent conjectures by giving a proof for the special case where all input strings have length at most 3 (which until recently had been the only case where the Greedy Conjecture was proven). We also tested our conjectures on millions of instances of SCS.

We prove that the standard Greedy Conjecture implies Greedy Hierarchical Conjecture, while the latter is sufficient for an efficient greedy 2-approximate approximation of SCS. Except for its (conjectured) good approximation ratio, the Greedy Hierarchical Algorithm provably finds a 3.5-approximation, and finds *exact* solutions for the special cases where we know polynomial time (not greedy) exact algorithms: (1) when the input strings form a spectrum of a string (2) when all input strings have length at most 2.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Approximation algorithms analysis

Keywords and phrases superstring, shortest common superstring, approximation, greedy algorithms, greedy conjecture

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.26

Category APPROX

Related Version A full version of the paper is available at `https://arxiv.org/abs/1809.08669`.

Funding *Alexander Golovnev*: Supported by a Rabin Postdoctoral Fellowship.



© Alexander Golovnev, Alexander S. Kulikov, Alexander Logunov, Ivan Mihajlin, and Maksim Nikolaev;
licensed under Creative Commons License CC-BY



1 Introduction

The *shortest common superstring problem* (abbreviated as SCS) is: given a set of strings, find a shortest string that contains all of them as substrings. This problem finds applications in genome assembly [33, 24], and data compression [9, 8, 28]. We refer the reader to the excellent surveys [10, 21] for an overview of SCS, its applications and algorithms. SCS is known to be **NP**-hard [9] and even **MAX-SNP**-hard [3], but it admits constant-factor approximation in polynomial time.

The best known approximation ratios are $2\frac{11}{23}$ due to Mucha [22] and $2\frac{11}{30}$ due to Paluch [23] (see [12, Section 2.1] for an overview of the previous approximation algorithms and inapproximability results). While these approximation algorithms use an algorithm for Maximum Weight Perfect Matching as a subroutine, the 30 years old *Greedy Conjecture* [28, 30, 31, 3] claims that the trivial *Greedy Algorithm*, whose pseudocode is given in Algorithm 1, is 2-approximate. Ukkonen [32] shows that for a fixed alphabet, the Greedy Algorithm can be implemented in linear time. It should be noted that GA is not deterministic as we do not specify how to break ties in case when there are many pairs of strings with maximum overlap. For this reason, GA may produce different superstrings for the same input.

Algorithm 1 Greedy Algorithm (GA).

Input: set of strings $\mathcal{S}$.
Output: a superstring for $\mathcal{S}$.

```
while S contains at least two strings do
    extract from S two strings with the maximum overlap
    add to S the shortest superstring of these two strings
return the only string from S
```

▶ **Greedy Conjecture.** *For any set of strings $\mathcal{S}$,* $\mathrm{GA}(\mathcal{S})$ *constructs a superstring that is at most twice longer than an optimal one.*

Blum et al. [3] prove that the Greedy Algorithm returns a 4-approximation of SCS, and Kaplan and Shafrir [15] improve this bound to 3.5. A slight modification of the Greedy Algorithm gives a 3-approximation of SCS [3], and other greedy algorithms are studied from theoretical [3, 25] and practical perspectives [26, 4].

It is known that the Greedy Conjecture holds for the case when all input strings have length at most 3 [30, 7], and it was recently shown to hold in the case of strings of length 4 [18]. Also, the Greedy Conjecture holds if the Greedy Algorithm happens to merge strings in a particular order [35, 19]. The Greedy Algorithm gives a 2-approximation of a different metric called compression [30]. The compression is defined as the sum of the lengths of all input strings minus the length of a superstring (hence, it is the number of symbols saved with respect to a naive superstring resulting from concatenating the input strings).

Most of the approaches for approximating SCS are based on the *overlap graph* or the equivalent *suffix graph*. The suffix graph has input strings as nodes, and a pair of nodes is joined by an arc of weight equal to their suffix (see Section 2.1 for formal definitions of overlap and suffix). SCS is equivalent to (the asymmetric version of) the Traveling Salesman Problem (TSP) in the suffix graph. While TSP cannot be approximated within any polynomial time computable function unless $\mathbf{P} = \mathbf{NP}$ [27], its special case corresponding to SCS can be

approximated within a constant factor.[1] We do not know the full characterization of the graphs in this special case, but we know some of their properties: Monge inequality [20] and Triple inequality [35]. These properties are provably not sufficient for proving Greedy Conjecture [35, 19].

While the overlap and suffix graphs give a convenient graph structure, our current knowledge of their properties is provably not sufficient for showing strong approximation factors. Thus, the known approximation algorithms (including the Greedy Algorithm) estimate the approximation ratio via the overlap graph, and also separately take into account some string properties not represented by the overlap graph. The goal of this work is to develop a simple combinatorial framework which captures all features of the input strings needed for proving approximation ratios of algorithms.

1.1 Our contributions

We continue the study of the so-called *hierarchical graph* introduced by Golovnev et al. [13]. (See also [5] for a related notion of the superstring graph.) This graph is designed specifically for the SCS problem, in some sense it generalizes de Bruijn graph, and it contains more information about the input strings than just all pairwise overlaps. Given an instance of SCS, the vertex set of the corresponding hierarchical graph is just the set of substrings of all the input strings. For a string s and two symbols α, β, the graph contains the arcs: $(s, s\alpha)$ and $(\beta s, s)$. Now, every superstring of the given set of string corresponds to an Eulerian walk in the hierarchical graph (which passes through the vertices corresponding to the input strings), and vice versa. (See Section 2.2 for formal definition and statements.)

1.1.1 Collapsing Conjecture

We define a simple normalization procedure of a walk in the hierarchical graph: replace the pair of arcs $(\alpha s, \alpha s\beta), (\alpha s\beta, s\beta)$ with the pair $(\alpha s, s), (s, s\beta)$ as long as it does not violate connectivity of the walk. It is easy to see that such a normalization never increases the length of the corresponding solution of SCS. First, we observe a surprising property of this normalization procedure: if one takes *any* solution, doubles all of its arcs in the hierarchical graph, and then applies the normalization procedure, then the resulting set of arcs is always the same (i.e., it does not depend on the initial solution). Collapsing Conjecture makes this observation formal (see Section 3). Note that this conjecture implies an extremely simple 2-approximate algorithm for Shortest Common Superstring: take any solution (for example, write down all input strings one after another), then double each arc in the hierarchical graph, and apply the simple normalization procedure. This procedure will result in some superstring S. On the other hand, if one started with an optimal solution, doubled each of its arcs, and normalized the result, then the resulting solution would have length at most twice the length of the optimal solution. By Collapsing Conjecture, this resulting superstring would also be S, which implies that S is a 2-approximation.

[1] We remark that SCS is also a special case of TSP for costs satisfying the triangle inequaliy. This case of TSP can be approximated within a constant factor [29], but this factor is currently much worse than that for SCS.

1.1.2 Greedy Hierarchical Conjecture

We also propose a simple and natural greedy algorithm for SCS in the hierarchical graph: start from the nodes corresponding to the input strings, and greedily build an Eulerian walk passing through all of them. While this Greedy Hierarchical Algorithm (GHA) is as simple as the Greedy Algorithm (GA), it provably performs better in some cases. For example, there are two well-known polynomially solvable special cases of SCS: strings of length 2 and a spectrum of a string. While GA does not always find optimal solutions in these cases, GHA solves them exactly (see Sections B.1 and B.2).

Greedy Hierarchical Conjecture (see Section 4) claims that the set of arcs produced by GHA exactly matches the set of arcs from the Collapsing Conjecture: whichever initial solution one takes, after doubling its arcs and normalization, the resulting set of arcs is exactly the solution found by GHA. Clearly, this conjecture implies Collapsing Conjecture. Perhaps surprisingly, we prove that the two conjectures are equivalent (see Section 5.1): if all doubled solutions after normalization result in the same set of arcs, then this set of arcs is the GHA solution.

The weak form of Greedy Hierarchical Conjecture claims that GHA is a 2-approximate algorithm for SCS. We prove (see Section 5.2) that GHA is an instantiation of GA with some tie-breaking rule. That is, there is an algorithm which always merges some pair of strings with the longest overlap and outputs the same solution as GHA. This result has two consequences. First, by the known results for GA, we immediately have that GHA finds a 3.5-approximation for SCS. Second, this gives us that Greedy Conjecture implies Weak Greedy Hierarchical Conjecture.

1.1.3 Evidence for the Conjectures

We support the Collapsing Conjecture (and the equivalent Greedy Hierarchical Conjecture) by proving its special case and verifying it empirically. We prove the conjecture for the special case where all input strings have length at most 3, which until recently had been the only case where the Greedy Conjecture was proven (see Section A). Despite tesing the conjecture on millions of datasets (both hand-crafted and generated randomly according to various distributions), we have not found a counter-example. Note that even the Weak Greedy Hierarchical Conjecture suffices for getting a 2-approximation for SCS, and this conjecture is not harder to prove than the standard Greedy Conjecture. We implemented the Greedy Hierarchical Algorithm [11], and we invite the reader to its web interface [34] to see step by step executions of the described algorithms and to verify the conjectures on custom datasets.

2 Definitions

2.1 Shortest Common Superstring Problem

For a string s, by $|s|$ we denote the length of s. For strings s and t, by $\mathrm{overlap}(s,t)$ we denote the longest suffix of s that is also a prefix of t. By $\mathrm{pref}(s,t)$ we denote the first $|s| - |\mathrm{overlap}(s,t)|$ symbols of s. Similarly, $\mathrm{suff}(s,t)$ is the last $|t| - |\mathrm{overlap}(s,t)|$ symbols of t. By $\mathrm{pref}(s)$ and $\mathrm{suff}(s)$ we denote, respectively, the first and the last $|s| - 1$ symbols of s. See Figure 1 for a visual explanation. We denote the empty string by ε.

Throughout the paper by $\mathcal{S} = \{s_1, \ldots, s_n\}$ we denote the set of n input strings. We assume that no input string is a substring of another (such a substring can be removed from $\mathcal{S}$ in the preprocessing stage). Note that SCS is a *permutation problem*: to find a shortest

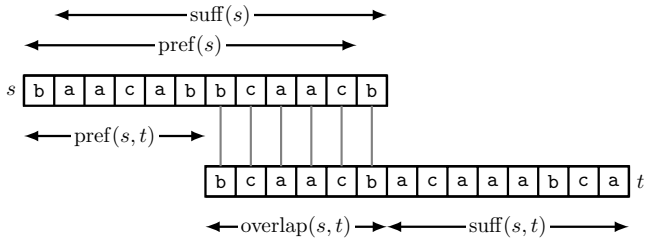


Figure 1 Pictorial explanations of pref, suff, and overlap functions.

string containing all s_i's in a *given order* one just overlaps the strings in this order, see Figure 2. (This simple observation relates SCS to other permutation problems, including various versions of the Traveling Salesman Problem.) It will prove convenient to view the SCS problem as a problem of finding an optimum permutation. It should be noted at the same time that the correspondence between permutations and superstrings is not one-to-one: there are superstrings that do not correspond to any permutation. For example, the concatenation of input strings is clearly a superstring, but it ignores the fact that neighbor strings may have non-trivial overlaps and for this reason may fail to correspond to a permutation. Still, clearly, any *shortest* superstring corresponds to a permutation of the input strtings.

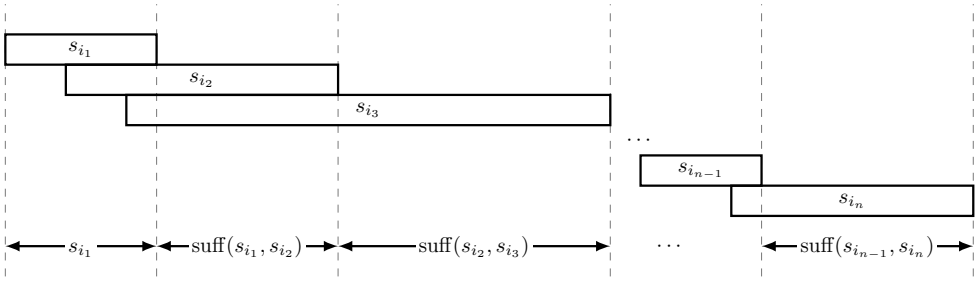


Figure 2 SCS is a permutation problem. The length of a superstring corresponding to a permutation $(s_{i_1}, \ldots, s_{i_n})$ is $|s_{i_1}|$ plus the sum of the lengths of suffixes of consecutive pairs of strings. It is also equal to $\sum_{i=1}^{n} |s_i| - \sum_{j=1}^{n-1} |\operatorname{overlap}(s_{i_j}, s_{i_{j+1}})|$.

2.2 Hierarchical Graph

For a set of strings $\mathcal{S}$, the *hierarchical graph* $HG = (V, E)$ is a weighted directed graph with $V = \{v\colon v \text{ is a substring of some } s \in \mathcal{S}\}$. For every $v \in V$, $v \neq \varepsilon$, the set of arcs E contains an *up-arc* $(\operatorname{pref}(v), v)$ of weight 1 and a *down-arc* $(v, \operatorname{suff}(v))$ of weight 0. The meaning of an up-arc is appending one symbol to the end of the current string (and that is why it has weight 1), whereas the meaning of a down-arc is cutting down one symbol from the beginning of the current string. Figure 3(a) gives an example of the hierarchical graph and shows that the terminology of up- and down-arcs comes from placing all the strings of the same length at the same level, where the i-th level contains strings of length i. In all the figures in this paper, the input strings are shown in rectangles, while all other vertices are ellipses.

What we are looking for in this graph is a shortest walk from ε to ε going through all the nodes from $\mathcal{S}$. It is not difficult to see that the length of a walk from ε to ε equals the length of the string spelled by this walk. This is just because each up-arc has weight 1 and adds one symbol to the current string. See Figure 3(b) for an example.

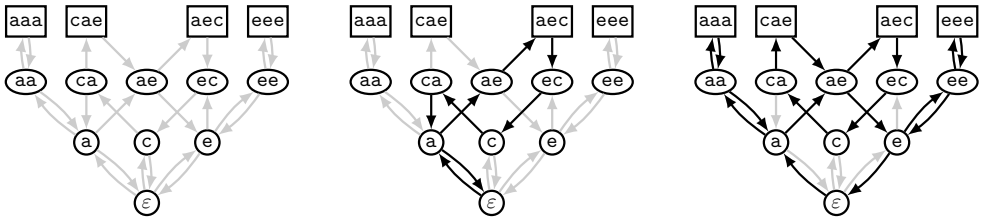


Figure 3 (a) Hierarchical graph for the dataset $\mathcal{S} = \{\texttt{aaa}, \texttt{cae}, \texttt{aec}, \texttt{eee}\}$. (b) The walk $\varepsilon \to \texttt{a} \to \texttt{ae} \to \texttt{aec} \to \texttt{ec} \to \texttt{c} \to \texttt{ca} \to \texttt{a} \to \varepsilon$ has length (or weight) 4 and spells the string aeca of length 4. (c) An optimal superstring for $\mathcal{S}$ is aaaecaeee. It has length 9, corresponds to the permutation (aaa, aec, cae, eee), and defines the walk of length 9 shown in black.

Hence, the SCS problem is equivalent to finding a shortest closed walk from ε to ε that visits all nodes from $\mathcal{S}$. Note that a walk may contain repeated nodes and arcs. The multiset of arcs of such a walk must be Eulerian (each vertex must have the same in- and out-degree, and the set of arcs must be connected). It will prove convenient to define an *Eulerian solution* in a hierarchical graph as an Eulerian multiset of arcs D that goes through ε and all nodes from $\mathcal{S}$. Given such a solution D, one can easily recover an Eulerian cycle (that might not be unique). This cycle spells a superstring of $\mathcal{S}$ of the same length as D. Figure 3(c) shows an optimal Eulerian solution.

A solution to SCS defines a permutation $(s_{i_1}, \ldots, s_{i_n})$ of the input strings, and this permutation naturally gives a "zig-zag" Eulerian solution in the hierarchical graph:

$$\varepsilon \to s_{i_1} \to \text{overlap}(s_{i_1}, s_{i_2}) \to s_{i_2} \to \text{overlap}(s_{i_2}, s_{i_3}) \to \cdots \to s_{i_n} \to \varepsilon\,. \tag{1}$$

This Eulerian solution is shown schematically in Figure 4(a). This schematic illustration is over simplified as the shown path usually has many self-intersections. Still, this point of view is helpful in understanding the algorithms presented later in the text. Figure 4(b) shows an "untangled" optimal Eulerain solution from Figure 3(c): by contracting nodes with equal labels into the same node, one gets exactly the solution from Figure 3(c).

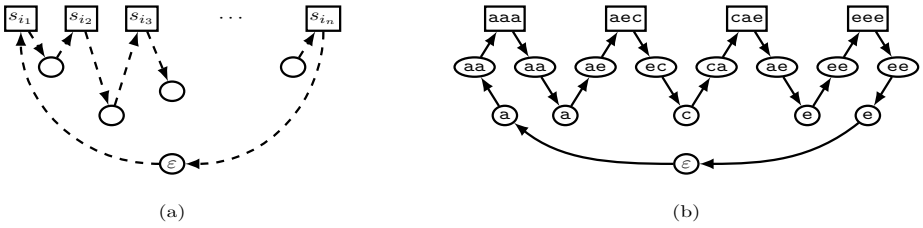


Figure 4 (a) A schematic illustration of a normalized Eulerian solution. (b) Untangled optimal Eulerian solution from Figure 3(c).

Not every Eulerian solution in the hierarchical graph has a nice zig-zag structure described above. In the next section, we introduce a normalization procedure (that we call collapsing) that allows us to focus on nice Eulerian solutions only.

2.3 Normalizing a Solution

In this section, we describe a natural way of normalizing an Eulerian solution D. Informally, it can be viewed as follows. Imagine that all arcs of D form one circular thread, and that there is a nail in every node $s \in \mathcal{S}$ corresponding to an input string. We apply *"gravitation"* to the thread, i.e., we replace every pair of arcs $(\mathrm{pref}(v), v)$, $(v, \mathrm{suff}(v))$ with a pair $(\mathrm{pref}(v), \mathrm{pref}(\mathrm{suff}(v)))$, $(\mathrm{pref}(\mathrm{suff}(v)), \mathrm{suff}(v))$, if there is no nail in v and if this does not disconnect D. We call this *collapsing*, see Figure 5.



Figure 5 Collapsing a pair of arcs is replacing a pair of dashed arcs with a pair of solid arcs: general case (left) and example (right). The "physical meaning" of this transformation is that to get `bac` from `aba` one needs to cut `a` from the beginning and append `c` to the end and these two operations commute.

A formal pseudocode of the collapsing procedure is given in Algorithm 2. The pseudocode, in particular, reveals an important exception (not covered in Figure 5): if $|v| = 1$, then $\mathrm{pref}(\mathrm{suff}(v))$ is undefined and we just remove the pair of arcs $(\mathrm{pref}(v), v)$ and $(v, \mathrm{suff}(v))$.

Algorithm 2 Collapse.

Input: hierarchical graph $HG(V, E)$, Eulerian solution D, node $v \in V$.

if $(\mathrm{pref}(v), v), (v, \mathrm{suff}(v)) \in D$ **then**
  $D \leftarrow D \setminus \{(\mathrm{pref}(v), v), (v, \mathrm{suff}(v))\}$
  if $|v| > 1$ **then**
    $D \leftarrow D \cup \{(\mathrm{pref}(v), \mathrm{pref}(\mathrm{suff}(v))), (\mathrm{pref}(\mathrm{suff}(v)), \mathrm{suff}(v))\}$

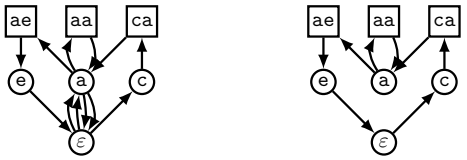
Algorithm 3, that we call Collapsing Algorithm (CA), uses the property described above to normalize any solution. It drops down all pairs of arcs that are not needed for connectivity. (Recall that a set of edges is called an Eulerian solution if it is connected and goes through all initial nodes and ε.)

Algorithm 3 Collapsing Algorithm (CA).

Input: set of strings $\mathcal{S}$, Eulerian solution D in HG.
Output: Eulerian solution D': $|D'| \leqslant |D|$

for level l in HG in descending order **do**
  for all $v \in V$ s.t. $|v| = l$ in lexicographic order: **do**
    while $(\mathrm{pref}(v), v), (v, \mathrm{suff}(v)) \in D$ and collapsing it keeps D an Eulerian solution **do**
      COLLAPSE(HG, D, v)
return D

It is easy to show (we prove this formally in Claim 2 on page 14) that any normalized solution is of the form (1). But it is not true that every zig-zag solution of the form (1) is a normalized solution: see Figure 6 for an example. The normalization procedure does not just turn a solution into some standard form, but it may also decrease its length.



(a) (b)

Figure 6 (a) An Eulerian solution corresponding to the permutation (ae, aa, ca). (b) The solution from (a) after normalization results in a shorter solution corresponding to the permutation (ca, aa, ae). This example also shows that though collapsing a pair of edges is a local change in the graph, it may drastically change the resulting superstring. In this case, it replaces a superstring aeaaca with a shorter superstring caae.

3 Collapsing Conjecture

We are now ready to conjecture an astonishing structural property of the hierarchical graph:

> Take any Eulerian solution, double every arc of it, and normalize the resulting solution; the result is the same for all initial solutions!

For the formal statement of the conjecture we use the following notation: If U and V are two multisets, then $U \sqcup V$ is the multiset W such that each $w \in W$ has multiplicity equal to the sum of multiplicities it has in sets U and V. Formally, the conjecture is stated as follows.

▶ **Collapsing Conjecture.** *For any set of strings $\mathcal{S}$ and any two Eulerian solutions D_1, D_2 of $\mathcal{S}$,*

$$CA(\mathcal{S}, D_1 \sqcup D_1) = CA(\mathcal{S}, D_2 \sqcup D_2)\,.$$

Figures 7 and 8 illustrate the action of the Collapsing Algorithm for optimal and naive solutions, respectively. Note that the resulting solutions are equal. When processing level $l > 1$ nodes, the collapsing procedure does not change the total length of the solution. What one normally sees at the beginning of the $l = 1$ iteration is an Eulerian solution with many redundant pairs of arcs of the form $(\mathtt{a}, \varepsilon)$, $(\varepsilon, \mathtt{a})$. It is exactly this stage of the algorithm where the total length of a solution is decreased by the Collapsing Algorithm.

We have verified the conjecture on millions of datasets (both handcrafted and randomly generated), and we invite the reader to see its visualizations and to check the conjecture on arbitrary datasets at the webpage [34]. Moreover, we support the conjecture by proving that it holds in the (NP-hard) special case where the input strings have length at most 3 in Section A.

If the Collapsing Conjecture is true, then there is a simple and natural 2-approximate algorithm for SCS: take *any* Eulerian solution (e.g., merge the input strings in arbitrary order), double it, and apply the Collapsing Algorithm. Under the conjecture, this results in the same Eulerian solution as for doubled optimal solution and hence the length of the result is at most twice the optimal length.

4 Greedy Hierarchical Conjecture

In this section, we present one more curious property of the Collapsing Algorithm that reveals its intricate connection to greedy algorithms. For this, we introduce the so called Greedy Hierarchical Algorithm (GHA) that constructs an Eulerian solution in a stingy fashion, i.e., tries to add as few arcs as possible:

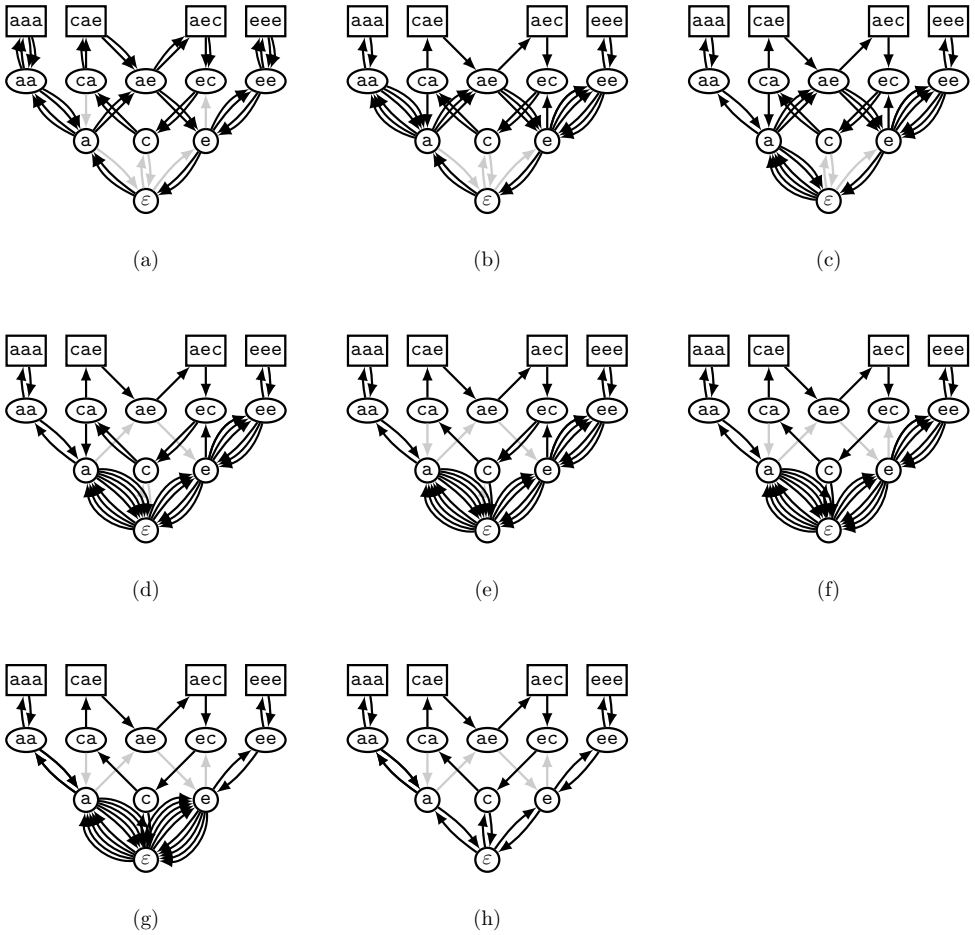


Figure 7 Stages of applying the Collapsing Algorithm to the dataset {aaa, cae, aec, eee} and its **optimal** solution. (a) We start by doubling every arc of the optimal solution from Figure 3(c). (b) After collapsing all nodes at level $l = 3$. (c) After processing the node aa at level $l = 2$. Note that the algorithm leaves a pair of arcs (a, aa), (aa, a) as they are needed to connect the component {aa, aaa} to the rest of the solution. (d) After processing the ae node. The algorithm collapses all pairs of arcs for this node as it lies in the same component as the node c. (e) After processing the ca node. (f) After processing the ec node. (g) After processing the ee node. Note that at this point the solution has exactly the same length as at the very beginning (at stage (a)). (h) Finally, after collapsing all the unnecessary pairs of arcs from the level $l = 1$.

Proceed nodes from top to bottom. For each node, ensure that it is balanced and connected to the rest of the solution.

This is best illustrated with an example, see Figure 9. We start constructing an Eulerian solution D by processing the nodes at level 3. The solution D must visit all these four nodes, so we add all incoming and outgoing arcs to D, see Figure 9(a). We then process the level 2. The node aa is balanced, but if we skip it, it will not be connected to the rest of the solution, so we add to D the arcs (a, aa) and (aa, a). The node ae is balanced, we do nothing for it. The node ca is imbalanced, so we add an arc (c, ca) to D. We balance the node ec similarly. The node ee is processed similarly to the node aa. The result of processing the second level is shown in Figure 9(b). On the last stage we connect the nodes a, b, and c to ε to ensure connectivity, see Figure 9(c). Hence, when processing level l, we only add arcs between levels l and $l - 1$.

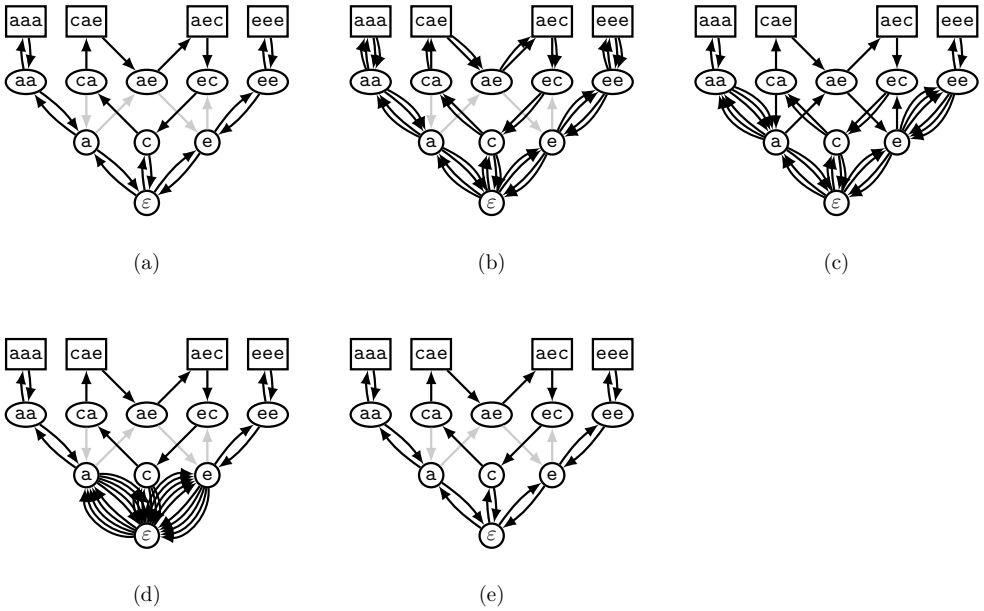


Figure 8 Stages of applying the Collapsing Algorithm to the dataset {aaa, cae, aec, eee} and its **naive** solution resulting from overlapping the input strings in the same order as they are given. (a) The solution of length 10 corresponding to the superstring aaacaeceee. (b) The doubled solution. (c) After collapsing the $l = 3$ level. (d) After collapsing the $l = 2$ level. (e) After collapsing the $l = 1$ level.

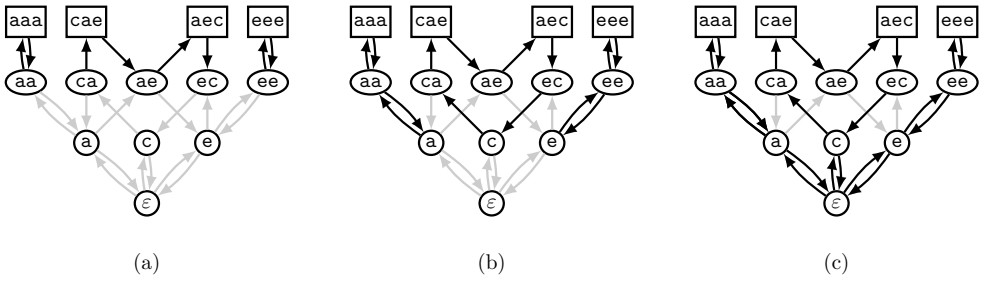


Figure 9 (a) After processing the $l = 3$ level. (b) After processing the $l = 2$ level. Note that for the node aa we add two lower arcs ((a, aa) and (aa, a)) since otherwise the corresponding weakly connected component ({aa, aaa}) will not be connected to the rest of the solution. At the same time, when processing the node ae we observe that it lies in a weakly connected component that contains imbalanced nodes (ca and ec), hence there is no need to add two lower arcs to ae. (c) After processing the $l = 1$ level. The resulting solution has length 10 and is, therefore, suboptimal (compare it with the optimal solution shown in Figure 3(c)).

More formally, GHA first considers the input strings $\mathcal{S}$. Since we assume that no $s \in \mathcal{S}$ is a substring of another $t \in \mathcal{S}$, there is no down-path from t to s in HG. This means that any walk through ε and $\mathcal{S}$ goes through the arcs $\{(\mathrm{pref}(s), s), (s, \mathrm{suff}(s)) \colon s \in \mathcal{S}\}$. The algorithm adds all of them to the constructed Eulerian solution D and starts processing all the nodes level by level, from top to bottom. At each level, we process the nodes in the lexicographic order. If the degree of the current node v is imbalanced, we balance it by adding an appropriate number of incoming (i.e., $(\mathrm{pref}(v), v)$) or outgoing (i.e., $(v, \mathrm{suff}(v))$)

arcs from the previous (i.e., lower) level. In the case when v is balanced, we just skip it. The only exception when we cannot skip it is when v *lies in an Eulerian component and* v *is the last chance of this component to be connected to the rest of the arcs in* D. (See, for example, the vertex `aa` in Figure 9(a)). The pseudocode is given in Algorithm 4.

Algorithm 4 Greedy Hierarchical Algorithm (GHA).

Input: set of strings $\mathcal{S}$.
Output: Eulerian solution D.

$HG(V, E) \leftarrow$ hierarchical graph of $\mathcal{S}$
$D \leftarrow \{(\text{pref}(s), s), (s, \text{suff}(s))\colon s \in \mathcal{S}\}$
for level l from $\max\{|s|\colon s \in \mathcal{S}\}$ downto 1 **do**
    for node $v \in V$ with $|v| = l$ in the lexicographic order **do**
        if $|\{(u, v) \in D\colon |u| = |v| + 1\}| \neq |\{(v, w) \in D\colon |w| = |v| + 1\}|$ **then**
            balance the degree of v in D by adding an appropriate number of lower arcs
        else
            $\mathcal{C} \leftarrow$ weakly connected component of v in D
            $u \leftarrow$ the lexicographically largest string among shortest strings in $\mathcal{C}$
            if $\mathcal{C}$ is Eulerian, $\varepsilon \notin \mathcal{C}$, and $v = u$ **then**
                $D \leftarrow D \cup \{(\text{pref}(v), v), (v, \text{suff}(v))\}$
return D

While GHA is almost as simple as the standard Greedy Algorithm (GA), GHA has several provable advantages over GA:

One advantage of GHA over GA is that GHA is more flexible in the following sense. On every step, GA selects two strings and fixes tightly their order. GHA instead works to ensure connectivity. When the resulting set D is connected, an actual order of input strings is given by the corresponding Eulerian cycle through D. This is best illustrated on the following toy example. For the dataset $\mathcal{S} = \{\texttt{ae}, \texttt{ea}, \texttt{ee}\}$, GA might produce a suboptimal solution `aeaee` if it merges the strings `ae` and `ea` at the first step. At the same time, it is not difficult to see that GHA finds an optimal solution for $\mathcal{S}$.

Another advantage of GHA is that, in contrast to GA, it solves *exactly* two well known polynomially solvable special cases of SCS: when the input strings have length at most two and when the input strings form a k-spectrum of an unknown string (that is, the input strings constitute all k-substrings of a string). We prove this formally in Sections B.1 and B.2. Informally, this happens because for such datasets there are no connectivity issues for GHA: for k-spectrum, after processing the highest level GHA gets a weakly connected component; for 2-SCS, after processing the level 2, GHA gets several weekly connected components such that different components do not share common letters and therefore are completely independent. Figure 6(b) illustrates this: while GA may produce a permutation $(\texttt{ca}, \texttt{ae}, \texttt{aa})$, GHA constructs an optimal permutation $(\texttt{ca}, \texttt{aa}, \texttt{ae})$.

In Section B.3, we also show a dataset where GHA produces a solution that is almost two times longer than the optimal one.

In Section 5.2, we show that the approximation guarantee of GHA is no worse than that of GA. Combining with the result of Kaplan and Shafrir [15], this implies immediately that GHA is 3.5-approximate. Moreover, we prove that the standard Greedy Conjecture implies 2-approximation of GHA, which makes it natural to study the approximation ratio of GHA.

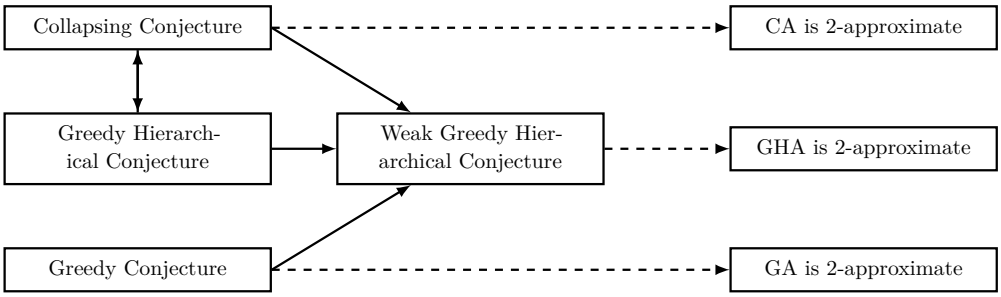


Figure 10 Relations between the conjectures (left), and the 2-approximate algorithms they imply (right). Collapsing and Greedy Hierarchical Conjectures are equivalent. They imply the weak version of the Greedy Hierarchical Conjecture, which also follows from the standard Greedy Conjecture. Each conjecture implies that the corresponding algorithm finds a 2-approximate solution for SCS.

We are now ready to state our second conjecture: the results of the Collapsing Algorithm and Greedy Hierarchical Algorithm coincide!

▶ **Greedy Hierarchical Conjecture.** *For any set of strings $\mathcal{S}$ and any Eulerian solution D,*

$$CA(\mathcal{S}, D \sqcup D) = GHA(\mathcal{S}) .$$

While the Greedy Hierarchical Conjecture implies that GHA finds a 2-approximate solution, we separately state this weak version of the conjecture.

▶ **Weak Greedy Hierarchical Conjecture.** *GHA is a factor 2 approximation algorithm for the Shortest Common Superstring problem.*

5 Relations between the Conjectures

In this section we prove some relations between the Collapsing and Greedy conjectures. Namely, in Section 5.1 we prove the equivalence of Collapsing and Greedy Hierarchical conjectures. In Section 5.2 we prove that the standard Greedy Conjecture implies Weak Hierarchical Greedy Conjecutre (which is sufficient for a simple 2-approximate greedy algorithm for SCS). Finally, it is easy to see that Greedy Hierarchical Conjecture implies its weak version: indeed, if every doubled solution results in the solution obtained by GHA, then GHA does not exceed twice the optimal superstring length. In Figure 10, we show the proven relations between the conjectures, together with 2-approximate algorithm which follow from each of the conjecture.

5.1 Equivalence of Collapsing and Greedy Hierarchical Conjectures

In this section we prove the equivalence of Collapsing Conjecture and Greedy Hierarchical Conjecture. Recall that Collapsing Conjecture claims that for any pair of Eulerian solutions D_1 and D_2 for the input strings $\mathcal{S}$, we have

$$CA(\mathcal{S}, D_1 \sqcup D_1) = CA(\mathcal{S}, D_2 \sqcup D_2) .$$

The Greedy Hierarchical Solution extends this statement to

$$CA(\mathcal{S}, D_1 \sqcup D_1) = GHA(\mathcal{S}) .$$

Greedy Hierarchical Conjecture trivially implies Collapsing conjecture, and in order to prove their equivalence, it suffices to show that the collapsing procedure applied to the doubled GHA solution results in the GHA solution:

$$CA(\mathcal{S}, GHA(\mathcal{S}) \sqcup GHA(\mathcal{S})) = GHA(\mathcal{S}) .$$

▶ **Theorem 1.** *For any set of strings $\mathcal{S}$,*

$$CA(\mathcal{S}, GHA(\mathcal{S}) \sqcup GHA(\mathcal{S})) = GHA(\mathcal{S}) .$$

Proof. Let us denote two copies of the $GHA(\mathcal{S})$ solution by B and R, which stand for a blue-copy and a red-copy. We will prove the theorem statement by showing that $CA(R \sqcup B)$ collapses all arcs of B and keeps untouched the arcs of R, as this implies that $CA(R \sqcup B) = R = GHA(\mathcal{S})$. For this, without loss of generality assume that the Collapsing Algorithm collapses blue arcs first, that is, if for a vertex v, CA can collapse a blue pair of arcs $(\text{pref}(v), v), (v, \text{suff}(v))$, it does so. Recall that CA processes vertices in the descending order of levels (Algorithm 3, line 1). We will prove that before processing level l, all the arcs above it (i.e., the arcs with at least one vertex at a level $> l$) do satisfy the desired property: all blue arcs are collapsed, and all red arcs are untouched.

The base case trivially holds for $l := \max\{|s| \,:\, s \in \mathcal{S}\}$, since the set of arcs above the level l is empty. Assume the claim is true for the level $k > 0$, and let us prove the claim for the level $k - 1$. Note that regardless of the number of collapse operations applied to B, B remains a set of walks: indeed, the collapse procedure keeps the balance of incoming and outgoing arcs for each vertex. By the induction hypothesis all the blue arcs above the level k are collapsed, so we have that if for a vertex v at level k there is an arc $(\text{pref}(v), v)$ in B, then there is also an arc $(v, \text{suff}(v))$ in B, and vice versa. Recall that CA collapses blue arcs when possible, and since every vertex has the same number of blue incoming and outgoing arcs, all pairs collapsed at the level k are monotone.

Now let us show that no red pair can be collapsed. Indeed, if for some vertex v at level k there is a red pair $(\text{pref}(v), v)$, $(v, \text{suff}(v))$, then by construction of GHA v is either in $\mathcal{S}$ or is the last chance of the corresponding component $\mathcal{C} \ni v$ to be connected to the remaining arcs in R (note that the first case is a subcase of the second one, as then $\mathcal{C}$ contains only one vertex). It follows that if CA collapses such a pair of arcs, then v has no blue arcs (as they have been collapsed before the red arcs), and all other vertices in the component $\mathcal{C}$ at the level k collapsed all arcs (since v is the last vertex in $\mathcal{C}$ in lexicographic order). Therefore, this pair is also the last chance of $\mathcal{C}$ to be connected to the rest of the arcs in R, thus, CA cannot collapse it.

It remains to show that all blue pairs at level k are collapsed. This trivially holds because no red pair is collapsed, and, thus, the connectivity of $R \sqcup B$ is maintained by R. This finishes the proof. ◀

5.2 Greedy Implies Greedy Hierarchical

Consider a permutation of the input strings. We say that it is a *valid greedy permutation* if it can be constructed by the Greedy Algorithm: there exist $n - 1$ merges of the n input strings that lead to this permutation such that at every step the two merged strings have the largest overlap. We will prove that GHA always returns a solution which corresponds to a greedy permutation of the input string. That is, while the standard Greedy Algorithm does not determine how to break ties, the Greedy Hierarchical Algorithm is a specific instantiation of the Greedy Algorithm with some tie-breaking rule.

We will use the following simple property of solutions constructed by the GHA algorithm.

▷ Claim 2. Let D be an Eulerian solution constrtucted by GHA. Then D has a "zig-zag" form as in (1).

Proof. First we prove that D is normalized, that is, any application of the collapsing procedure of Algorithm 2 to D will violate the property of Eulerian solution. Indeed, Algorithm 2 can only collapse pairs of arcs of the form $(\text{pref}(s), s), (s, \text{suff}(s))$. The Greedy Hierarchical Algorithm adds such pairs to its solution in two cases: (i) s is an input string (line 2 of Algorithm 4); (ii) s is the the lexicographically largest among the shortest strings in its Eulerian component (line 11 of Algorithm 4). Now note that in the former case, the collapsing procedure applied to s would violate the property that D must contain all input strings, and in the latter case, the collapsing procedure would violate the connectivity property of D.

We finish the proof by showing that every normalized solution is of the form (1). Let $\pi = (s_1, \ldots, s_n)$ be the permutation of the input strings corresponding to a normalized Eulerian solution D. Let us follow the arcs of D in the order of the permutation π, and let P be the set of arcs between the input strings s_i and s_{i+1}. We will prove that P is the union of the sets of arcs of the paths $s_i \to \text{overlap}(s_i, s_{i+1})$ and $\text{overlap}(s_i, s_{i+1}) \to s_{i+1}$. If P contains a pair of consecutive up- and down-arcs, that is, there exists a pair of arcs $(\text{pref}(s), s), (s, \text{suff}(s))$ in P, then this pair would have been collapsed by Algorithm 2, line 2. Therefore, the path P consists of a number of down-arcs followed by a number of up-arcs. It remains to show that the number of down-arcs in P is $d = |s_i| - |\operatorname{overlap}(s_i, s_{i+1})|$. Note that by the definition of $\text{overlap}(\cdot, \cdot)$, the number of down-arcs in P is at least d. On the other hand, if the number of down-arcs in P is strictly greater than d, then both the down-path and up-path in P contain the vertex $\text{overlap}(s_i, s_{i+1})$. This implies that the pair of arcs $(\text{pref}(s), s), (s, \text{suff}(s))$ for $s = \text{overlap}(s_i, s_{i+1})$ would have been collapsed by Algorithm 2, line 2, as it does not violate the connectivity of the solution. Therefore, the number of down-arcs in P is exactly d, which implies that P is the path $s_i \to \text{overlap}(s_i, s_{i+1})$ followed by the path $\text{overlap}(s_i, s_{i+1}) \to s_{i+1}$. ◁

▶ **Theorem 3.** *Every permutation $\pi = (s_1, \ldots, s_n)$ of the input strings constructed by GHA is a valid greedy permutation.*

Proof. Consider the following algorithm A: it starts with the sequence $(s_1, \ldots, s_n)$ obtained by GHA, and at every step it merges two neighboring strings in this sequence that have the largest overlap. It is a greedy algorithm, but instead of considering all pairwise overlaps, it only considers overlaps of neighboring strings in the sequence. Of course, in the end, this algorithm constructs exactly the permutation π. To show that π is a valid greedy permutation, we show that at every iteration of A no two strings have longer overlap than the two strings merged by A.

Consider, for the sake of contradiction, the first iteration when the algorithm A merges some pair of neighboring strings with overlap of length k whereas there are non-neighboring strings p and q with $v = \text{overlap}(p, q)$, $|v| > k$. At this point, p is a merger of input strings $s_a, s_{a+1}, \ldots, s_b$ and q is a merger of input strings $s_c, s_{c+1}, \ldots, s_d$. Then, from the assumption that no input string contains another input string, we have that $v = \text{overlap}(p, q) = \text{overlap}(s_b, s_c)$. Since the algorithm A merges neighboring strings in the decreasing order of overlap lengths, we have that $|\operatorname{overlap}(s_b, s_{b+1})| \le k < |v|$ and $|\operatorname{overlap}(s_{c-1}, c_c)| \le k < |v|$.[2]

Now we consider the Eulerian solution D constructed by GHA in the hierarchical graph. By Claim 2, D has a "zig-zag" form, thus, it contains all arcs from the path $s_b \to \text{overlap}(s_b, s_{b+1}) \to s_{b+1}$, and all arcs from the path $s_{c-1} \to \text{overlap}(s_{c-1}, s_c) \to s_c$.

[2] In the case when s_b is the last string in the solution (or s_c is the first string in the solution) we think of it being followed by ε, and $|\operatorname{overlap}(s_b, \varepsilon)| = 0 < |v|$ still holds.

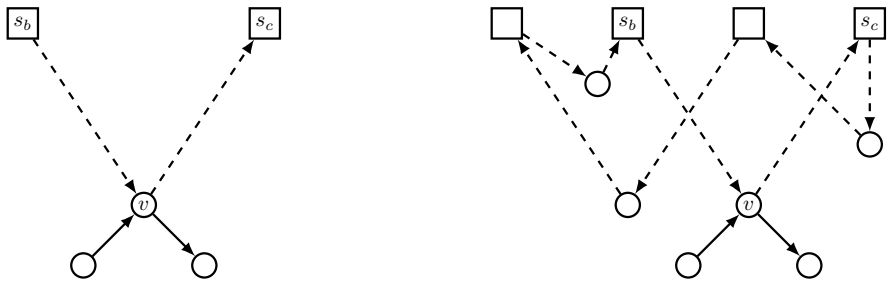


Figure 11 (a) In the Eulerian solution the node $v = \text{overlap}(s_b, s_c)$ has a pair of lower arcs. (b) For this reason, above v, there is an Eulerian component.

Recall that $v = \text{overlap}(s_b, s_c)$, and that $|\text{overlap}(s_b, s_{b+1})| < |v|$ and $|\text{overlap}(s_{c-1}, s_c)| < |v|$. In particular, the paths $s_b \to \text{overlap}(s_b, s_{b+1})$ and $\text{overlap}(s_{c-1}, s_c) \to s_c$ pass through the vertex v, which implies that the vertex v in the solution D has at least one incoming arc from the previous level and at least one outgoing arc to the previous level (see Figure 11(a)). Such a pair of arcs in the Eulerian solution D constructed by GHA may only occur when v is the last chance of its connected component to be connected to the rest of the solution (see line 11 of Algorithm 4). This, in turn, implies that right before the pair of arcs $(\text{pref}(v), v)$ and $(v, \text{suff}(v))$ was added to the Eulerian solution, there was an Eulerian component where v was the lexicographically largest among all shortest nodes. This component is shown schematically in Figure 11(b). All overlap-nodes (the nodes which are equal to $\text{overlap}(s_i, s_{i+1})$) of this component lie on levels $\geqslant k$. Note that the pair $(\text{pref}(v), v)$ and $(v, \text{suff}(v))$ is added to the solution by GHA exactly once (line 11 of Algorithm 4). Therefore, any path following the arcs of D, after going through the arc $(\text{pref}(v), v)$ must traverse the overlying component containing s_b and s_c (as otherwise the path could not reach the overlying component). In turn, this implies that after considering all overlaps of length $|v| > k$, s_b and s_c are already merged into one string, so they cannot be merged at this stage. ◀

Theorem 3 has two immediate corollaries.

▶ **Corollary 4.** *The Greedy Conjecture implies the Weak Greedy Hierarchical Conjecture: if the Greedy Algorithm is 2-approximate, then so is the Greedy Hierarchical Algorithm.*

Since every valid greedy permutation is a 3.5-approximation to the Shortest Common Superstring problem [15], we have the following corollary.

▶ **Corollary 5.** *GHA is a factor* 3.5 *approximation algorithm for the Shortest Common Superstring problem.*

6 Further Directions and Open Problems

The most immediate open problems are to prove the Collapsing Conjecture or the Weak Greedy Hierarchical Conjecture.

6.1 Applications of Hierarchical Graphs

It would also be interesting to find other applications of the hierarchical graphs. We list two such potential applications below.

Exact algorithms. Can one use hierarchical graphs to solve SCS exactly in time $(2-\varepsilon)^n$? It was shown in Section 1 that the SCS problem is a special case of the Traveling Salesman Problem. The best known exact algorithms for Traveling Salesman run in time $2^n \operatorname{poly}(|\operatorname{input}|)$ [2, 14, 17, 16, 1]. These algorithms stay the best known for the SCS problem as well. The hierarchical graphs were introduced [13] for an algorithm solving SCS on strings of length at most r in time $(2-\varepsilon)^n$ (where ε depends only on r). Can one use the hierarchical graph to solve exactly the general case of SCS in time $(2-\varepsilon)^n$ for a constant ε?

Genome assembly. The hierarchical graph in a sense generalizes de Bruijn graph. The latter one is heavily used in genome assembly [24]. Can one adopt the hierarchical graph for this task? For this, one would need to come up with a compact representation of the graph (as datasets in genome assembly are massive) as well as with a way of handling errors in the input data. Cazaux and Rivals [6] propose a linear-space counterpart of the hierarchical graph.

6.2 Optimal Cycle Covers

A superstring corresponds to a Hamiltonian path in the overlap graph, thus, a minimum-weight cycle cover gives a natural lower bound on its length. The Greedy Conjecture claims that a greedy solution never exceeds twice the length of an optimal solution. It is also believed (see, e.g., [35, 19]) that the greedy solution does not exceed the length of an optimal solution plus the length of an optimal cycle cover. This has interesting counterparts in the hierarchical graphs.

- Note that an optimal cycle cover in the overlap graph can be constructed by a straightforward greedy algorithm: keep taking heavy edges till the cycle cover is constructed. The proof of correctness of this algorithm relies on the Monge inequality. Interestingly, to construct an optimal cycle cover in the hierarchical graph, it suffices to invoke the Greedy Hierarchical Algorithm with lines 7–11 commented out! In a sense, the Monge inequality is satisfied in the hierarchical graph automatically as it contains more information about input strings than just its pairwise overlaps.
- As discussed in Section A, for strings of length 3 even a more general fact than Collapsing Conjecture holds: it suffices to have double edges adjacent to input strings. One simple way to force a particular solution to satisfy this property is to double every edge of it. At the same time, adding a shortest cycle cover to it is guaranteed to be as good.
- Hence, the more general version of the Collapsing Conjecture is the following: take any solution, add any cycle cover to it, and collapse; the result is always the same. We tested this stronger conjecture and did not find any counter-examples.

References

1 Eric Bax and Joel Franklin. A Finite-Difference Sieve to Count Paths and Cycles by Length. *Inf. Process. Lett.*, 60:171–176, 1996.

2 Richard Bellman. Dynamic Programming Treatment of the Travelling Salesman Problem. *J. ACM*, 9:61–63, 1962.

3 Avrim Blum, Tao Jiang, Ming Li, John Tromp, and Mihalis Yannakakis. Linear approximation of shortest superstrings. In *STOC 1991*, pages 328–336. ACM, 1991.

4 Bastien Cazaux, Samuel Juhel, and Eric Rivals. Practical lower and upper bounds for the Shortest Linear Superstring. In *SEA 2018*, volume 103, pages 18:1–18:14. LIPIcs, 2018.

5 Bastien Cazaux and Eric Rivals. A linear time algorithm for Shortest Cyclic Cover of Strings. *J. Discrete Algorithms*, 37:56–67, 2016.

6 Bastien Cazaux and Eric Rivals. Hierarchical Overlap Graph. *arXiv preprint*, 2018. `arXiv:1802.04632`.

7 Bastien Cazaux and Eric Rivals. Relationship between superstring and compression measures: New insights on the greedy conjecture. *Discrete Appl. Math.*, 245:59–64, 2018.

8 John Gallant. *String compression algorithms*. PhD thesis, Princeton, 1982.

9 John Gallant, David Maier, and James A. Storer. On finding minimal length superstrings. *J. Comput. Syst. Sci.*, 20(1):50–58, 1980.

10 Theodoros P. Gevezes and Leonidas S. Pitsoulis. *The shortest superstring problem*, pages 189–227. Springer, 2014.

11 Collapsing Superstring Conjecture. GitHub repository. `https://github.com/alexanderskulikov/greedy-superstring-conjecture`, 2018.

12 Alexander Golovnev, Alexander S. Kulikov, and Ivan Mihajlin. Approximating shortest superstring problem using de Bruijn graphs. In *CPM 2013*, pages 120–129. Springer, 2013.

13 Alexander Golovnev, Alexander S. Kulikov, and Ivan Mihajlin. Solving SCS for bounded length strings in fewer than 2^n steps. *Inf. Process. Lett.*, 114(8):421–425, 2014.

14 Michael Held and Richard M. Karp. The Traveling-Salesman Problem and Minimum Spanning Trees. *Math. Program.*, 1:6–25, 1971.

15 Haim Kaplan and Nira Shafrir. The greedy algorithm for shortest superstrings. *Inf. Process. Lett.*, 93(1):13–17, 2005.

16 Richard M. Karp. Dynamic Programming Meets the Principle of Inclusion and Exclusion. *Oper. Res. Lett.*, 1(2):49–51, 1982.

17 Samuel Kohn, Allan Gottlieb, and Meryle Kohn. A Generating Function Approach to the Traveling Salesman Problem. In *ACN 1977*, pages 294–300, 1977.

18 Alexander S. Kulikov, Sergey Savinov, and Evgeniy Sluzhaev. Greedy conjecture for strings of length 4. In *CPM 2015*, pages 307–315. Springer, 2015.

19 Uli Laube and Maik Weinard. Conditional inequalities and the shortest common superstring problem. *Int. J. Found. Comput. Sci.*, 16(06):1219–1230, 2005.

20 Gaspard Monge. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences de Paris*, 1781.

21 Marcin Mucha. A tutorial on shortest superstring approximation, 2007.

22 Marcin Mucha. Lyndon Words and Short Superstrings. In *SODA 2013*, pages 958–972. SIAM, 2013.

23 Katarzyna Paluch. Better approximation algorithms for maximum asymmetric traveling salesman and shortest superstring. *arXiv preprint*, 2014. `arXiv:1401.3670`.

24 Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. An Eulerian path approach to DNA fragment assembly. *Proc. Natl. Acad. Sci. U.S.A.*, 98(17):9748–9753, 2001.

25 Eric Rivals and Bastien Cazaux. Superstrings with multiplicities. In *CPM 2018*, volume 105, pages 21:1–21:16, 2018.

26 Heidi J. Romero, Carlos A. Brizuela, and Andrei Tchernykh. An experimental comparison of two approximation algorithms for the common superstring problem. In *ENC 2004*, pages 27–34. IEEE, 2004.

27 Sartaj Sahni and Teofilo Gonzalez. P-Complete Approximation Problems. *J. ACM*, 23:555–565, 1976.

28 James A. Storer. *Data compression: methods and theory*. Computer Science Press, Inc., 1987.

29 Ola Svensson, Jakub Tarnawski, and László A. Végh. A constant-factor approximation algorithm for the asymmetric traveling salesman problem. In *STOC 2018*, pages 204–213. ACM, 2018.

30 Jorma Tarhio and Esko Ukkonen. A greedy approximation algorithm for constructing shortest common superstrings. *Theor. Comput. Sci.*, 57(1):131–145, 1988.

31 Jonathan S. Turner. Approximation algorithms for the shortest common superstring problem. *Inf. Comput.*, 83(1):1–20, 1989.

32 Esko Ukkonen. A linear-time algorithm for finding approximate shortest common superstrings. *Algorithmica*, 5(1-4):313–323, 1990.

33 Michael S. Waterman. *Introduction to computational biology: maps, sequences and genomes*. CRC Press, 1995.

34 Collapsing Superstring Conjecture. Webpage. `http://compsciclub.ru/scs/`, 2018.

35 Maik Weinard and Georg Schnitger. On the greedy superstring conjecture. *SIAM J. Discrete Math.*, 20(2):502–522, 2006.

A Proof of Collapsing Conjecture for Strings of Length 3

In this section, we show that the Collapsing Conjecture holds for the special case when input strings have length at most three. Remarkably, this follows from a more general theorem stated below.

▶ **Theorem 6.** *Let $\mathcal{S}$ contain strings of length at most 3 and let L be an Eulerian solution that for each $s \in \mathcal{S}$ contains at least two copies of arcs* $(\mathrm{pref}(s), s)$ *and* $(s, \mathrm{suff}(s))$*. Then* $CA(\mathcal{S}, L) = GHA(\mathcal{S})$.

It is not difficult to see that the theorem indeed implies the Collapsing Conjecture: clearly, $L = D \sqcup D$, where D is an Eulaerian solution, satisfies the condition. Moreover, this also works for $L = D_1 \sqcup D_2$, where D_1, D_2 are arbitrary Eulerian solutions, and for $L = D \sqcup CC$, where CC is a cycle cover, i.e., a set of cycles that go through all input strings. The main difference between an Eulerian solution and a cycle cover is that the later is not required to be connected. For this reason, any Eulerian solution is also a cycle cover (but not vice versa) and hence an *optimal* cycle cover is definitely not longer than an optimal Eulerian solution: $OPT \le OPTCC$. Hence, Theorem 6 says that the result of GHA is not just no longer than $2 \cdot OPT$, but even no longer than $OPT + OPTCC$.

Before proving Theorem 6, we introduce some notation and prove two auxiliary results. Recall that the Collapsing Algorithm processes the nodes level by level. Denote by L_i an intermediate Eulerian solution right before is starts collapsing the nodes at level i (that is, in L_i all the nodes at levels $> i$ are already collapsed). For an arbitrary Eulerian solution U, by $\mathrm{above}(U, i)$ denote the part of U that lies above the level i: $\mathrm{above}(U, i) = \{(u, v) \in U \colon |u|, |v| \ge i\}$. We show that $\mathrm{above}(D, i) = \mathrm{above}(L_i, i)$ for every i. This is enough since then

$$CA(\mathcal{S}, L) = \mathrm{above}(L_0, 0) = \mathrm{above}(D, 0) = GHA(\mathcal{S})\,.$$

▶ **Lemma 7.** *Let w be a walk from u to v in an Eulerian solution with all its nodes at levels $\le k$. Consider a single collapsing step for a node t that is either an* intermediate *node of p at level k or is a node at level $< k$ that do not belong to p. Then w is still a walk from u to v in the resulting solution.*

Proof. Indeed, if t does not belong to w, then collapsing it does not change w at all. Otherwise t is an intermediate node of w at level k. Since w does not have any node above level k, w goes through $(\mathrm{pref}(t), t), (t, \mathrm{suff}(t))$. Clearly, collapsing t keeps w a walk. ◀

▶ **Lemma 8.** *Let v be a node in L_2 at level $1 \le l \le 2$ (i.e., $l = |v|$). Then there is a walk from ε to v and a walk from ε to v in L_l that does not contain nodes at level 3.*

Proof. We start by proving that there is a walk from v to ε for $|v| = 2$ (the existence of a walk from ε to v is proved in a similar fashion).

Consider a walk w from v to ε in L (there is such a walk as L is an Eulerian solution). All repeated nodes in w may be removed, so one may assume that w passes through its nodes at level 3 exactly once. Then, it is sufficient to show that each such node is collapsed.

Consider a node s of w at level 3 and a pair of arcs $(\mathrm{pref}(s), s), (s, \mathrm{suff}(s)) \in w$ going through it. If s is not at input string (i.e., $s \notin \mathcal{S}$), then CA collapses this pair of arcs and this does not disconnect w. On the other hand, if s is an input string ($s \in \mathcal{S}$), then there are two copies of $(\mathrm{pref}(s), s), (s, \mathrm{suff}(s))$ in L. At least one copy of this pair is collapsed in L and therefore belongs to L_2.

The statement for v with $|v| = 1$ follows from Lemma 7. ◀

Proof of Theorem 6. As discussed above, it suffices to prove that $\mathrm{above}(D, i) = \mathrm{above}(L_i, i)$ for every $i = 2, 1, 0$.

Level $i = 2$. The base case $i = 2$ is straightforward: clearly, the Collapsing Algorithm leaves exactly one copy of arcs $(\mathrm{pref}(s), s)$ and $(s, \mathrm{suff}(s))$ for every $s \in \mathcal{S}$ and fully collapses all other nodes at level 3. Then, $\mathrm{above}(L_2, 2) = \mathrm{above}(D, 2)$ as $(\mathrm{pref}(s), s), (s, \mathrm{suff}(s))$ for $s \in \mathcal{S}$ are the only edges between levels 2 and 3 in D.

Level $i = 1$. Note that $\mathrm{above}(L_2, 2) \subseteq L_2$ and L_2 is an Eulerian cycle. Hence, $\mathrm{above}(L_2, 2)$ is a collection of walks. Consider such a walk w and consider two cases.

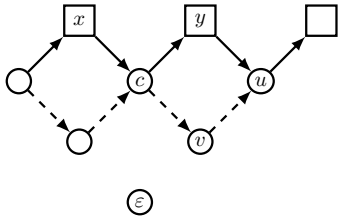
- w is a closed walk. Let v be the lexicographically largest node of w at level 2. What we want to show is that in L_1 this closed walk w is connected to the rest of L_1 through a pair of arcs $(\mathrm{pref}(v), v)$ $(v, \mathrm{suff}(v))$ only.
  By Lemma 8, there is a path from v to ε in L_2 and hence $(v, \mathrm{suff}(v)) \in L_2$; similarly, $(\mathrm{pref}(v), v) \in L_2$. Since v is lexicographically largest at level 2 in w, when CA starts processing the node v, all other nodes at level 2 in w are fully collapsed, i.e., for any such node u, $(\mathrm{pref}(u), u) \notin L_1$ and $(u, \mathrm{suff}(u)) \notin L_1$. Moreover, CA does not collapse the pair of arcs $(\mathrm{pref}(v), v), (v, \mathrm{suff}(v))$ as this would disconnect w from the rest of the solution.
- w is not closed. Denote by v_1 and v_k its first and last nodes. All other nodes of w in $\mathrm{above}(L_2, 2)$ are balanced. What we want to show is that in L_1 the only edges between levels 1 and 2 that connect w to the rest of the solution are $(\mathrm{pref}(v_1), v_1)$ and $(v_k, \mathrm{suff}(v_k))$.
  We prove this for v_k (for v_1 is it shown similarly). By Lemma 8, there is a path from v_k to ε in L_2 and hence $(v_k, \mathrm{suff}(v_k)) \in L_2$. The algorithm CA always works with an Eulerian solution and hence every node is balanced at every stage (i.e., its in-degree is equal to its out-degree). This means that $(v_k, \mathrm{suff}(v_k)) \in L_1$ and that all intermediate nodes of w are not connected to level 1 nodes in L_1.

Level $i = 0$. Note that $\mathrm{above}(L_1, 1)$ is a collection of walks. The case of a non-closed walk in this case is easy as it must be connected to ε directly. For this reason, we focus on a closed walk w in $\mathrm{above}(L_1, 1)$.

We show that for every node v of w with $|v| = 1$, L_1 contains arcs (ε, v) and (v, ε) (recall that for $|v| = 1$, $\mathrm{pref}(v) = \mathrm{suff}(v) = \varepsilon$). This suffices as then CA (when processing level one nodes) collapses all nodes of w at level 1 except for the lexicographically largest one, and this is exactly how w is connected to ε in D_0. Below, we show that $(\varepsilon, v) \in L_1$. It then follows that $(v, \varepsilon) \in L_1$ (as L_1 must be Eulerian).

Lemma 8 guarantees that L_1 contains a path from v to ε that does not contain nodes at level 3. If the first arc of this path goes down to ε, then there is nothing to prove. Hence, consider a case when the first arc goes up to a node u (and hence $v = \mathrm{pref}(u)$). The next arc then must go down to $\mathrm{suff}(u)$. Hence, $(\mathrm{pref}(u), u), (u, \mathrm{suff}(u)) \in D_1$. This

may happen in two cases only: either u is an input string (i.e., $u \in \mathcal{S}$) or u is the last chance of its component to be connected to the rest of the solution (i.e., exactly for this reason GHA added these two edges to the solution). The former case is straightforward: then there were at least two copies of the arcs $(\mathrm{pref}(u), u), (u, \mathrm{suff}(u))$ and CA collapsed at least one copy. Let us then focus on the latter case.
Let $x, y \in \mathcal{S}$ be such that $u = \mathrm{suff}(y)$ and $c := \mathrm{suff}(x) = \mathrm{pref}(y)$, see the picture below (solid arcs belong to L, dashed arc belong to L_2).



Note that

$$v = \mathrm{pref}(u) = \mathrm{pref}(\mathrm{suff}(y)) = \mathrm{suff}(\mathrm{pref}(y)) = \mathrm{suff}(c)\,.$$

Hence, $(c, v), (v, u) \in L_2$ (resulting from collapsing at least one pair of arcs $(c, y), (y, u) \in L$). L_2 also contains a pair of arcs $(\mathrm{pref}(x), \mathrm{suff}(\mathrm{pref}(x))), (\mathrm{suff}(\mathrm{pref}(x)), c)$. When processing the node c, CA collapses the pair of arcs $(\mathrm{pref}(c), c), (c, v)$ as there is an arc (v, u). Hence, $(\varepsilon, v) \in L_1$, as required. (It may be the case that $x = y$. Then $x = \mathtt{aaa}$, $v = \{a\}$. Then the first pair of arcs of the considered path is $\mathtt{a} \to \mathtt{aa} \to \mathtt{a}$ and one may just drop them.)

As a final remark, note that if a walk $w \in \mathrm{above}(L_i, i)$ is connected to the rest of a solution through some a of arcs $(\mathrm{pref}(v), v), (u, \mathrm{suff}(u))$ (v and u may coincide), then any other balanced node in w at level i can be fully collapsed, as every such collapse, thanks to Lemma 7, does not disconnect w or any other walk from $\mathrm{above}(L_i, i)$ from the rest of the solution. ◀

B Greedy Hierarchical Algorithm and Special Cases of SCS

B.1 Strings of Length 2

Gallant et al. [9] show that SCS on strings of length 3 is **NP**-hard, but SCS on strings of length at most 2 is solvable in polynomial time. In this section we show that GHA finds an optimal solution in this case as well. We note that the standard Greedy Algorithm does not necessarily find an optimal solution in this case. For example, if $\mathcal{S} = \{\mathtt{ab}, \mathtt{ba}, \mathtt{bb}\}$, the Greedy Algorithm may first merge `ab` and `ba`, which would lead to a suboptimal solution `ababb` (recall also Figure 6).

First, we can assume that all input strings from $\mathcal{S}$ have length exactly 2. Indeed, since we assume that no input string is a substring of another input string, all strings of length 1 are unique symbols which do not appear in other strings. Take any such s_i of length 1. The optimal superstring length for $\mathcal{S}$ is k if and only if the optimal superstring length for $\mathcal{S} \setminus \{s_i\}$ is $k - 1$. The Greedy Hierarchical Algorithm has the same behavior: In Step 2, GHA will include the arcs $(\varepsilon, s_i), (s_i, \varepsilon)$ in the solution, and it will never touch the vertex s_i again (because it is balanced and connected to ε). Thus, s_i adds 1 to the length of the Greedy Hierarchical Superstring as well. By the same reasoning, we can assume that each string of length two is primitive, i.e., contains two distinct symbols.

When considering primitive strings $\mathcal{S} = \{s_1, \ldots, s_n\}$ of length exactly 2, it is convenient to introduce the following directed graph $G = (V, E)$, where V contains a vertex for every symbol which appears in strings from $\mathcal{S}$. The graph has $|E| = n$ arcs corresponding to n input strings: for every string $s_i = ab$, there is an arc from a to b. It is known [9] that the length of an optimal superstring in this case is $n + k$ where k is the minimum number such that E can be decomposed into k directed paths, or, equivalently:

▶ **Proposition 9** ([9]). *Let G be the graph defined above, and let $G_1 = (V_1, E_1), \ldots, G_c = (V_c, E_c)$ be its weakly connected components. Then the length of an optimal superstring is*

$$n + \sum_{i=1}^{c} \max\left(1, \sum_{v \in V_i} \frac{|\operatorname{indegree}(v) - \operatorname{outdegree}(v)|}{2}\right). \tag{2}$$

We will now show that in this case, GHA finds an optimal solution.

▶ **Lemma 10.** *Let $\mathcal{S} = \{s_1, \ldots, s_n\}$ be a set of strings of length at most 2, and let s be an optimal superstring for $\mathcal{S}$. Then $GHA(\mathcal{S})$ returns a superstring of length $|s|$.*

Proof. We showed above that it suffices to consider the case of n primitive strings $\{s_1, \ldots, s_n\}$ of length exactly 2. For $1 \leqslant i \leqslant n$, let $s_i = a_i b_i$, where $a_i \neq b_i$. Consider the partial greedy hierarchical solution D after the Step 2 of the GHA algorithm: $D = \{(a_i, a_i b_i), (a_i b_i, b_i) : 1 \leqslant i \leqslant n\}$. (We abuse notation by identifying the set of arcs D with the graph induced by D.) This partial solution has n up-arcs, so its current weight is n.

Note that by the definition of the graph G above, G contains an arc (a, b) if and only if D has the arcs (a, ab) and (ab, b) of the graph HG. Thus, the indegree (outdegree) of a vertex a in G equals the indegree (outdegree) of the vertex a in the partial solution D. Also, two vertices a and b of G belong to one weakly connected component in G if and only if they belong to one weakly connected component in D. Therefore, the expression (2) in G has the same value in the partial solution graph D. (Indeed, the vertices of D corresponding to strings of length 2 are balanced and do not form weakly connected components.)

Now we proceed to Steps 3–11 of GHA. GHA will go through all strings of length 1, and add $|\operatorname{indegree}(v) - \operatorname{outdegree}(v)|$ arcs for each unbalanced vertex v. The Steps 8–11 ensure that each weakly connected component adds at least a pair of arcs. Since exactly a half of added arcs are up-arcs, we have increased the weight of the partial solution D by

$$\sum_{i=1}^{c} \max\left(1, \sum_{v \in V_i} \frac{|\operatorname{indegree}(v) - \operatorname{outdegree}(v)|}{2}\right). \qquad \blacktriangleleft$$

B.2 Spectrum of a String

By a k-spectrum of a string s (of length at least k) we mean a set of all substrings of s of length k. Pevzner et al. [24] give a polynomial time exact algorithm for the case when the input strings form a spectrum of an unknown string. We show that GHA also finds an optimal solution in this case.

▶ **Lemma 11.** *Let $\mathcal{S} = \{s_1, \ldots, s_n\}$ be a k-spectrum of an unknown string s. Then $GHA(\mathcal{S})$ returns a superstring of length at most $|s|$.*

Proof. Since s has n distinct substrings of length k, $|s| \geqslant n + k - 1$. We will show that GHA finds a superstring of length $n + k - 1$. After Step 2 of GHA, the partial solution $D = \{(\operatorname{pref}(s), s), (s, \operatorname{suff}(s)) : s \in \mathcal{S}\}$. In particular, D is of weight n. For $1 \leqslant i \leqslant k - 1$,

let u_i be the first i symbols of s, and let v_i be the last i symbols of s. Note that u_{k-1} and v_{k-1} are the only unbalanced vertices of the partial solution D after Step 2: all other strings of length $k-1$ appear equal number of times as prefixes and suffixes of strings from $\mathcal{S}$. Therefore, while processing the level $\ell = k-1$, GHA will add one arc to each of the vertices u_{k-1} and v_{k-1}, and will not add arcs to other strings of length $k-1$.

In general, while processing the level $\ell = i$, GHA adds one up-arc to u_i and one down-arc to v_i. In order to show this, we consider two cases. If $u_i \neq v_i$, then u_i has an incoming arc from the previous step and does not have outgoing arcs, therefore GHA adds an up-arc to u_i in Step 6. Similarly, GHA adds a down-arc from v_i. Note that there are no other strings of length $i < k-1$ in the partial solution, so the algorithm moves to the next level. In the case when $u_i = v_i$, we have that all vertices are balanced, but the string u_i is now the shortest string in this only connected component $\mathcal{C}$ of the graph. Therefore, for $i > 0$ we have $\varepsilon \notin \mathcal{C}$, and GHA adds an up- and down-arc to u_i in Step 11.

We just showed that GHA solution for a k-spectrum of a string has the initial set of arcs $D = \{(\mathrm{pref}(s), s), (s, \mathrm{suff}(s))\colon s \in \mathcal{S}\}$, and also the arcs $\{(u_{i-1}, u_i), (v_i, v_{i-1})\colon 1 \leqslant i \leqslant k-1\}$. Thus, the total number of up-arcs (and the weight of the solution) is $n + k - 1$. ◀

B.3 Tough Dataset

There is a well-known dataset consisting of just three strings where the classical greedy algorithm produces a superstring that is almost twice longer than an optimal one: $s_1 = \mathtt{cc}(\mathtt{ae})^n$, $s_2 = (\mathtt{ea})^{n+1}$, $s_3 = (\mathtt{ae})^n\mathtt{cc}$. Since $\mathrm{overlap}(s_1, s_3) = 2n$, while $\mathrm{overlap}(s_1, s_2) = \mathrm{overlap}(s_2, s_3) = 2n-1$, the greedy algorithm produces a permutation (s_1, s_3, s_2) (or (s_2, s_1, s_3)). I.e., by greedily taking the massive overlap of length $2n$ it loses the possibility to insert s_2 between s_1 and s_3 and to get two overlaps of size $2n-1$. The resulting superstring has length $4n+6$. At the same time, the optimal superstring corresponds to the permutation (s_1, s_2, s_3) and has length $2n+8$.

The algorithm GHA makes a similar mistake on this dataset, see Figure 12. When processing the node $(\mathtt{ea})^n$, GHA does not add two lower arcs to it and misses a chance to connect two components. It is then forced to connect these two components through ε. This example shows that GHA also does not give a better than 2-approximation for SCS.

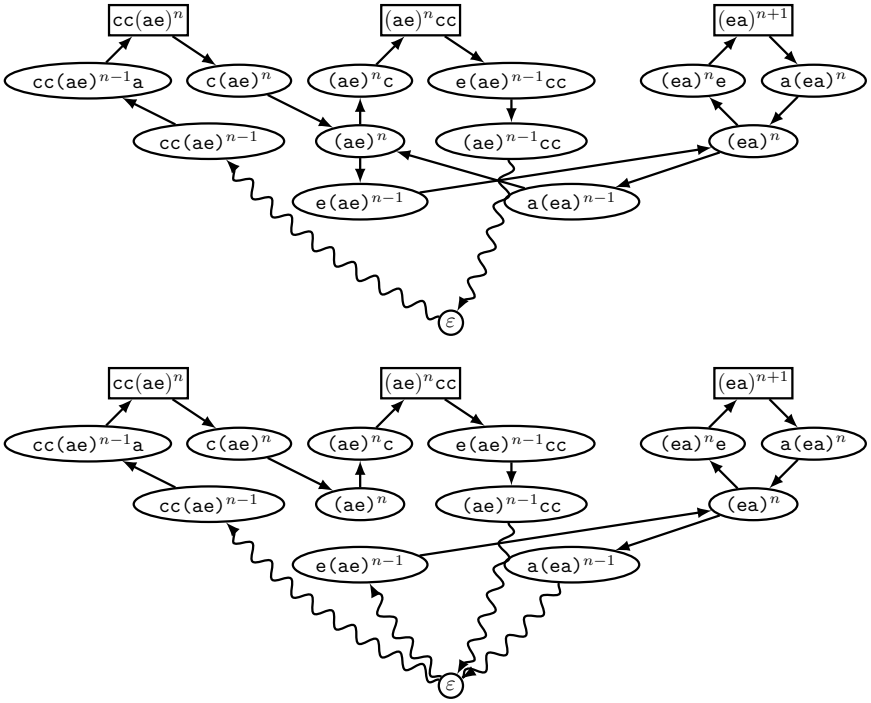


Figure 12 Top: optimal solution for the dataset $\{\mathtt{cc}(\mathtt{ae})^n, (\mathtt{ea})^{n+1}, (\mathtt{ae})^n\mathtt{cc}\}$. Bottom: solution constructed by GHA.

Improved Algorithms for Time Decay Streams

Vladimir Braverman
Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA
vova@cs.jhu.edu

Harry Lang
MIT CSAIL, Cambridge, MA, USA
harry1@mit.edu

Enayat Ullah
Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA
enayat@jhu.edu

Samson Zhou
School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA
samsonzhou@gmail.com

Abstract

In the time-decay model for data streams, elements of an underlying data set arrive sequentially with the recently arrived elements being more important. A common approach for handling large data sets is to maintain a *coreset*, a succinct summary of the processed data that allows approximate recovery of a predetermined query. We provide a general framework that takes any offline-coreset and gives a time-decay coreset for polynomial time decay functions.

We also consider the exponential time decay model for k-median clustering, where we provide a constant factor approximation algorithm that utilizes the online facility location algorithm. Our algorithm stores $O(k \log(h\Delta) + h)$ points where h is the half-life of the decay function and Δ is the aspect ratio of the dataset. Our techniques extend to k-means clustering and M-estimators as well.

2012 ACM Subject Classification Theory of computation → Facility location and clustering; Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Streaming algorithms, approximation algorithms, facility location and clustering

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.27

Category APPROX

Acknowledgements This material is based upon work supported in part by the National Science Foundation under Grant No. 1447639, by the Google Faculty Award and by DARPA grant N660001-1-2-4014. Its contents are solely the responsibility of the authors and do not represent the official view of DARPA or the Department of Defense.

1 Introduction

The *streaming model* of computation has become an increasingly popular model for processing massive datasets. In this model, the data is presented sequentially, and the objective is to answer some pre-defined query. The overwhelmingly large size of the dataset imposes a number of restrictions on any algorithm designed to answer the pre-defined query. For example, a streaming algorithm is permitted only a few passes, or in many cases, only a single pass over the data. Moreover, the algorithm should also use space sublinear in, or even logarithmic in, the size of the data. For more details on the background and applications of the streaming model, [4, 45, 1] provide excellent surveys.


© Vladimir Braverman, Harry Lang, Enayat Ullah, and Samson Zhou;
licensed under Creative Commons License CC-BY



Informally, a coreset for a given problem is a small *summary* of the dataset such that the *cost* of any candidate solution on the coreset is approximately the same as the cost in the original set. Coresets have been used in a variety of problems, including generalized facility locations [29], k-means clustering [31, 9], principal component analysis [33], and ℓ_p-regression [24]. Coresets also have a number of applications in distributed models (see [39, 44, 6, 3], for example). To maintain the coresets throughout the data stream, one possible approach is the so called merge-and-reduce method, in which the multiple sets may be adjusted and combined. Several well-known coreset constructions [37, 17] for the k-median and k-means problems are based on the merge-and-reduce paradigm.

1.1 Motivation

Many applications discard obsolete data, choosing to favor relatively recent data to base their queries. This motivates the *time decay* model, in which there exists a function w so that the weight of the t^{th} most recent item is $w(t)$. Note that this is a generalization of both the *insertion-only* streaming model, where $w(t) = 1$ for all t, and the *sliding-window* model, where $w(t) = 1$ for the most recent W items, and $w(t) = 0$ for $t > W$. In this paper, we study the problem of maintaining coresets over a polynomial decay model, where $w(t) = \frac{1}{t^s}$ for some parameter $s > 0$, and an exponential decay model, where $w(t) = 2^{\frac{T-t+1}{h}}$ at time T for some *half-life* parameter $h > 0$.

Although exponential decay model is well-motivated by natural phenomena that exhibit half-life behavior, [20] notices that exponential decay and the sliding window model is often insufficient for many applications because the decay occurs too quickly and suggests that polynomial decay may be a reasonable alternative for some applications, such as availability of network links. For example, consider a network link that fails at every time between 10 and 60 and a second network link that fails once at time 75. Intuitively, it seems like the second link should be better, but under many parameters, the exponential decay model and sliding window model will both agree that the first link is better. Fortunately, under the polynomial decay model, events that occur near the same time have approximately the same weight, and we will obtain some view in which the first link is preferred [40]. In practice, time decay functions have been used in natural language understanding to give more importance to recent utterances than the past ones [47].

Organization. The rest of the paper is organized as follows. In Section 2, we summarize the main results of the paper and the algorithmic approaches. In Section 3, we discuss the related work, and in Section 4, we formalize the problem and discuss the preliminaries required. In Sections 5 and 6, we handle the polynomial and exponential decay, respectively, in detail, wherein we present the algorithmic details as well as the complete analysis.

2 Our Contributions

We summarize our results and give a high-level idea of our approach for problems in the polynomial and exponential decay models in the following subsections respectively. The reader is encouraged to go through Sections 5 and 6 for details.

2.1 Polynomial decay

In the polynomial decay model, a stream of points P arrives sequentially and the weight of the t^{th} most recent point, denoted as $w(t)$, is $w(t) = \frac{1}{t^s}$ where $s > 0$ is a given constant parameter of the decay function. We first state a theorem that shows that we can use an offline coreset construction mechanism to give a coreset for the polynomial decay model.

▶ **Theorem 1.** *Given an algorithm that takes a set of n points as input and constructs an ϵ-coreset of $F(n,\epsilon)$ points in $\mathcal{O}(nT(\epsilon))$ time, there exists a polynomial decay algorithm that maintains an ϵ-coreset while storing $\mathcal{O}\left(\epsilon^{-1}\log n\, F\left(n,\frac{\epsilon}{\log n}\right)\right)$ points and with update time $\mathcal{O}\left(\epsilon^{-1}\ \log n\ F(n,\epsilon)\,T(\epsilon/\log n)\right)$.*

Theorem 1 applies to any time-decay problem on data streams that admits an approximation algorithm using coresets. Among its applications are the problems of k-median and k-means clustering, M-estimator clustering, projective clustering, and subspace approximation. We list a few of these results in Table 1. Our result is a generalization of the *vanilla* merge-and-reduce approach used to convert offline coresets to streaming counterparts. In particular, plugging in $s = 0$, we get the vanilla streaming model, and the theorem recovers the corresponding guarantees.

Table 1 Coresets for some problems in polynomial decay streams.

Problem	Coreset size	Offline algorithm
Metric k-median clustering	$\mathcal{O}\left(\frac{s}{\epsilon^3}k\log k\log^4 n\right)$	[30]
Metric k-means clustering	$\mathcal{O}\left(\frac{s}{\epsilon^3}k\log k\log^4 n\right)$	[9]
Metric M-estimator	$\mathcal{O}\left(\frac{s}{\epsilon^3}k\log k\log^4 n\right)$	[9]
j^{th} subspace approximation	$\mathcal{O}\left(\frac{j^2 s}{\epsilon^4}\log^8 n\log\left(\frac{\log n}{\epsilon}\right)\right)$	[30]
Low rank approximation	$\mathcal{O}\left(\frac{s}{\epsilon^2}kd\log n\right)$	[34]

Approach. A natural starting point would be to attempt to generalize existing sliding window algorithms to time decay models. These algorithms typically use a histogram data structure [14], in which multiple instances of streaming algorithms are started at various points in time, one of which well-approximates the objective evaluated on the data set represented by the sliding window. However, generalizing these histogram data structures to time-decay models does not seem to work since the weights of all data points changes upon each new update in time-decay model, whereas streaming algorithms typically assume static weights for each data point.

Instead, our algorithm partitions the stream into blocks, where each block represents a disjoint collection of data point between certain time points. Each arriving element initially begins as its own block, containing one element. The algorithm maintains an unweighted coreset for each block, and merges blocks (i.e corresponding coresets) as they become older. However, at the end, each block is to be weighted according to some function, and so the algorithm chooses to merge blocks when the weights of the blocks become "close". Thus, a coreset for each block will represent the set of points well, as the weights of the points in each block do not differ by too much.

2.2 Exponential decay

We also provide an algorithm that achieves a constant approximation for k-median clustering in the exponential decay model. Our guarantees also extend to k-means clustering and M-estimators.

Given a set P of points in a metric space, let Δ denote its aspect ratio i.e the ratio between the largest and (non-zero) smallest distance between any two points in P. The weight of the t^{th} most recent point at time T is $w(t) = 2^{\frac{T-t+1}{h}}$ where $h > 0$ is the half-life parameter of the exponential decay function.

▶ **Theorem 2.** *There exists a streaming algorithm that given a stream P of points with exponentially decaying weights, with aspect ratio Δ and half-life h, produces an $\mathcal{O}(1)$-approximate solution to k-median clustering. The algorithm runs in $\mathcal{O}(nk \log(h\Delta))$ time and uses $\mathcal{O}(k \log(h\Delta) + h)$ space.*

Approach. Although our previous framework will work for other decay models, the algorithm may use prohibitively large space. The intuition behind the polynomial decay approach is that a separate coreset is maintained for each set of points that roughly have the same weight. In other words, the previous framework maintains a separate coreset each time the weight of the points decrease by some constant amount, so that if R is the ratio between the largest weight and the smallest weight, then the total number of coresets stored by the algorithm is roughly $\log R$. In the polynomial decay model, the number of stored coresets is $\mathcal{O}(\log n)$, but in the exponential decay model, the number of stored coresets would be $\mathcal{O}(n)$, which would no longer be sublinear in the size of the input. Hence, we require a new approach for the exponential decay model.

Instead, we use the online facility location (OFL) algorithm of Meyerson [43] as a subroutine to solve k-median clustering in the exponential decay model. In the online facility location problem, we are given a metric space along with a facility cost for each point/location that appears in the data stream. The objective is to choose a (small) number of facility locations to minimize the total facility cost plus the service cost, where the service cost of a point is its distance to the closest facility. For more details, please see Section 6.

Our algorithm for the exponential time decay model proceeds on the data stream, working in phases. Each phase corresponds to an increasing "guess" for the value of the *cost* of the optimal clustering. Using this guess, each phase queries the corresponding instance of OFL. If the guess is *correct*, then the subroutine selects a bounded number of facilities. On the other hand, if either the cost or the number of selected facilities surpasses a certain quantity, then the guess for the optimal cost must be incorrect, and the algorithm triggers a phase change. Upon a phase change, our algorithm uses an offline k-median clustering algorithm to cluster the facility set and produces exactly k points. It then runs a new instance of OFL with a larger guess, and continues processing the data stream.

However, there is a slight subtlety in this analysis. The number of points stored by OFL is dependent on the weights of the point. In an exponential decay function, the ratio between the largest weight and smallest weight of points in the data set may be exponentially large. Thus to avoid OFL from keeping more than a logarithmic number of points, we force OFL to terminate after seeing $\log(h\Delta)$ points during a phase. Furthermore, we store points verbatim until we see $k + h$ *distinct* points, upon whence we will trigger a phase change. We show that forcing this phase change does indeed correspond with an increase in the guess of the value for the optimal cost.

3 Related Work

The first insertion-only streaming algorithm for the k-median clustering problem was presented in 2000 by Guha, Mishra, Motwani, and O'Callaghan [36]. Their algorithm uses $\mathcal{O}(n^\epsilon)$ space for a $2^{\mathcal{O}(1/\epsilon)}$ approximation, for some $0 < \epsilon < 1$. Subsequently, Charikar *et al* [16] present an $\mathcal{O}(1)$-approximation algorithm for k-means clustering that uses $\mathcal{O}\left(k \log^2 n\right)$ space. Their algorithm uses a number of phases, each corresponding to a different guess for the value of the cost of optimal solution. The guesses are then used in the online facility location (OFL) algorithm of [43], which provides a set of centers whose number and cost allows the algorithm

to reject or accept the guess. This technique is now one of the standard approaches for handling k-service problems. Braverman *et al* [13] improve the space usage of this technique to $\mathcal{O}(k \log n)$. [11] and [12] develop algorithms for k-means clustering on sliding windows, in which expired data should not be included in determining the cost of a solution.

Another line of approach for k-service problems is the construction of coresets, in particular when the data points lie in the Euclidean space. Har-Peled and Mazumdar [37] give an insertion-only streaming algorithm for k-medians and k-means that provides a $(1+\epsilon)$-approximation, using space $\mathcal{O}\left(k\epsilon^{-d}\log^{2d+2} n\right)$, where d is the dimension of the space. Similarly, Chen [17] introduced an algorithm using $\mathcal{O}\left(k^2 d\epsilon^{-2}\log^8 n\right)$ space, with the same approximation guarantees.

Cohen and Strauss [20] study problems in time-decaying data streams in 2003. There are a number of results [40, 22, 21, 23] in this line of work, but the most prominent time-decay model is the sliding window model. Datar *et al* [25] introduced the exponential histogram as a framework in the sliding window for estimating statistics such as count, sum of positive integers, average, and ℓ_p norms. This initiated an active line of research, including improvements to count and sum [35], frequent itemsets [18, 10], frequency counts and quantiles [2, 42], rarity and similarity [26], variance and k-medians [5] and other geometric and numerical linear algebra problems [28, 15, 8].

4 Preliminaries

Let $\mathcal{X}$ be the set of possible points in a space with metric d. A weighted set is a pair (P, w) with a set $P \subset \mathcal{X}$ and a weight function $w : P \to [0, \infty)$. A query space is a tuple (P, w, f, Q) that combines a weighted set with a set Q of possible queries and a function $f : \mathcal{X} \times Q \to [0, \infty)$. A query space induces a function

$$\bar{f}(P, w, q) = \sum_{p \in P} w(p) f(p, q).$$

We now instantiate the above with some simple examples.

▶ **Example 3** (k-means)**.** Let Q be all sets of k points in $\mathbb{R}^d$, and for $C \in Q$ define $f(p, C) = \min_{c \in C} d^2(p, c)$. The k-means cost of (P, w) to C is

$$\sum_{p \in P} w(p) \min_{c \in C} d^2(p, c).$$

▶ **Example 4** (k-median)**.** Let Q be all sets of k points in $\mathbb{R}^d$, and for $C \in Q$ define $f(p, C) = \min_{c \in C} d(p, c)$. The k-median cost of (P, w) to C is

$$\sum_{p \in P} w(p) \min_{c \in C} d(p, c).$$

Note that both k-median and k-means are captured in $\bar{f}(P, w, C)$. We now define an ϵ-coreset.

▶ **Definition 5** (ϵ-coreset)**.** *A ϵ-coreset for the query space (P, w, f, Q) is a tuple (Z, u), where $Z \subseteq \mathcal{X}$ is a set of points and $u : Z \to [0, \infty)$ are their corresponding weights, such that for every $q \in Q$*

$$(1-\epsilon)\bar{f}(P, w, q) \leq \bar{f}(Z, u, q) \leq (1+\epsilon)\bar{f}(P, w, q).$$

An important property of coresets is that they are *closed* under operations like union and composition. We formalize this below.

▶ **Proposition 6** (Merge-and-reduce, [17]). *Coresets satisfy the following two properties.*

1. *If $\mathcal{S}_1$ and $\mathcal{S}_2$ are ϵ-coresets of disjoint sets $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively, then $\mathcal{S}_1 \cup \mathcal{S}_2$ is an ϵ-coreset of $\mathcal{P}_1 \cup \mathcal{P}_2$.*
2. *If $\mathcal{S}_1$ is an ϵ-coreset of $\mathcal{S}_2$ and $\mathcal{S}_2$ is a δ-coreset of $\mathcal{S}_3$, then $\mathcal{S}_1$ is a $((1+\epsilon)(1+\delta)-1)$-coreset of $\mathcal{S}_3$.*

We now define approximate triangle inequality, a property that allows us to extend our results obtained in metric spaces to ones with *semi-distance* functions. In particular, this allows us to extend results for k-median clustering to k-means and M-estimators in exponential decay streams.

▶ **Definition 7** (λ-approximate triangle inequality). *A function $d(\cdot,\cdot)$ on a space $\mathcal{X}$ satisfies the λ-approximate triangle inequality if for all $x, y, z \in \mathcal{X}$,*

$$d(x,z) \leq \lambda(d(x,y) + d(y,z)).$$

5 Polynomial Decay

We consider a time decay, wherein a point p in the stream, which arrived at time t, has weight $w(p) = (T-t+1)^{-s}$ at time $T > t$, for some parameter $s > 0$. Equivalently, the t^{th} most recent element has weight t^{-s} for some $s > 0$.

We present a general framework which, for given problem, takes an offline coreset construction algorithm and adapts it to polynomial decay streams. Our technique can be viewed as a generalization of merge-and-reduce technique of Bentley and Saxe [7]. We also briefly discuss some applications towards that end. We start with stating our main theorem for polynomial decay streams.

▶ **Theorem 8.** *Given an offline algorithm that takes a set of n points as input and constructs an ϵ-coreset of $F(n,\epsilon)$ points in $\mathcal{O}(n\,T(\epsilon))$ time, there exists a polynomial decay algorithm that maintains an ϵ-coreset while storing $\mathcal{O}\left(\epsilon^{-1} s \log n\, F(n, \epsilon/\log n)\right)$ points and with update time*

$$\mathcal{O}\left(\epsilon^{-1} s\ \log n\ F(n,\epsilon)\, T(\epsilon/\log n)\right).$$

Notation. We use $\mathbb{N}$ to denote the set of natural numbers. We use CS-RAM to denote an offline coreset construction algorithm, which given n points, constructs an ϵ-coreset in time $\mathcal{O}(n\,T(\epsilon))$ and takes space $F(n,\epsilon)$. We abuse notation by using $F(n,\epsilon)$ to also refer to the corresponding coreset.

5.1 Algorithm

We start with giving a high-level intuition of the algorithm. Given a stream of points, the algorithm implicitly maintains a partition of the streams into disjoint *blocks*. A block is a collection of consecutive points in the stream, and is represented by two positive integers a and b as $[a,b]$, where a represents the position of the first point in the block and b the last point, relative to the start of the stream. Let the set of blocks be denoted by $\mathcal{B}$. Our algorithm stores points of a given block by maintaining a coreset for the points in that block. As the stream progresses, we merge older blocks i.e. the corresponding coresets. Informally, the merge happens when the weights of the blocks become *close*.

We first define a set of integer *markers* x_i, which for a given $i \in \mathbb{N}$, depends on the decay parameter s and target ϵ. These markers dictate when to merge blocks as the stream progresses. For a given $i \in \mathbb{N}$, we define x_i to be the minimum integer greater than or equal to 2^i such that

$$\frac{1-\epsilon}{(x_i - 2^i + 1)^s} \leq \frac{1+\epsilon}{x_i^s}.$$

Equivalently, we can write $\left(\frac{x_i}{x_i - 2^i + 1}\right)^s \leq \frac{1+\epsilon}{1-\epsilon}$. Note that each of the 2^i points following x_i in the stream, has weight within $\frac{1+\epsilon}{1-\epsilon}$ times the weight of x_i. Moreover, x_i's can be exactly pre-computed from the equation and we therefore assume that these are implicitly stored by the algorithm. Each new element in the stream starts as a new block. As mentioned before, the blocks are represented by two integers $[a, b]$ and the points are stored as a coreset. When a block $[a, b]$ *reaches* x_i, then algorithm merges all of $[x_i - 2^i + 1, x_i]$ points into a single coreset. In the end, the algorithm outputs the *weighted* union of the coresets of the blocks.

To visualize this, consider the integer line, and suppose that we have x_i's marked on the positive side of the line, for example $x_1 = 2, x_2 = 4 \ldots$. The tuple indices of the blocks represent the relative position of the point in the stream, with the start being 1 and the end point being n. At the start, the stream is on the non-positive end with the first point at 0. As the time progresses, the stream moves to the right side. Therefore, when we observe the first element, it moves to the point 1. We then store it as a new block, represented by $[1, 1]$; we also simultaneously store a coreset corresponding to it. As time progresses, a block reaches x_i for some i which can be formally expressed as $a + x_i \leq n$. We then merge all blocks in the range $[a, a + 2^i - 1]$. Note that by definition of x_i, we would have observed all these elements and also we will not merge partial blocks. We present this idea in full in Algorithm 1 and intuition in Figure 1. We remark that when we construct coresets, we use an offline algorithm CS-RAM which given a set of n points P and a query space (P, w, f, q) produces an ϵ-coreset.

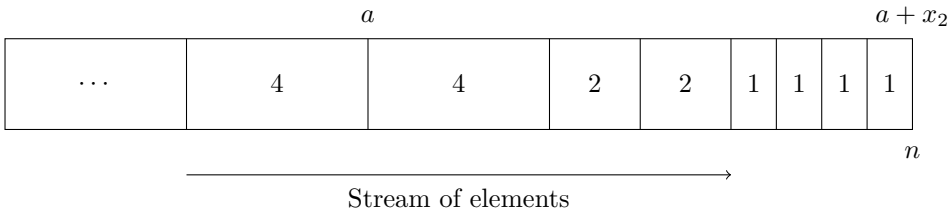


Figure 1 The algorithm merges blocks in each interval $[a, a + 2^i - i]$ for $a \leq n - x_i$.

Algorithm 1 ϵ-coreset for polynomial decaying streams.

Input: Stream P, polynomial decay function $w(t) = \frac{1}{t^s}$, for some $s > 0$, an offline coreset construction algorithm CS-RAM

Output: $(1 + \epsilon)$ coreset.

Initialize $\mathcal{B} = \emptyset$
for each element p_n of the stream **do**
  Insert $[n, n]$ into $\mathcal{B}$ as a new block and construct a coreset
  for each block $[a, b] \in \mathcal{B}$ **do**
    if $a + x_i < n$ for some i **then**
      Merge the blocks in $[a, a + 2^i - 1]$ and *reduce* to get an $\frac{\epsilon}{3 \log n}$-coreset
    end if
  end for
end for
for each block $[a, b] \in \mathcal{B}$ **do**
  Give the block weight $u(a, b) = \frac{1}{2}\left(\frac{1-\epsilon}{a^s} + \frac{1+\epsilon}{b^s}\right)$
end for

5.2 Analysis

We first show that a weighted combination of blocks gives us an ϵ-coreset. For a block $[a, b]$, let the weight of the block be denoted as $u(a, b)$. We set $u(a, b) = \bar{u}$ where $\bar{u}$ satisfies

$$\frac{1-\epsilon}{a^s} \leq \bar{u} \leq \frac{1+\epsilon}{b^s}.$$

The following lemma shows that any such $\bar{u}$ produces a 3ϵ-coreset.

▶ **Lemma 9.** *Let (Z, u) be an ϵ-coreset for (P, w, f, Q). Let $\bar{u} : Z \to [0, \infty)$ be such that $(1-\epsilon)u(z) \leq \bar{u}(z) \leq (1+\epsilon)u(z)$ for every $z \in Z$, then $(Z, \bar{u})$ is a 3ϵ-coreset for (P, w, f, Q).*

Proof. Since (Z, u) is an ϵ-coreset for (P, w, f, Q), therefore for every $q \in Q$,

$$\begin{aligned}
&(1-\epsilon)\bar{f}(P, w, q) \leq \bar{f}(Z, u, q) \leq (1+\epsilon)\bar{f}(P, w, q) \\
\iff &(1-\epsilon)\sum_{p\in P} w(p)f(p, q) \leq \sum_{z\in Z} u(z)f(z, q) \leq (1+\epsilon)\sum_{p\in P} w(p)f(p, q) \\
\iff &(1-\epsilon)^2\sum_{p\in P} w(p)f(p, q) \leq \sum_{z\in Z} \bar{u}(z)f(z, q) \leq (1+\epsilon)^2\sum_{p\in P} w(p)f(p, q).
\end{aligned}$$

Note that for $\epsilon < 1$, we have $(1-2\epsilon)\bar{f}(P, w, q) \leq (1-\epsilon)^2\bar{f}(P, w, q) \leq \bar{f}(Z, \bar{u}, q) \leq (1+\epsilon)^2\bar{f}(P, w, q) \leq (1+3\epsilon)f(P, w, q)$. Therefore $(Z, \bar{u})$ is a 3ϵ-coreset for (P, w, f, Q). ◀

Having assigned weights to the blocks, we can take the union to get the coreset of $\mathcal{B}$. For simplicity, we choose $u(a, b) = \frac{1}{2}\left(\frac{1-\epsilon}{a^s} + \frac{1+\epsilon}{b^s}\right)$ in Algorithm 1. We now present a lemma that bounds the number of blocks maintained by the algorithm.

▶ **Lemma 10.** *Given a polynomial decay stream of n points as input to Algorithm 1, the number of blocks produced is $\mathcal{O}\left(\epsilon^{-1}s\log n\right)$.*

Proof. Consider any two adjacent blocks. By the definition of the x_i's, the ratio between the weights of the oldest and youngest elements is at least $(1+\epsilon)/(1-\epsilon)$. In the full stream, the oldest element has weight $1/n^s$ and the youngest element has weight 1. Let B be the number of blocks so that $\left(\frac{1+\epsilon}{1-\epsilon}\right)^{\lfloor B\rfloor} \leq n^s$. Solving for B, we get $B \leq \frac{s\log n}{\log((1+\epsilon)/(1-\epsilon))}$. We will now lower bound the denominator using the numerical inequality $\ln(1+x) \geq \frac{2x}{2+x}$ for $x > 0$; equivalently $\log(1+x) \geq c \cdot \frac{2x}{2+x}$ for $x > 0$ and $c = \Theta(1)$. We get $\log\left(\frac{1+\epsilon}{1-\epsilon}\right) = \log\left(1 + \frac{2\epsilon}{1-\epsilon}\right) \geq 2c\epsilon$, and therefore we have $B = \mathcal{O}\left(\epsilon^{-1}s\log n\right)$. ◀

We now give the proof of the main theorem for the polynomial decay model.

Proof of Theorem 8. From Proposition 6, we get that when we merge disjoint blocks, we do not sacrifice the coreset approximation parameter ϵ. However, when we reduce, for instance two ϵ-corsets, we get a 2ϵ-coreset. For n points observed in the stream, note that there would be at most $\log n$ reduces. This follows from the fact that the size of successive blocks increase exponentially. Therefore using an offline ϵ'-coreset construction algorithm CS-RAM with $\epsilon' = \epsilon/3\log n$, we get that merging and reducing the blocks produces an $\epsilon/3$-coreset (by Proposition 6). Finally, from Lemma 9, we get that taking a union of these blocks weighted by $u(a, b) = \frac{1}{2}\left(\frac{1-\epsilon}{a^s} + \frac{1+\epsilon}{b^s}\right)$ gives us an ϵ-coreset.

For the space bound, we have from Lemma 10 that the number of blocks is $\mathcal{O}\left(\epsilon^{-1}s\log n\right)$. Since we maintain an $\epsilon/\log n$ coreset for each block, we get that the offline coreset construction algorithm takes space $F(n, \epsilon/\log n)$. Therefore, we get that the space complexity is

$\mathcal{O}\left(\epsilon^{-1} s \log n\, F(n, \epsilon/\log n)\right)$. For update time, note that for n points, we have $\mathcal{O}\left(\epsilon^{-1} s \log n\right)$ blocks and we use an $(\epsilon/\log n)$-coreset algorithm which takes time $\mathcal{O}\left(F(n,\epsilon)\ T(\epsilon/\log n)\right)$ per block. We therefore get a total time of $\mathcal{O}\left(\epsilon^{-1} s\ \log n\ F(n,\epsilon)\, T(\epsilon/\log n)\right)$ ◀

Applications. Coresets have been designed for a wide variety of geometric, numerical linear algebra and learning problems. Some examples include k-median and k-means clustering [17], low rank approximation [46], ℓ_p regression [19], projective clustering [27], subspace approximation [32], kernel methods [48], Bayesian inference [38] etc. We instantiate our framework with a few of these problems, and present the results in Table 1.

6 Exponential Decay

We now discuss another model of time decay in which the weights of previous points decay exponentially with time. Analogous to our polynomial decay model, a point that first appeared in the stream at time $t \leq T$ has weight $2^{\frac{T-t+1}{h}}$ at time T, where the parameter $h > 0$ is the half-life of the decay function. We however consider a different viewpoint to simplify the analysis; we maintain that the weight of a point observed at time t is *fixed* to be $2^{t/h}$ where $h > 0$ is the half-life parameter. These are equivalent since the ratio of weights between successive points is the same in both the models.

Online Facility Location. We first discuss the problem of Online Facility Location (OFL) as our algorithm uses it as a sub-routine. The problem of facility location, given a set of points $P \subseteq \mathcal{X}$, called *demands*, a distance function $d(\cdot,\cdot)$ and fixed cost $f > 0$, conventionally called the facility cost, asks to find a set of points $\mathcal{C}$ that minimizes the following objective.

$$\min_{\mathcal{C}\subseteq\mathcal{X}} \sum_{p\in P} \min_{c\in\mathcal{C}} d(p,c) + |\mathcal{C}|\, f$$

Informally, it seeks a set of points such that the *cumulative* cost of serving the demands (known as *service cost*), which is $d(p,c)$ and opening new facilities f, is minimized. Online Facility Location is the variant of the above problem in the streaming setting, wherein the facility assignments and service costs incurred are irrevocable. That is to say, once a point is assigned to a facility, it cannot be reassigned to a different facility at a later point in time, even if the newer facility is closer. A simple and popular algorithm to this problem is by Meyerson [43], wherein upon receiving a point, it calculates its distance to the nearest facility and flips a coin with bias equal to the distance divided by facility cost. If the outcome is heads (or 1), it opens a new facility, otherwise the nearest point serves this demand and it incurs a service cost, equal to the distance. From here on, we abuse notation and use OFLto refer to the algorithm of Meyerson [43].

6.1 Algorithm

Our algorithm for exponential decaying streams is a variant of the popular k-median clustering algorithm [13, 16], which uses OFL as a sub-routine. We first briefly discuss the algorithm of [13] and then elucidate on how we adapt this to exponential decay streams. The algorithm operates in *phases*, where in each phase it maintains a *guess*, denoted by L, to the lower bound on optimal cost. It then uses this guess to instantiate the OFL algorithm of [43] on a set of points in the stream. If the service cost of OFL grows high or the number of facilities grows large, it infers that the guess is too low and triggers a *phase change*. It then increases the guess by a factor of β (to be set appropriately) and the facilities are put back at the start of the stream and another round of OFL is run.

Notation. We first define and explain some key quantities. The *aspect ratio* of a set is defined as the ratio between the largest distance and the smallest non-zero distance between any two points in the set. We use Δ to denote the aspect ratio of the stream P. For simplicity of presentation, we assume that the minimum non-zero distance between two points is at least 1. We define W as the total weight of the first $h \log \Delta$ points in the stream divided by the minimum weight. Suppose the stream starts at $t = z$, then for any $h = \Omega(1)$,

$$W = \frac{1}{2^{z/h}} \sum_{t=z}^{h \log(\Delta+1)} 2^{t/h} = \frac{\Delta}{2^{1/h} - 1} = \Theta(h\Delta).$$

For a set $P \subseteq (\mathcal{X}, d)$, we use $\mathsf{OPT}_k(P)$ to denote the optimal k-median clustering cost for the set. For two sets P and S, we use $\mathsf{COST}(P, S)$ to denote the cost of clustering P with S as medians. Whenever we use OPT, it corresponds to the optimal cost of k-median clustering of the stream seen till the point in context. We use KM-RAM to denote an offline constant c_r-approximate k-median clustering algorithm in the random access model (RAM). Given a set of points P and a positive integer k, KM-RAM outputs $(\mathcal{C}, \lambda)$, where $\mathcal{C}$ is a set of k points and $\lambda = \mathsf{COST}(P, \mathcal{C}) \leq c_r \cdot \mathsf{OPT}_k(P)$.

Our Algorithm. Our algorithm, inspired from [16, 13], works in phases. We however have important differences. Each of our phases are again sub-divided into two *sub-phases*. In the first sub-phase we execute OFL same as [16, 13] and after each point we check if the cost or the number of facilities is too large. If this is indeed the case, we trigger a phase change. However, if we read $h \log \Delta$ points in a phase, then we move on to the second sub-phase of the algorithm. Here we simply count points and store them verbatim. Upon reading $k + h$ points, we trigger a phase change. The intuition for this sub-phase is that a phase change is triggered when OPT increases by a factor of β. After $h \log \Delta$ points, subsequent points are so heavy relative to points of the previous phase that any service cost will be large enough to ensure OPT has increased. Therefore, we restrict the algorithm to read at most $h \log \Delta + k + h$ points in a single phase. When we start a new phase, we cluster the existing facility set to extract exactly k points using an off-the-shelf constant approximate KM-RAM algorithm and continue processing the stream. We present the above idea in full in Algorithm 2. We now state our main theorem for exponential decay streams.

▶ **Theorem 11.** *There exists a streaming algorithm that given a stream P of exponential decaying points with aspect ratio Δ and half-life h, produces an $\mathcal{O}(1)$-approximate solution to k-median clustering. The algorithm runs in $\mathcal{O}(nk \log(h\Delta))$ time and uses $\mathcal{O}(k \log(h\Delta) + h)$ space.*

6.2 Analysis

We first analyze the service cost and space complexity of OFL. For the t^{th} point in the stream p_t, the weight of p_t, denoted $w(p_t)$, is $w(p_t) = 2^{t/h}$. The following two lemmas will establish bounds on the service cost and number of facilities of OFL.

▶ **Lemma 12.** *When OFL is run on a stream of n points with exponentially decaying weights, with facility cost $f = \frac{L}{k(1+\log W)}$ where $L > 0$, it produces a service cost of at most $6\mathsf{OPT}_k(P) + 2L$ with probability at least $1/2$.*

Proof. The proof follows the standard analysis of Online Facility Location. Let P is the set of points read in a phase. Instead of looking at $|P|$ distinct points with varying weights, we view it as *repeated* points of unit or minimum weight. The total number of points is therefore at most $W = \Theta(h\Delta)$.

Algorithm 2 k-median clustering in exponential decay streams.

Input: k, stream P, an offline constant approximate k-median clustering algorithm KM-RAM.

$L \leftarrow 1$, $\mathcal{C} \leftarrow \emptyset$
while solution not found **do**
  $i \leftarrow 0$, COST $\leftarrow 0$, $f \leftarrow \dfrac{L}{k(1+h\log\Delta)}$.
  while stream not ended **do**
    $p \leftarrow$ next point from stream
    $q \leftarrow$ closest point to p in $\mathcal{C}$
    $\sigma \leftarrow \left(\min\left(\dfrac{w(p)\cdot d(p,q)}{f}, 1\right)\right)$
    if probability σ **then** ▷*do with probability* σ
      $\mathcal{C} \leftarrow \mathcal{C} \cup \{p\}$
    else
      COST $\leftarrow$ COST $+ w(p)\cdot d(p,q)$
      $w(q) \leftarrow w(q) + w(p)$
    end if
    $i \leftarrow i+1$
    if COST $> \gamma L$ or $|\mathcal{C}| > (\gamma-1)k(1+\log W)$ **then** ▷*cost or number of facilities too large*
      break and raise flag ▷*trigger phase change*
    else if $i \geq h\log\Delta$ **then** ▷*second sub-phase*
      for $l = 1$ to $h+k$ **do** ▷*count points and store them verbatim*
        $p \leftarrow$ next point from stream
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{p\}$
      end for
      break and raise flag
    end if
  end while
  if flag raised **then** ▷*phase change*
    $(\mathcal{C}, \lambda) \leftarrow$ KM-RAM $(\mathcal{C}, k)$ ▷*cluster existing facilities*
    $L \leftarrow \max\left(\beta L, \frac{\lambda}{c_r\gamma}\right)$
  else
    Declare solution found
  end if
  $(\mathcal{C}, \lambda) \leftarrow$ KM-RAM $(\mathcal{C}, k)$
end while
Output: $\mathcal{C}$, COST

We remind the reader that $\mathsf{OPT}_k(P) = \min_{K\subseteq P, |K|=k} \sum_{p\in P} \min_{y\in K} d(p,y)$ is the optimal cost and COST(P) is the total service cost incurred by OFL. Let $\mathcal{C}^*$ be the set of corresponding facilities allocated by OPT, and c_i^*'s denote the optimum k facilities where $i \in [k]$ and C_i^* the set of points from P served by the facility c_i^*. Let $A_i = \sum_{x\in C_i^*} d(x, c_i^*)$ be the service cost of C_i^*. We now further partition each region into *rings*. Let S_i^1 be the first ring around c_i^* that contains half the nearest points in C_i^*. Formally, $S_i^1 = \underset{K, |K| = \left|C_i^*\right|/2}{\arg\min} \sum_{x\in K} d(x, c_i^*)$.

Furthermore, S_i^2 is the second ring around c_i^* containing one-quarter of the points in C_i^* and so on. Therefore, we can inductively define $S_i^j = \underset{K,|K|=|C_i^*|/2^j}{\arg\min} \sum_{x\in K\setminus\cup_{l=1}^{j-1}S_i^l} d(x,c_i^*)$. Note that S_i^j may be not be uniquely identifiable, but their existence suffices for the sake of analysis. Let $A_i^j = \sum_{x\in S_i^j} d(x,c_i)$ be the cost of set S_i^j. For a point p, use d_p^* and d_p for its optimal cost and cost incurred in the algorithm respectively.

We look at two cases. In the first case, suppose each region has a facility open; let the facility of S_i^j be s_i^j. We look at the cost incurred by subsequent points arriving in this region. Consider the set S_i^j and let q be a facility in S_i^j. A subsequent point p incurs a cost $d_p = d(p,q)$. By triangle inequality, we have $d_p \le d_p^* + d_q^*$. By definition of S_i^j, we have $d_q^* \le d_z^*$ for any point $z \in S_i^{j+1}$. We sum over all z in S_i^{j+1} and get $d_q^* \le \frac{A_i^{j+1}}{|S_i^{j+1}|}$. We therefore get $d_p \le d_p^* + \frac{A_i^{j+1}}{|S_i^{j+1}|}$. Summing over all points is S_i^j, we get $\mathsf{COST}(S_i^j, s_i^j) \le A_i^j + \frac{|S_i^j| A_i^{j+1}}{|S_i^{j+1}|} = A_i^j + 2\cdot A_i^{j+1}$. Summing over all j's, we get $\mathsf{COST}(C_i^*, c_i^*) \le 3A_i$. Finally, summing over i's, we get that in the first case $\mathsf{COST}(P,\mathcal{C}^*) \le 3\mathsf{OPT}_k(P)$. We now look at the second case wherein each region has a facility open. The number of points is at most W, therefore, the number of regions is at most $k(1+\log(W))$. The expected service cost incurred by a region before opening a facility is at most f (See Fact 1, [41]). Therefore, the total service cost $\le f\,k(1+\log(W)) = L$. Combining the two cases, we get that $\mathsf{COST}(P,\mathcal{C}^*) \le 3\mathsf{OPT}_k(P) + L$. Note that when we store points verbatim, we do not incur any service cost. With a simple application of Markov inequality, we get that with probability at least 1/2, $\mathsf{COST}(P,\mathcal{C}^*) \le 6\mathsf{OPT}_k(P) + 2L$. ◀

▶ **Lemma 13.** *When OFL is run on a stream of n points with exponentially decaying weights, with facility cost $f = \frac{L}{k(1+\log W)}$ where $L > 0$, the number of facilities produced is at most $(2 + \frac{6}{L}\mathit{OPT}_k(P))k(1+\log W)$, with probability at least 1/2.*

Proof. Considering the points as repeated points of minimum weight, the total number of points is at most W and the total number of regions is at most $k(1+\log W)$. One facility in each region gives us $k(1+\log W)$ facilities. After opening a facility in a region, each subsequent point has probability $\frac{d_p}{f}$ to open a facility. Therefore, the expected number of facilities is $\sum_p \frac{d_p}{f}$. We showed in Lemma 12 that $\sum_p d_p \le 3\ \mathsf{OPT}_k(P)$. Hence, the expected number of facilities is at most $\frac{3\mathsf{OPT}_k(P)}{f} = \frac{3\mathsf{OPT}_k(P)k(1+\log W)}{L}$. A simple application of Markov's inequality completes the proof. ◀

***k*-median clustering.** We now state some key lemmas that will help us establish that the algorithm produces a $\mathcal{O}(1)$ approximation to the k-median clustering cost. We then show how these come together and present the detailed guarantees in Theorem 17.

▶ **Lemma 14.** *At every phase change, with probability at least 1/2, $\mathit{OPT}_k(P) > L$ if $\beta \le 2$ and $\gamma \ge 9$.*

Proof. The phase change is triggered in two ways, either the cost or the number of facilities grows large (more precisely, cost more that γL or the number of facilities greater than $(\gamma-1)k(1+\log W)$), or we read too many points. Let us look at the first case. Assume that $L \ge \mathsf{OPT}_k(P)$, then from Lemma 12 and 13, we get that with probability at least 1/2, $\mathsf{COST} \le 8L$ and the number of facilities is $\le 8k(1+\log W)$ respectively. However with $\gamma \ge 9$, neither of the two conditions are met and therefore the premise that a phase change was triggered gives us a contradiction. Hence, in the first case, we get $L < \mathsf{OPT}_k(P)$ with probability at least 1/2.

In the other case, we store points exactly (incurring no additional cost). The only danger in this case is performing a phase change too early (before OPT has doubled). Let $\underline{\mathsf{OPT}}$ be the value of OPT at the beginning of the phase, which we assume starts at time $t = z$. Since points cannot be at distance greater than Δ, then

$$\begin{aligned}\underline{\mathsf{OPT}} &\leq \Delta(1 + 2^{1/h} + \ldots + 2^{z/h}) \\ &\leq \Delta \frac{2^{(z+1)/h} - 1}{2^{1/h} - 1}\end{aligned}$$

Now let $\overline{\mathsf{OPT}}$ be the value of OPT after terminating the phase (which occurs after reading $k + h$ distinct points after the initial $h \log \Delta$ points of the phase). We must prove that $\overline{\mathsf{OPT}} \geq 2\underline{\mathsf{OPT}}$. Observe that after reading $k + h$ distinct points, we must cluster at least h points across a distance of at least 1 (since we can have at most k centers). The weights of these points begin at $2^{(z+h\log\Delta+1)/h}$. Therefore,

$$\begin{aligned}\overline{\mathsf{OPT}} &\geq \underline{\mathsf{OPT}} + \sum_{i=z+h\log\Delta}^{z+h\log\Delta+h} 2^{i/h} \\ &= \underline{\mathsf{OPT}} + \frac{2^{(z+h\log_2(\Delta)+h)/h} - 2^{(z+h\log_2(\Delta))/h}}{2^{1/h} - 1} \\ &\geq \underline{\mathsf{OPT}} + \Delta\left(\frac{2^{(z+1)/h} - 1}{2^{1/h} - 1}\right) \\ &\geq 2\underline{\mathsf{OPT}},\end{aligned}$$

where the second inequality follows from straightforward arithmetic. Let L' be the value of L in the previous phase. Thus,

$$\overline{\mathsf{OPT}} \geq 2\underline{\mathsf{OPT}} > 2L' = \frac{2}{\beta}L$$

where the second inequality holds with probability at least $1/2$, as justified above. Setting $\beta \leq 2$ completes the proof. ◀

▶ **Lemma 15.** *At any part in the algorithm, we have* $\mathsf{COST}(P, \mathcal{C}) \leq \left(\gamma + \frac{1+c_r\beta}{\beta-1}\right) L$.

Proof. We know that the increase of $\mathsf{COST}(P, \mathcal{C})$ in the current phase is upper bounded by the variable COST (see Algorithm 2). In a single phase, we have $\mathsf{COST} \leq \gamma L$. Therefore, outside the phase loop, we just need to show that it is at most $\frac{1+c_r\beta}{\beta-1}L$. Note that it changes only by the KM-RAM algorithm, which incurs cost of $\lambda \leq c_r\gamma L$. Suppose that it holds in the previous phase and let L' be the value of L in the previous phase. Then the cost outside the loop is $\gamma L' + \frac{1+c_r\beta}{\beta-1}L' + \lambda \leq \frac{1+c_r\beta}{\beta-1}L$, which finishes the proof. ◀

▶ **Lemma 16.** *With probability at least* $1/2$, $L \leq \left(1 + \frac{1}{\gamma} + \frac{1+c_r\beta}{\gamma(\beta-1)}\right) \mathsf{OPT}_k(P)$.

Let L' and $\mathcal{C}'$ denote the values of L and $\mathcal{C}$ in the previous phase. We condition on the event that $L' < \mathsf{OPT}_k(P)$, which we know from Lemma 14 occurs with probability at least $1/2$. From the update equation of L, we either have $L = \beta L'$ or $L = \frac{\lambda}{c_r\gamma}$. In the first case, we directly get $L \leq \beta\mathsf{OPT}_k(P)$. With $\beta \leq 2$, we get the claim of the lemma. We now look at the second case, where we have $\gamma c_r L \leq \lambda \leq c_r\mathsf{OPT}_k(\mathcal{C}')$ from the guarantee of the KM-RAM algorithm. It is easy to see that $\mathsf{OPT}_k(\mathcal{C}') \leq \mathsf{OPT}_k(P) + \mathsf{COST}(P, \mathcal{C}')$ by a simple application of triangle inequality on all the points. Moreover, from Lemma 15, we have $\mathsf{COST}(P, \mathcal{C}') \leq \left(\gamma + \frac{1+c_r\beta}{\beta-1}\right) L' \leq \left(\gamma + \frac{1+c_r\beta}{\beta-1}\right) \mathsf{OPT}_k(P)$. Combining these, we get $L \leq \left(\frac{1}{\gamma} + 1 + \frac{1+c_r\beta}{\gamma(\beta-1)}\right) \mathsf{OPT}_k(P)$.

We now restate the theorem for the exponential decay model but tailored to Algorithm 2 with all the algorithmic details precisely stated.

▶ **Theorem 17.** *Let P be a stream of n points with exponential decaying weights parametrized by the half-life parameter h and let k be some positive integer. Algorithm 2 run with $\beta \leq 2, \gamma \geq 9, W = \mathcal{O}(h\Delta)$ on the stream P outputs k points, which produce an $\mathcal{O}(1)$ approximation to the optimal cost of k-median clustering on P with high probability. The algorithm runs in time $\mathcal{O}(nk \log W)$ and uses space $\mathcal{O}(k \log W + h)$.*

Proof. Combining Lemma 15 and 16, we get that

$$\mathsf{COST}(P, \mathcal{C}) \leq \left(\gamma + \frac{1 + c_r\beta}{\beta - 1}\right)\left(\frac{1}{\gamma} + 1 + \frac{1 + c_r\beta}{\gamma(\beta - 1)}\right)\mathsf{OPT}_k(P).$$

Setting $\beta = 2, \gamma = 10$ and $c_r = 3$ gives us that $\mathsf{COST}(P, \mathcal{C}) \leq 40\mathsf{OPT}_k(P)$.

We emphasize that we give a *streaming* guarantee, that is, given a fixed point in the stream, it will hold for all the points seen till then. Note that in the proofs of Lemma 14 and 16, we only need that the random event hold with probability at least $1/2$ *only* in the previous phase. We can therefore amplify the probability of success by running $\log(1/\delta)$ parallel instances to get the bounds to hold with probability at least $1 - \delta$. The space bound of the algorithm is $\mathcal{O}(k \log W + h) = \mathcal{O}(k \log(h\Delta) + h)$, which simply follows from the condition in the algorithm that we don't allow the number of facilities to grow beyond $\mathcal{O}(k(1 + \log(W))$ combined with the fact that we store $k + h$ points verbatim in the second sub-phase. ◀

Extensions. As in [41], our algorithm can easily be extended to other distance functions that satisfy the approximate triangle inequality (see Definition 7). In particular, we get constant approximate algorithms for k-means clustering and M-estimators in the exponential decay model.

References

1 Charu C. Aggarwal, editor. *Data Streams - Models and Algorithms*, volume 31 of *Advances in Database Systems*. Springer, 2007.

2 Arvind Arasu and Gurmeet Singh Manku. Approximate Counts and Quantiles over Sliding Windows. In *Proceedings of the Twenty-third ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 286–296, 2004.

3 Sepehr Assadi and Sanjeev Khanna. Randomized Composable Coresets for Matching and Vertex Cover. In *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures, (SPAA)*, pages 3–12, 2017.

4 Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and Issues in Data Stream Systems. In *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, pages 1–16, 2002.

5 Brian Babcock, Mayur Datar, Rajeev Motwani, and Liadan O'Callaghan. Maintaining variance and k-medians over data stream windows. In *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, pages 234–243, 2003.

6 Rafael da Ponte Barbosa, Alina Ene, Huy L. Nguyen, and Justin Ward. A New Framework for Distributed Submodular Maximization. In *IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 645–654, 2016.

7 Jon Louis Bentley and James B Saxe. Decomposable searching problems I. Static-to-dynamic transformation. *Journal of Algorithms*, 1(4):301–358, 1980.

8 Vladimir Braverman, Petros Drineas, Jalaj Upadhyay, David P. Woodruff, and Samson Zhou. Numerical Linear Algebra in the Sliding Window Model. *arXiv preprint*, 2018. `arXiv:1805.03765`.

9 Vladimir Braverman, Dan Feldman, and Harry Lang. New frameworks for offline and streaming coreset constructions. *arXiv preprint*, 2016. `arXiv:1612.00889`.

10 Vladimir Braverman, Elena Grigorescu, Harry Lang, David P. Woodruff, and Samson Zhou. Nearly Optimal Distinct Elements and Heavy Hitters on Sliding Windows. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 7:1–7:22, 2018.

11 Vladimir Braverman, Harry Lang, Keith Levin, and Morteza Monemizadeh. Clustering on Sliding Windows in Polylogarithmic Space. In *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS*, pages 350–364, 2015.

12 Vladimir Braverman, Harry Lang, Keith Levin, and Morteza Monemizadeh. Clustering Problems on Sliding Windows. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1374–1390, 2016.

13 Vladimir Braverman, Adam Meyerson, Rafail Ostrovsky, Alan Roytman, Michael Shindler, and Brian Tagiku. Streaming k-means on well-clusterable data. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 26–40. Society for Industrial and Applied Mathematics, 2011.

14 Vladimir Braverman and Rafail Ostrovsky. Smooth Histograms for Sliding Windows. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 283–293, 2007.

15 Timothy M. Chan and Bashir S. Sadjad. Geometric Optimization Problems over Sliding Windows. *Int. J. Comput. Geometry Appl.*, 16(2-3):145–158, 2006. A preliminary version appeared in the Proceedings of Algorithms and Computation, 15th International Symposium (ISAAC), 2004.

16 Moses Charikar, Liadan O'Callaghan, and Rina Panigrahy. Better streaming algorithms for clustering problems. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 30–39. ACM, 2003.

17 Ke Chen. On coresets for k-median and k-means clustering in metric and euclidean spaces and their applications. *SIAM Journal on Computing*, 39(3):923–947, 2009.

18 Yun Chi, Haixun Wang, Philip S. Yu, and Richard R. Muntz. Catch the moment: maintaining closed frequent itemsets over a data stream sliding window. *Knowl. Inf. Syst.*, 10(3):265–294, 2006. A preliminary version appeared in the Proceedings of the 4th IEEE International Conference on Data Mining (ICDM), 2004.

19 Kenneth L Clarkson and David P Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 205–214. ACM, 2009.

20 Edith Cohen and Martin Strauss. Maintaining time-decaying stream aggregates. In *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 223–233, 2003.

21 Graham Cormode, Flip Korn, and Srikanta Tirthapura. Time-decaying aggregates in out-of-order streams. In *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pages 89–98, 2008.

22 Graham Cormode, Srikanta Tirthapura, and Bojian Xu. Time-decaying sketches for sensor data aggregation. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing, PODC*, pages 215–224, 2007.

23 Graham Cormode, Srikanta Tirthapura, and Bojian Xu. Time-Decayed Correlated Aggregates over Data Streams. In *Proceedings of the SIAM International Conference on Data Mining, SDM*, pages 271–282, 2009.

24 Anirban Dasgupta, Petros Drineas, Boulos Harb, Ravi Kumar, and Michael W. Mahoney. Sampling Algorithms and Coresets for ℓ_p Regression. *SIAM J. Comput.*, 38(5):2060–2078, 2009. A preliminary version appeared in the Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA) 2008.

25 Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining Stream Statistics over Sliding Windows. *SIAM J. Comput.*, 31(6):1794–1813, 2002. A preliminary version appeared in the Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2002.

26 Mayur Datar and S. Muthukrishnan. Estimating Rarity and Similarity over Data Stream Windows. In *Algorithms - ESA 2002, 10th Annual European Symposium, Proceedings*, pages 323–334, 2002.

27 Amit Deshpande, Luis Rademacher, Santosh Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1117–1126. Society for Industrial and Applied Mathematics, 2006.

28 Joan Feigenbaum, Sampath Kannan, and Jian Zhang. Computing Diameter in the Streaming and Sliding-Window Models. *Algorithmica*, 41(1):25–41, 2005.

29 Dan Feldman, Amos Fiat, and Micha Sharir. Coresets forWeighted Facilities and Their Applications. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 315–324, 2006.

30 Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC*, pages 569–578, 2011.

31 Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the 23rd ACM Symposium on Computational Geometry (SoCG)*, pages 11–18, 2007.

32 Dan Feldman, Morteza Monemizadeh, Christian Sohler, and David P Woodruff. Coresets and sketches for high dimensional subspace approximation problems. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 630–649. Society for Industrial and Applied Mathematics, 2010.

33 Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, PCA and projective clustering. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA), 2013*, pages 1434–1453, 2013.

34 Mina Ghashami, Edo Liberty, Jeff M Phillips, and David P Woodruff. Frequent directions: Simple and deterministic matrix sketching. *SIAM Journal on Computing*, 45(5):1762–1792, 2016.

35 Phillip B. Gibbons and Srikanta Tirthapura. Distributed streams algorithms for sliding windows. In *SPAA*, pages 63–72, 2002.

36 Sudipto Guha, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams. In *Foundations of computer science, 2000. proceedings. 41st annual symposium on*, pages 359–366. IEEE, 2000.

37 Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 291–300, 2004.

38 Jonathan Huggins, Trevor Campbell, and Tamara Broderick. Coresets for scalable bayesian logistic regression. In *Advances in Neural Information Processing Systems*, pages 4080–4088, 2016.

39 Piotr Indyk, Sepideh Mahabadi, Mohammad Mahdian, and Vahab S. Mirrokni. Composable core-sets for diversity and coverage maximization. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 100–108, 2014.

40 Tsvi Kopelowitz and Ely Porat. Improved Algorithms for Polynomial-Time Decay and Time-Decay with Additive Error. In *Theoretical Computer Science, 9th Italian Conference, ICTCS Proceedings*, pages 309–322, 2005.

41 Harry Lang. Online Facility Location on Semi-Random Streams. *arXiv preprint*, 2017. `arXiv:1711.09384`.

42 Lap-Kei Lee and H. F. Ting. A simpler and more efficient deterministic scheme for finding frequent items over sliding windows. In *Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 290–297, 2006.

43 Adam Meyerson. Online facility location. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 426–431. IEEE, 2001.

44 Vahab S. Mirrokni and Morteza Zadimoghaddam. Randomized Composable Core-sets for Distributed Submodular Maximization. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing (STOC)*, pages 153–162, 2015.

45 S. Muthukrishnan. Data Streams: Algorithms and Applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.

46 Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 143–152. IEEE, 2006.

47 Shang-Yu Su, Pei-Chieh Yuan, and Yun-Nung Chen. How time matters: Learning time-decay attention for contextual spoken language understanding in dialogues. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2133–2142, 2018.

48 Yan Zheng and Jeff M Phillips. Coresets for Kernel Regression. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 645–654. ACM, 2017.

Approximation Algorithms for Partially Colorable Graphs

Suprovat Ghoshal
Indian Institute of Science, Bangalore, India
suprovat@iisc.ac.in

Anand Louis
Indian Institute of Science, Bangalore, India
anandl@iisc.ac.in

Rahul Raychaudhury
Indian Institute of Science, Bangalore, India
rahulr@iisc.ac.in

Abstract

Graph coloring problems are a central topic of study in the theory of algorithms. We study the problem of partially coloring *partially colorable graphs*. For $\alpha \leq 1$ and $k \in \mathbb{Z}^+$, we say that a graph $G = (V, E)$ is α-partially k-colorable, if there exists a subset $S \subset V$ of cardinality $|S| \geq \alpha|V|$ such that the graph induced on S is k-colorable. Partial k-colorability is a more robust structural property of a graph than k-colorability. For graphs that arise in practice, partial k-colorability might be a better notion to use than k-colorability, since data arising in practice often contains various forms of noise.

We give a polynomial time algorithm that takes as input a $(1-\epsilon)$-partially 3-colorable graph G and a constant $\gamma \in [\epsilon, 1/10]$, and colors a $(1-\epsilon/\gamma)$ fraction of the vertices using $\tilde{O}\left(n^{0.25+O(\gamma^{1/2})}\right)$ colors. We also study natural semi-random families of instances of partially 3-colorable graphs and partially 2-colorable graphs, and give stronger bi-criteria approximation guarantees for these family of instances.

2012 ACM Subject Classification Mathematics of computing → Approximation algorithms

Keywords and phrases Approximation Algorithms, Vertex Coloring, Semi-random Models

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.28

Category APPROX

Related Version A full version of the paper is available at `https://arxiv.org/abs/1908.11631`.

Funding *Anand Louis*: Supported in part by SERB Award ECR/2017/003296.

Acknowledgements The first author thanks Pasin Manurangsi for pointing him to the Odd Cycle Transversal problem.

1 Introduction

Graph coloring problems are a central topic of study in the theory of algorithms [33, 17, 4, 20]. An undirected graph $G = (V, E)$ is said to be k-colorable if there exists an assignment of colors $f : V \to [k]$ such that $f(u) \neq f(v)$ for each $\{u, v\} \in E$. For a graph G, the minimum value of k for which it is k-colorable is called its chromatic number. Computing a 3-coloring of a 3-colorable graph is a fundamental NP-hard problem. Efficiently computing a coloring of a 3-colorable graph which only uses a few colors is a major open problem in the study of algorithms. The current best known algorithm colors a 3-colorable graph on n vertices using $O(n^{0.199})$ colors [20]. We study the problem of coloring partially colorable graphs.


© Suprovat Ghoshal, Anand Louis, and Rahul Raychaudhury;
licensed under Creative Commons License CC-BY

▶ **Definition 1.** *An undirected graph $G = (V, E)$ is defined to be α-partially k-colorable, denoted by α-PkC , if there exists a subset $V_{\text{good}} \subset V$ such that $|V_{\text{good}}| \geq \alpha |V|$ and the graph induced on V_{good} is k-colorable. We will call such a set V_{good} the set of* good *vertices, and $V_{\text{bad}} \stackrel{\text{def}}{=} V \setminus V_{\text{good}}$ the set of* bad *vertices.*

We remark that for a given graph the partitioning of the vertex set V into V_{good} and V_{bad} may not be unique. In such cases, the claims we make in this paper will hold for any such fixed partition.

It is well known that for a fixed k, the problem of determining whether a given graph is k-colorable is an NP-hard problem [18]. Therefore, determining whether a graph belongs to 1-PkC is an NP-hard problem, and hence, computing the largest value of α for which a graph belongs to α-PkC is also an NP-hard problem.

Note that a graph that is $(1-\epsilon)$-partially 3-colorable can have chromatic number as large as $|V_{\text{bad}}| = \epsilon n$. Therefore, the notion of the chromatic number of the graph does not capture the structural property (3-colorability) satisfied by most of the graph. Partial k-colorability is a more robust stuctural property than k-colorability. Therefore, for graphs that arise in practice, partial k-colorability might be a better notion to use than k-colorability, since data arising in practice often contains various forms of noise; the notion of bad vertices can be used to capture some types of noisy vertices in the graph.

Other notions of partial k-coloring

Another related notion of partial coloring is the following.

▶ **Definition 2.** *An undirected graph $G = (V, E)$ is defined to be α-partially k-colorable, if there exists a coloring of the vertices $f : V \to [k]$ such that for at least $\alpha |E|$ edges $\{u, v\}$, $f(u) \neq f(v)$.*

This definition, which asks that the coloring should "satisfy" at least α fraction of the edges, can be viewed as the *edge* version of partial k-colorability, whereas Definition 1 can be viewed as the *vertex* version of partial k-colorability. For a fixed constant k, computing the maximum value of α for which the input graph satisfies Definition 2 can be formulated as a Max-2-CSP with alphabet size k; approximation algorithms for Max-2-CSPs have been extensively studied in the literature [29, 30, 7] etc. Therefore, we focus our attention on Definition 1.

1.1 Our Results

We give an efficient (bi-criteria) approximation algorithm for coloring partially 3-colorable graphs.

▶ **Theorem 3.** *There exists a polynomial time algorithm that takes as input a $(1-\epsilon)$-P3C graph $G = (V, E)$ and any fixed choice of $\gamma \in [\epsilon, 1/100]$, and produces a set $S \subset V$ such that $|S| \leq (3\epsilon/\gamma)|V|$ and a coloring of $V \setminus S$ using $\tilde{O}(n^{0.25+O(\gamma^{1/2})})$ colors*[1].

We point out that the above theorem gives a bi-criteria approximation guarantee which exhibits the tradeoff between the size of the set S, and the number of colors used to color the remaining graph $G[V \setminus S]$. In particular, setting $\gamma = \sqrt{\epsilon}$ in the above theorem gives

[1] $\tilde{O}(\cdot)$ hides factors polylogarithmic in n.

us the following guarantee. Given a $(1-\epsilon)$-P3C graph, one can color $(1-\sqrt{\epsilon})$-fraction of its vertices using $\tilde{O}(n^{0.25+\epsilon^{1/4}})$-colors. Using similar techniques we can give an efficient approximation algorithm for the partial 2-coloring setting as well. For completeness, we formally state the result below[2] :

▶ **Proposition 4.** *There exists a polynomial time algorithm that takes as input a $(1-\epsilon)$-P2C graph $G=(V,E)$ and any fixed choice of $\gamma \in [\epsilon, 1/100]$, and produces a set $S \subset V$ such that $|S| \leq (\epsilon/\gamma)|V|$ and a coloring of $V \setminus S$ using $\tilde{O}(n^{C\gamma})$ colors, for some constant $C > 0$.*

The proof of the above proposition can be found in the full version and uses exactly the same techniques as Theorem 3. We also study a semi-random family of partially colorable graphs α-PkC$^{\mathcal{R}}(n,p)$, which we define as follows.

▶ **Definition 5.** *An instance of α-PkC$^{\mathcal{R}}(n,p)$ is generated as follows.*

1. *Let V be a set of n vertices. Arbitrarily partition V into sets V_{good} and V_{bad} such that $|V_{\text{good}}| \geq \alpha n$.*
2. *Add edges between an arbitrary number of arbitrarily chosen pairs of vertices in V_{good} such that the graph induced on V_{good} is k-colorable.*
3. *Add edges between an arbitrary number of arbitrarily chosen pairs of vertices in V_{bad}.*
4. *Between each pair of vertices in $V_{\text{good}} \times V_{\text{bad}}$, independently add an edge with probability p. We call this set of edges E_0.*
5. *Add arbitrary number of edges between pairs of vertices of $V_{\text{good}} \times V_{\text{bad}}$. We call this set of edges E_1.*

Output the resulting graph.

In the study of approximation algorithms for NP-hard problems, there have been many works studying algorithms random and semi-random instances of various problems [11, 15, 21, 24, 25]. Random and semi-random instances are often good models for instances arising in practice; designing algorithms specifically for such instances, whose performance guarantee is significantly better than guarantees for general instances, could have more applications in practice. Moreover, from a theoretical perspective, designing algorithms for semi-random instances helps us to better understand what aspects of a problem make it intractable. We study our semi-random model α-PkC$^{\mathcal{R}}(n,p)$ for the same reasons. The following is our main result.

▶ **Theorem 6.** *Suppose there exists an efficient algorithm which colors a 3-colorable graph using n^{θ} colors. Then the following holds for all choices of $\epsilon = \Omega(\log n/n)$ and $p \geq (\epsilon\theta^{-2})^{O(\theta)}$. There exists a polynomial time algorithm that takes as input a graph G sampled from $(1-\epsilon)$-P3C$^{\mathcal{R}}(n,p)$ and produces a set S such that $|S| = O\left(\epsilon\theta^{-2}np^{-(O(1/\theta))}\right)$ and a coloring of $V \setminus S$ using at most n^{θ} colors with high probability. Moreover, the algorithm runs in time $n^{O(1/\theta)}\text{poly}(n)$.*

In particular, instantiating the above theorem with the algorithm from [20], w.h.p., we can color $(1-O(\epsilon))n$ fraction of vertices with $\tilde{O}(n^{0.199})$-colors. We also study the partial 2-coloring problem in the semi-random setting. Our guarantees for this setting are as follows:

▶ **Theorem 7.** *Let $\epsilon = \Omega(\log n/n)$ and $p > \sqrt{\epsilon}$. Then, there exists a polynomial time algorithm that takes as input a graph G sampled from $(1-\epsilon)$-P2C$^{\mathcal{R}}(n,p)$, and with high probability, produces a set $S \subseteq V$ such that $|S| = O\left(\epsilon np^{-2}\right)$ and the induced subgraph on the remaining vertices $G[V \setminus S]$ is 2-colorable.*

[2] We implicitly use the algorithm in the degree reduction step of the algorithm from Theorem 3. See Claim 18 for details.

In particular, in the above theorem the number of vertices removed is bounded by $O(\epsilon n)$ which is stronger than the best known bound of $O(\sqrt{\log n}.\epsilon n)$ [1] in the adversarial setting.

1.2 Related Work

3-colorable graphs. There is extensive literature on algorithms for coloring 3-colorable graphs. Wigderson [33] gave a simple combinatorial algorithm that used $O(n^{\frac{1}{2}})$ colors. Blum [9] improved the number of colors used to $\tilde{O}(n^{\frac{3}{8}})$. These algorithms used purely combinatorial techniques. Karger, Motwani and Sudan [17] used semidefinite programming to develop an algorithm, which when balanced with Wigderson's technique [33] used $\tilde{O}(n^{\frac{1}{4}})$ colors. Blum and Karger [10] improved the number of colors used to $\tilde{O}(n^{\frac{3}{14}})$ by combining the techniques used in [9] and [17]. Arora, Chlamtac and Charikar [3] got the bound down to $\tilde{O}(\Delta^{0.21111})$ using techniques from the ARV algorithm [5], which was further improved by Chlamtac [12] to $\tilde{O}(n^{0.2072})$ using SDP hierarchies. Using new combinatorial techniques, Kawarabayashi and Thorup improved the approximation bound to $\tilde{O}(n^{0.2049})$ in [19]. Subsequently, by combining their techniques with [12], they were able to give a approximation of $\tilde{O}(n^{0.19996})$ [20], which is the current state of the art.

Partially 2-colorable graphs. The partial 2-coloring problem, better known as Odd Cycle Transversal (OCT) in the literature, has also been studied extensively. Formally, the setting here is as follows. We are given a $(1-\epsilon)$-partially 2-colorable graph $G = (V, E)$ and the objective is to find a set S of minimum size such that $G[V \setminus S]$ is 2-colorable (i.e., odd cycle free). Yannakakis first showed that it is NP-Complete in [34]. Later, Khot and Bansal [6] showed that OCT is hard to approximate to any constant factor, assuming the Unique Games Conjecture. From the algorithmic side, via a reduction through the Min2CNF Deletion problem, [16] gave a $O(\log n)$ approximation for the problem. This was later improved to $O(\sqrt{\log n})$ by [1] by using techniques from the Arora-Rao-Vazirani [5] algorithm for sparsest cut. This problem has also been studied under the lens of parameterized complexity. In [31], Reed et al. showed that OCT is fixed parameter tractable when parameterized by the number of bad vertices, following which a sequence of works [6, 28, 23] gave algorithms with improved running times.

Partially 3-colorable graphs. In contrast to the 3-colorable setting, there has been very little work on coloring partially 3-colorable graph. The paper which is closest to our setting is by Kumar, Louis and Tulsiani [22], which also addresses the partial 3-coloring problem, albeit in a more restrictive setting. Assuming that the $(1-\epsilon)$-partially 3-colorable graph has threshold rank r and the 3-coloring on the good vertices satisfies certain psuedorandomness properties, they give an algorithm which 3-colors $1 - O(\gamma + \epsilon)$ fraction of vertices in time $(r.n)^{O(r)}$.

Graph problems in Semi-random Models. The semi-random model used in this paper is similar to semi-random models which have been considered for the Max-Independent Set problem [11] [15] [32] [26]. Semi-random models offer a natural way of understanding the complexity of problems in settings which are less restrictive than worst case complexity, but are still far from being average case. While semi-random models were first introduced for studying graph coloring in [11], it has also subsequently been used to study several other fundamental problems such as Unique Games [21], Graph Partitioning [24], Clustering [25], to name a few.The problem of coloring 3-colorable graphs has also been studied in average-case and planted models. Alon and Kahale [2] gave an efficient algorithm that finds an exact

3-Coloring of a random 3-Colorable graph with high probability. David and Fiege [13] studied the complexity of finding a planted random/adversarial 3-coloring for both adversarial and random host graphs.

1.3 Discussion and Proof Overview

Adversarial Model. The key component in most approximation algorithms for 3-coloring involves solving a SDP relaxation of the 3-coloring problem, and followed by a randomized rounding procedure for coloring the graph. The standard SDP relaxation for 3-coloring is the following which was introduced in [17]:

▶ **SDP 8** (Exact 3-Coloring SDP).

$$\begin{aligned} &\textit{minimize} && 0 \\ &\textit{subject to} && v_i \cdot v_j \leq -\tfrac{1}{2} \qquad \forall \{i,j\} \in E \\ &&& \|v_i\|^2 = 1 \qquad \forall i \in V \end{aligned}$$

SDP 8 doesn't optimize any objective function, it finds a feasible solution which satisfies all the constraints of SDP. The intended solution to the above SDP is as follows. Let $\sigma : V \to \{1,2,3\}$ be any legal coloring of G. Furthermore, let $u_1, u_2, u_3 \in \mathbb{R}^2$ be any three unit vectors satisfying $\langle u_i, u_j \rangle = -1/2$ for every $i, j \in \{1,2,3\}, i \neq j$. We identify the vector u_i with the color i, and assign $v_j = u_{\sigma(j)}$ for every $j \in V$. It can be easily verified that this is a feasible solution to the above SDP. As is usual, while the SDP in general may not return the above vector coloring, one can round a feasible vector coloring to color the graph using not too many colors [17]. The approximation guarantee is usually of the form Δ^c (for some $c \in (0,1)$), where Δ is the maximum degree of the graph.

Since in general, one cannot hope to have a degree bound on the graph, the above step is usually preceded by a *degree reduction* sub-routine. Note that if a graph is 3-colorable (more generally k-colorable), then the graph induced on the neighbours of any vertex v is 2-colorable (more generally $k-1$ colorable). Since a 2-colorable graph can be colored with 2 colors efficiently, the graph induced on any vertex and its neighbours can be colored efficiently with 3 colors. Therefore, fixing a threshold Δ, this procedure iteratively removes vertices (and their neighbours) having degree larger than Δ from the graph while coloring them with few colors, and terminates when maximum degree of the remaining graph is at most Δ. In particular, if the degree reduction step uses $f(n,\Delta)$ colors, then the total number of colors used by the algorithm is at most $f(n,\Delta) + \Delta^c$. Then one can optimize the choice of Δ for giving the best possible approximation guarantee. This degree reduction approach and its variants, first studied by Wigderson [33], has been subsequently used in almost all known approximation algorithms for graph coloring

In translating the above template to the setting of partially 3-colorable graphs, we face several immediate challenges. SDP 8 is guaranteed to return a feasible solution only for 3-colorable graphs, it might be infeasible if the graph is not 3-colorable. If we could compute the set of good vertices then we could use SDP 8 only on the set of good vertices. However, in general, the problem of identifying the set of good vertices is NP-hard (Fact 27). Finally, the preprocessing steps for degree reduction rely heavily on the combinatorial structural properties of the neighborhood of vertices in *exactly* 3-colorable graphs, which, in general, may not be satisfied by *partially* 3-colorable graphs.

Our approach is to begin with an SDP relaxation that tries to solve both problems together: identifying the set of bad vertices, and coloring the set of good vertices. We introduce variables $w_1, w_2, \ldots, w_n$ where the i^{th} variable w_i is meant to indicate if vertex

i is bad. Additionally, for every edge $(i,j) \in E$, we introduce slack variables z_{ij} which are meant to indicate if at least one of the vertices i, j is bad. Using the slack variables we relax the edge constraints as $\langle v_i, v_j \rangle \leq -1/2 + (3/2)z_{ij}$. Finally, we connect the edge indicator variables with vertex indicator variables using constraints of the form $z_{ij} \leq w_i + w_j$. Since we want the set of bad vertices to be small, our objective function will be to minimize $\sum_{i \in V} w_i$. Our SDP relaxation is the following.

▶ **SDP 9** (Partial 3-Coloring SDP)**.**

$$\begin{array}{lll} \textit{minimize} & \sum_{i \in V} w_i & \\ \textit{subject to} & \langle v_i, v_j \rangle \leq -\frac{1}{2} + \frac{3}{2} z_{ij} & \forall \{i,j\} \in E \\ & z_{ij} \leq w_i + w_j & \forall \{i,j\} \in E \\ & 0 \leq z_{ij} \leq 1 & \forall \{i,j\} \in E \\ & 0 \leq w_i \leq 1 & \forall i \in V \\ & \|v_i\|^2 = 1 & \forall i \in V \end{array}$$

Since the optimal "integer solution" forms a feasible solution to the SDP relaxation, it is easy to show that for a $(1-\epsilon)$-partially 3-colorable graph, the optimal of the above SDP is at most ϵn. Therefore by Markov's inequality, we get that for a large fraction of $i \in [n]$, the w_i variables are small. Let $V' \subset V$ be the set of vertices with small w_i. Since $|V \setminus V'| = O(\epsilon n)$, we can focus on coloring the induced subgraph $G' = G[V']$. G' has the following nice property: *for every edge (i,j) in G', the corresponding edge constraint is approximately satisfied i.e., $\langle v_i, v_j \rangle \leq -1/2 + o_\epsilon(1)$, where the second term goes to 0 as ϵ goes to 0*. We call such graphs as being approximately vector 3-colorable (See Definition 11 for a formal description). We use this property crucially in designing our preprocessing step.

We observe that the neighborhood of any vertex in an approximately vector 3-colorable graph is approximately vector 2-colorable. Furthermore, we show that approximately vector 2-colorable graphs are *short odd cycle* free. Graphs having this property are known to have large independent sets which can be found efficiently [27]. Thus one can find such large independent sets recursively to color the neighborhood of large degree vertices using a small number of colors.

For the randomized rounding step, we observe that hyperplane rounding based procedures are naturally robust to small perturbations, and the arguments for analyzing the guarantees of such procedures hold even when the edge constraints are approximately satisfied. In particular, we can use known randomized rounding algorithm as is, while adapting the analysis to account for the edge constraints being satisfied approximately.

Semi-random model. While the guarantees of our algorithm from the adversarial setting also apply to the semi-random instances, here we seek to achieve the best known approximation bounds for exactly 3-colorable graphs. We begin by describing two distinct classes of instances which illustrate the technical challenges in designing such an algorithm.

In this setting, the adversary is free to choose $G[V_{\text{bad}}]$ in a way such that it is noisy and has large chromatic number (e.g, graphs sampled from Erdos Renyi random model). For such instances, it is easy to see that the only way an algorithm can have good approximation guarantees is when it can eliminate a significant fraction of from V_{bad}. Then, for a start, one can hope to address this setting by first using a preprocessing step that deletes V_{bad} and then running the best possible approximation algorithm on the graph induced on the remaining vertices.

On the other hand, the adversary can also choose $G[V_{\text{bad}}]$ in a way so that it is *structurally indistinguishable* from the good subgraph $G[V_{\text{good}}]$. For instance, suppose the good subgraph $G[V_{\text{good}}]$ is a randomly sampled unbalanced bipartite graph, where the smaller side (which we call V_S) has size at most ϵn. Then the adversary can choose V_{bad} to be an independent set, in which case the entire graph is 3-colorable. In particular, it is information theoretically impossible to distinguish the set V_S from V_{bad}, since they are both independent sets and the edges incident on them are identically distributed. While the instances constructed here make it difficult to identify V_{good}, they are also naturally easy instances for us. In particular, these instances are also $(1-\epsilon)$-partially 2-colorable, and one can use tools for coloring partially 2-colorable graphs to color these instances with small number of colors.

However, the two cases above clearly do not cover the full range of instances that we can encounter in our model. Therefore, we need a way to relax the above two characterizations which allows for a seamless transition from one class of instances to other. It turns out that we can robustly characterize both classes of instances by the number of *vertex disjoint short odd cycles* present in the graph. Informally, if the number of short odd cycles is large, then with high probability, they will show up in the neighborhood of the bad vertices, and therefore this can be used to identify and eliminate V_{bad}. We can then simply run the best known approximation algorithm on the remaining induced graph $G[V_{\text{good}}]$. On the other hand, if the number of short odd cycles is small, by eliminating a small fraction of vertices, we can make the graph short odd cycle free. Finally, as discussed in the adversarial model setting, such graphs can be colored efficiently using a small number of colors by recursively finding large independent sets [27].

2 Preliminaries

We introduce some notation used frequently in this paper. Throughout the paper, for a $(1-\epsilon)$-partially 3-colorable graph $G=(V,E)$, we will write $V = V_{\text{good}} \uplus V_{\text{bad}}$ where V_{good} and V_{bad} are the set of good vertices and bad vertices as defined in Definition 1. For a subset $V' \subseteq V$, we use $G[V']$ to denote the subgraph induced on the set of vertices V'. For a subgraph $G' \subseteq G$, we shall use $\text{vert}(G')$ to denote the vertex set of G'. Additionally, for any vertex $i \in \text{vert}(G')$, we use $N_{G'}(i)$ denote the set of neighbors of i in the graph G'. We use $\mathbb{1}(\cdot)$ to denote the indicator function, and $\tilde{O}(\cdot)$ to hide terms which are polylogarithmic in the number of vertices.

Approximate Vector Coloring

We begin by recalling the notion of vector coloring of a graph which was introduced in [17].

▶ **Definition 10** (Vector Coloring). *Given a positive integer $k \in \mathbb{N}$, we say that a graph $G=(V,E)$ is k-vector colorable if there exists unit vectors $v_1, v_2, \ldots, v_n \in \mathbb{R}^d$ for some $d \in \mathbb{N}$ which satisfy*

$$\langle v_i, v_j \rangle \leq -\frac{1}{k-1} \qquad \forall \{ i,j \} \in E.$$

We will use the notion of *approximate vector colorings* of a graph, which we define as follows.

▶ **Definition 11** (Approximate Vector Coloring). *Given a positive integer $k \in \mathbb{N}$ and a $\gamma > 0$, we say that a graph $G=(V,E)$ is (k,γ)-vector colorable if there exists unit vectors $v_1, v_2, \ldots, v_n \in \mathbb{R}^d$ for some $d \in \mathbb{N}$ which satisfy*

$$\langle v_i, v_j \rangle \leq -\frac{1}{k-1} + \gamma \qquad \forall \{ i,j \} \in E.$$

Observe that a graph that $(k, 0)$ vector colorable is vector-k-colorable. We now state a couple of lemmas which illustrate some useful properties of approximate vector colorings. In [17], it was observed that the vector chromatic number of sub-graph induced on the neighborhood of a vertex is strictly less than the vector chromatic number of the actual graph. In the following lemma, we observe that this property can be extended to approximate vector colorings as well.

▶ **Lemma 12.** *Let $G = (V, E)$ be $(3, \gamma)$-vector colorable, for some $0 < \gamma < 1/10$. Then for any vertex $i \in V$, the graph induced on $N(i)$ is $(2, 4\gamma)$-vector colorable.*

The next lemma says that approximately vector 2-colorable graphs cannot contain short odd cycles.

▶ **Lemma 13.** *Let $G = (V, E)$ be a $(2, \gamma)$-vector colorable, where $\gamma \leq 1/16$. Then G does not contain odd cycles of length at most $1/8\sqrt{\gamma}$.*

The proofs of the two lemmas above can be found in Appendix B.

Coloring graphs without short odd cycles

A key combinatorial tool used in our paper is the following Ramsey theoretic result which says that graphs without short odd cycles contain large independent sets which can be found efficiently.

▶ **Lemma 14** ([27])**.** *There exists a constant $\epsilon_0 \in (0, 1)$ such that for every choice of $0 < \epsilon < \epsilon_0$ the following holds. Let $G = (V, E)$ be a graph without odd cycles of length at most $1/\epsilon$. Then, G contains an independent set of size at least $|V|^{1-2\epsilon}$. Furthermore, there exists a polynomial time algorithm which finds such an independent set.*

Consequently, given a graph without short odd cycles, one can color it efficiently using a small number of colors, as stated in the following corollary.

▶ **Corollary 15.** *There exists a constant $\epsilon_0 \in (0, 1)$ for which the following holds. Given a graph $G = (V, E)$ which does not contain odd cycles of length at most $1/\epsilon$ where $\epsilon < \epsilon_0$, there exists a polynomial time algorithm which can compute a coloring of G using $\tilde{O}(n^{2\epsilon})$ colors.*

Establishing the above corollary using Lemma 14 is straightforward, and just uses the fact that one can keep removing large independent sets in the graph using Lemma 14, and recurse on the remaining vertices. For the sake of completeness, we include a proof in Appendix C.

3 Approximation algorithm for General Setting

In this section, we prove our approximation guarantees in the adversarial model, as formally stated in the following theorem:

▶ **Theorem 16** (Theorem 3 restated)**.** *There exists a polynomial time algorithm that takes as input a $(1-\epsilon)$-P3C graph $G = (V, E)$ and any fixed choice of $\gamma \in [\epsilon, 1/100]$, and produces a set $S \subset V$ such that $|S| \leq (3\epsilon/\gamma)|V|$ and a coloring of $V \setminus S$ using $\tilde{O}(n^{0.25+O(\gamma^{1/2})})$ colors.*

The algorithm for the above theorem is described in Algorithm 1. In the following subsections, we prove the correctness of the above algorithm. The proof of Theorem 3 can broken down into the analysis of steps (i),(ii) and (iii) of the Partial-3-Coloring algorithm. Broadly, we show the following: In step (i), we show that the optimal of the SDP-P3C is

Algorithm 1 Partial-3-Coloring.

Set $\Delta = n^{3/4}$;
Solve the Partial-3-Coloring SDP (SDP-P3C):

$$\begin{aligned}
\text{minimize } & \sum_{i \in V} w_i \\
\text{subject to } & \langle v_i, v_j \rangle \leq -\frac{1}{2} + \frac{3}{2} z_{ij} && \forall \{i,j\} \in E \\
& z_{ij} \leq w_i + w_j && \forall \{i,j\} \in E \\
& 0 \leq z_{ij} \leq 1 && \forall \{i,j\} \in E \\
& 0 \leq w_i \leq 1 && \forall i \in V \\
& \|v_i\|^2 = 1 && \forall i \in V
\end{aligned}$$

(i) Thresholding:
Let $S \leftarrow \{i \in V | w_i \geq \gamma/3\}$;
Let $G' \leftarrow G[V \setminus S]$ be the graph obtained after deleting S;
(ii) Coloring Large Degree vertices:
while $\exists i \in G'$ *such that* $\deg_{G'}(i) \geq \Delta$ **do**
  Color $G'[\{i\} \cup N_{G'}(i)]$ using $\tilde{O}(n^{C\sqrt{\gamma}})$ colors using the algorithm guaranteed by Corollary 15;
  Remove $\{i\} \cup N_{G'}(i)$ from G';
end
(iii) Coloring Low Degree vertices:
Use *randomized rounding* from Theorem 19 to color the remaining vertices in G';

small (i.e., at most ϵn), therefore by averaging, the fraction of large w vertices is small. Furthermore, the graph induced on the surviving vertices must satisfy the edge constraints from the SDP with small slack γ, and therefore must be approximately vector 3-colorable. As is usual in coloring algorithms, we first iteratively color large degree (i.e., $\geq \Delta$) vertices and their neighborhoods using small number of colors until the graph has degree bounded by Δ (Claim 18). Finally, the remaining graph is also approximately vector 3-colorable, and has degree bounded by Δ. Therefore, using a hyperplane based randomized rounding procedure to iteratively find large independent sets in G', we can give a $\tilde{O}(\Delta^{1/3+O(\sqrt{\gamma})})$ coloring of the remaining vertices (Theorem 19). In the following subsection, we formally prove the steps described above.

To begin with, we first show that the thresholding step throws away at most a small fraction of vertices.

▷ Claim 17 (Removing Large Slack Vertices). Let $S \subset V$ be as constructed in the thresholding step. Then $|S| \leq 3\epsilon n/\gamma$.

We defer the proof of the above claim to Appendix A. From the above claim, the graph $G' = G[V \setminus S]$ induced on the remaining vertices satisfies the following properties:

1. The graph G' contains at least $(1 - 3\epsilon/\gamma)n$ vertices.
2. The graph G' is $(3, \gamma)$-vector colorable. In particular, the vectors $(v_i)_{i \in V \setminus S}$ themselves are a $(3, \gamma)$-vector coloring of G'.

The second point shall be used crucially in the analysis of the remaining two steps. The next claim bounds the number of colors used while coloring the large degree vertices in step (ii).

▷ Claim 18 (Degree Reduction). In step (ii), over all the iterations of the while loop, the algorithm uses at most $(n/\Delta)\tilde{O}\left(n^{C\sqrt{\gamma}}\right)$ colors, where $C > 0$ is a constant.

Proof. Fix any vertex $i \in G'$, and let $\tilde{G}_i = G'[N(i)]$ the graph induced on the neighborhood of vertex i. Since the graph G' is $(3,\gamma)$-vector colorable, using Lemma 12 we know that $\tilde{G}_i$ is $(2, 4\gamma)$-vector colorable. Furthermore, from Lemma 13, we know that G' does not contain odd cycles of length at most $1/(8\sqrt{4\gamma})$. Therefore, we can use Corollary 15 to obtain a $\tilde{O}(n^{C\sqrt{\gamma}})$ coloring of $\tilde{G}_i \cup \{i\}$. Finally, note that each iteration of the for loop removes and colors at least $\Delta + 1$ vertices of the graph. Therefore, the total number of iterations of the for loop is bounded by n/Δ. Since in each such iteration we can color the vertex and its neighborhood using $n^{C\sqrt{\gamma}}$ number of colors, the claim follows. ◁

After steps (i) and (ii), we are left with the graph $G' = (V', E')$ which is $(3,\gamma)$-vector colorable graph and has degree at most Δ. In particular, for every edge $(i,j) \in E'$, the corresponding vectors satisfy $\langle v_i, v_j\rangle \leq -\frac{1}{2} + \gamma$. Since the independent set based rounding technique [17] [3] for coloring vector 3-colorable graphs is *robust*, we can still use it to round the vector coloring of approximately 3-colorable graphs with similar guarantees, as formally stated in the following theorem.

▶ **Theorem 19.** *Let $G = (V, E)$ be a graph with maximum degree Δ which is $(3,\alpha)$-vector colorable. Then there exists an efficient randomized algorithm that can color it using $O\left((\ln\Delta)^{1/2}\Delta^{\frac{\frac{3}{4}+\alpha-\alpha^2}{(\frac{3}{2}-\alpha)^2}}\ln n\right)$ colors.*

In particular, if $\alpha \leq 1/10$, then the algorithm uses at most $\tilde{O}\left((\ln\Delta)^{1/2}\Delta^{\frac{1}{3}+10\alpha}\right)$, where $\tilde{O}$ hides polylogarithmic factors in n.

The proof of the above theorem is an extension of the proofs from [17, 3] to the setting of approximately vector 3-colorable graphs. Due to space constraints, we skip the proof here and provide it in the full version. Here for simplicity assume that $\gamma \leq 1/10$. Instantiating the above theorem with $G = G'$ and $\alpha = \gamma$, we get that G' is colored using $\tilde{O}(\Delta^{1/3+10\gamma})$ colors. Overall, the algorithm throws away at most $2\epsilon/\gamma$ fraction of vertices in step (i). Furthermore, it uses a total of $\tilde{O}\left((n/\Delta)n^{O(\sqrt{\gamma})} + \Delta^{1/3+10\gamma}\right)$ colors in steps (ii) and (iii) respectively. Setting $\Delta = n^{3/4}$ in the previous expression, we get that the algorithm uses at most $\tilde{O}(n^{1/4+O(\sqrt{\gamma})})$ colors. This concludes the analysis of the Partial-3-Coloring algorithm and the proof of Theorem 3.

4 Algorithm for Semi-random instances

In this section, we prove Theorem 6, which we again state here for convenience.

▶ **Theorem 20** (Theorem 6 restated)**.** *Suppose there exists an efficient algorithm which colors a 3-colorable graph using n^θ colors. Then the following holds for all choices of $\epsilon = \Omega(\log n/n)$ and $p \geq (\epsilon\theta^{-2})^{O(\theta)}$. There exists a polynomial time algorithm that takes as input a graph G sampled from $(1-\epsilon)$-$P3C^{\mathcal{R}}(n,p)$ and produces a set S such that $|S| = O\left(\epsilon\theta^{-2}np^{-(O(1/\theta))}\right)$ and a coloring of $V \setminus S$ using at most n^θ colors with high probability. Moreover, the algorithm runs in time $n^{O(1/\theta)}\mathrm{poly}(n)$.*

Algorithm 2 P3C-Random.

Let $\mathcal{A}$ be the algorithm which can color 3-colorable graphs using n^θ colors;
Set $\delta = \theta/10$;

{*Many short odd cycles*}:
for *every vertex* $v \in V$ **do**
  Let $G_v := G[N_G(v)]$ the subgraph induced by the neighborhood of G;
  Greedily construct a maximal set $\mathcal{C}_v$ of vertex disjoint odd cycles of length at most $1/\delta$ in G_v;
end
Construct set $S \leftarrow \{v \in V : |\mathcal{C}_v| \geq 2\epsilon n\}$;
Let $G_0 \leftarrow G[V \setminus S]$ be the graph obtained after deleting S;
Let σ_1 be the coloring of $V \setminus S$ obtained by running algorithm $\mathcal{A}$ on G_0. Let L denote the number of colors used by the algorithm;

{*Few short odd cycles*}:
Compute a maximal set $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ of vertex disjoint odd cycles in G of length at most $1/\delta$ using greedy algorithm;
Let $V' = V \setminus \left(\bigcup_{i\in[m]} \mathrm{vert}(C_i)\right)$;
Use the algorithm guaranteed by Corollary 15 to give a $\tilde{O}\left(n^{4\delta}\right)$ coloring σ_2 of $G[V']$;

{*Output best coloring*}:
if $|S| \leq \epsilon n$ *and* $L \leq n^\theta$ **then**
  Output coloring σ_1 of $V \setminus S$
end
else
  Output coloring σ_2 of V';
end

We begin by describing the algorithm for the semi-random setting:

The algorithm proceeds case wise depending on whether there exists many vertex disjoint short odd cycles in G. If it does, then since V_{bad} is small, $G[V_{\mathrm{good}}]$ must also contain many vertex disjoint odd cycles. We show that these short cycles will show up in the neighborhood of the bad vertices with high probability, which can be used to identify them. On removing these vertices, we will be left with a 3-colorable graph. On the other hand, if the number of short odd cycles is small, we can remove them. The remaining graph will still contain most of the vertices and will be short odd cycle free. We can then use Lemma 14 to recover large independent sets. Finally, since the odd cycles we consider are of length at most $1/\delta$, we can work with a *maximal* set of vertex disjoint odd cycles, instead of the largest cardinality set of vertex disjoint odd cycles, while only losing a factor of $1/\delta$ in our analysis.

4.1 Correctness of the P3C-Random algorithm

Let $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_{m^*}^*\}$ be a *fixed largest cardinality set of vertex disjoint odd cycles* of length at most $1/\delta$ in $G[V_{\mathrm{good}}]$. *In particular,* $\mathcal{C}^*$ *and consequently* m^**, does not depend on the realization of the random and adversarial edges (i.e., the* E_0 *and* E_1 *edges) between* V_{good} *and* V_{bad}. We break our analysis into two cases depending on whether m^* is small or large.

Case (i): $m^* > 4\epsilon n/(\delta p^{1/\delta})$. For ease of exposition, we say that an odd cycle C in graph G is *good* if it consists of only good vertices, otherwise we call it *bad*. The first claim shows the set $\mathcal{C}_v$ must be small for good vertices.

▷ Claim 21. For every good vertex $v \in V$, we have $|\mathcal{C}_v| \leq \epsilon n$.

Proof. Fix a good vertex $v \in V_{\text{good}}$. We claim that a good cycle C can never appear in the neighborhood of a good vertex. For contradiction, let C be a good odd cycle appearing in the neighborhood of v. Let $\tilde{G} = G\Big[\text{vert}(C) \cup \{v\}\Big]$ be the subgraph induced on the vertex v and the vertices from cycle C. Since $\tilde{G} \subseteq G[V_{\text{good}}]$, the subgraph $\tilde{G}$ is also 3-colorable. Hence, the neighborhood of v in the induced subgraph $\tilde{G}$ must be 2-colorable, and therefore it cannot contain odd cycles, and in particular C. This gives us the contradiction.

Hence, any odd cycle which appears in the neighborhood $N_G(v)$ must be bad. Since the number of bad vertices is bounded by ϵn, and the cycles in $\mathcal{C}_v$ are vertex disjoint, the claim follows. ◁

On the other hand, with high probability, we show that $|\mathcal{C}_v|$ is large for all the bad vertices.

▷ Claim 22. With probability at least $1 - e^{-O(\epsilon n)}$, every vertex $v \in V_{\text{bad}}$ satisfies $|\mathcal{C}_v| \geq 2\epsilon n$.

Proof. Consider the subgraph $G'(V, E_0)$ consisting of edges from E_0 (i.e., the randomly distributed set of edges). Fix a bad vertex $v \in V_{\text{bad}}$, and let $G_v = G[N_G(v)]$ denote the subgraph induced by the neighborhood of v. We shall first give a high probability lower bound on the number of odd cycles from $\mathcal{C}^*$ which can appear in $N_G(v)$. Recall that $|\mathcal{C}^*| = m^*$. We also point out again that the choice of $\mathcal{C}^*$ is not affected by the choice of E_0 and E_1 edges, and can be fixed ahead.

For every $i \in [m^*]$, we define $Z_i := \mathbb{1}\Big(\text{vert}(C_i^*) \subseteq N_{G'}(v)\Big)$ to be the indicator random variable that the i^{th} cycle appears in the neighborhood of vertex v in the graph G'. Note that these random variables depend only on the realization of the E_0 edges. Then we have

$$\begin{aligned}
\mathbb{E}_G[Z_i] \geq \Pr_{E_0}\Big[\text{vert}(C_i^*) \subseteq N_G(v)\Big] &\geq \Pr_{E_0}\Big[\text{vert}(C_i^*) \subseteq N_{G'}(v)\Big] \\
&= \Pr_{E_0}\Big[\forall j \in \text{vert}(C_i^*), j \in N_{G'}(v)\Big] \\
&\geq p^{|C_i^*|} \geq p^{1/\delta}
\end{aligned}$$

Here the last step uses the fact that any cycle $C_i^* \in \mathcal{C}^*$ has length at most $1/\delta$. It follows that

$$\mathbb{E}_G\left[\sum_{i\in[m^*]} Z_i\right] = \sum_{i\in[m^*]} \mathbb{E}_G[Z_i] \geq m^* p^{1/\delta} \geq (4\epsilon/\delta)n \tag{1}$$

Furthermore, since the cycles $C_1^*, C_2^*, \ldots, C_{m^*}^*$ are vertex disjoint, the corresponding random variables $Z_1, Z_2, \ldots, Z_{m^*}$ are also independent. Therefore using Chernoff bound we get that

$$\Pr_G\left[\sum_{i\in[m^*]} Z_i < (2\epsilon/\delta)n\right] \leq \Pr_G\left[\sum_{i\in[m^*]} Z_i < \frac{1}{2}\mathbb{E}\Big[\sum_{i\in[m^*]} Z_i\Big]\right] \leq e^{-\epsilon n/4\delta} \tag{2}$$

Now let $\mathcal{C}_v^* = \{C_i^* : i \in [m^*], Z_i = 1\}$ be the set of cycles from $\mathcal{C}^*$ which appear in the neighborhood of v in graph G due to the E_0 edges. Furthermore, let $\widetilde{\mathcal{C}}_v$ be a *largest cardinality* set of vertex disjoint odd cycles of length at most $1/\delta$ in G_v (which contains edges from both E_0 and E_1). Then by definition we have $|\widetilde{\mathcal{C}}_v| \geq |\mathcal{C}_v^*|$. On the other hand, by construction, the set $\mathcal{C}_v$ is a *maximal set* of such vertex disjoint odd cycles in G_v, and therefore, it must be a δ-approximation to the largest cardinality set $\widetilde{\mathcal{C}}_v$ i.e., $|\mathcal{C}_v| \geq \delta|\widetilde{\mathcal{C}}_v|$ (see Proposition 26). Therefore using Equation 2, with probability at least $1 - e^{-\epsilon n/4\delta}$ we have

$$|\mathcal{C}_v| \geq \delta|\widetilde{\mathcal{C}}_v| \geq \delta|\mathcal{C}_v^*| \geq 2\epsilon n$$

Hence, for any fixed vertex $v \in V_{\text{bad}}$, w.h.p. we have $|\mathcal{C}_v| \geq 2\epsilon n$. Therefore, by a union bound and using the lower bound on ϵ, we get that $\Pr_G\left[\exists v \in V_{\text{bad}} : |\mathcal{C}_v| < 2\epsilon n\right] \leq \epsilon n e^{-\epsilon n/4\delta} \leq n e^{-\epsilon n/8\delta}$. ◁

Combining the two claims above, it follows that w.h.p. the set $(V \setminus S)$ must exactly be the set of good vertices, and therefore $G[V \setminus S]$ must be 3-colorable. Hence algorithm $\mathcal{A}$ will give a n^θ coloring of $G[V \setminus S]$.

Case (ii): $m^* \leq 4\epsilon n/(\delta p^{1/\delta})$. Let $\mathcal{C} = \mathcal{C}_{\text{good}} \uplus \mathcal{C}_{\text{bad}}$ be the partition of $\mathcal{C}$ into the set of good and bad cycles respectively. Then, since $\mathcal{C}_{\text{good}}$ is a set of vertex disjoint odd cycles of length at most $1/\delta$ in $G[V_{\text{good}}]$, it follows that $|\mathcal{C}_{\text{good}}| \leq |\mathcal{C}^*| \leq 4\epsilon n/(\delta p^{1/\delta})$. Furthermore, by arguments similar to the proof of Claim 21, we have $|\mathcal{C}_{\text{bad}}| \leq \epsilon n$. Therefore, combining the two bounds, we have $|\mathcal{C}| \leq 5\epsilon n/(\delta p^{1/\delta})$. Since every cycle $C \in \mathcal{C}$ contains at most $1/\delta$ vertices, the total number of vertices thrown away at this step is at most $5\epsilon n/(\delta^2 p^{1/\delta})$. Furthermore, using the *maximality* of $\mathcal{C}$, we know that the induced subgraph $G' = G[V']$ must be free of odd cycles of length at most $1/\delta$. Therefore, using Corollary 15, we can color G' using $\tilde{O}(n^{2\delta})$ colors. This concludes the analysis of case (ii).

Putting Things Together. If case (i) holds, then w.h.p., in the *Many short odd cycles* block of the algorithm, the set S constructed is identical to V_{bad}, in which case the algorithm $\mathcal{A}$ will find a n^θ-coloring of $G[V \setminus S] = G[V_{\text{good}}]$. In particular, this implies that the conditions of the "if" block will be satisfied and the algorithm will return a n^θ-coloring of $(1-\epsilon)n$ vertices.

On the other hand, if case (ii) holds, we know that $m \leq 5\epsilon n/(p^{1/\delta}\delta)$, and the *Few short odd cycles* block deletes at most $5\epsilon n/(p^{1/\delta}\delta^2)$ vertices, and colors the remaining vertices using $\tilde{O}(n^{2\delta})$ colors. Then the *else* block of the algorithm will return a $\tilde{O}(n^{2\delta})$ coloring of $\left(1 - 5\epsilon n/(p^{1/\delta}\delta^2)\right)n$ vertices. Since the else block is evaluated only when the conditions of the *if* block are not satisfied, it follows that in this case, the algorithm will throw away at most $\max\left(\epsilon n, 5\epsilon n/(\delta^2 p^{1/\delta})\right) = O(\epsilon n/\delta^2 p^{1/\delta})$ vertices, and color the remaining graph with at most $\max\left(n^\theta, \tilde{O}(n^{2\delta})\right) = n^\theta$ colors.

Combining the two above cases gives us Theorem 6.

5 Conclusion

In this work we consider the problem of coloring partial 3-colorable graphs in adversarial and semi-random settings. In the adversarial setting, we give an efficient approximation algorithm which can color $(1 - O(\epsilon^c))$-fraction of vertices using $\tilde{O}(n^{0.25+\epsilon^{c'}})$ colors. On the other hand, the best known approximation guarantees for 3-colorable graphs is $n^{0.199}$ [20]. An obvious open question here is to achieve analogous approximation bounds for partially 3-colorable graphs as well.

One direct way to improve on our approximation bounds in the adversarial setting is through the use of more efficient degree reduction mechanisms as typically done in the exact 3-coloring setting [10],[19, 20] using combinatorial techniques like Blum's coloring tools [8]. However, these tools rely on fragile combinatorial properties present in 3-colorable graphs (e.g. two vertices whose common neighborhood is not an independent set must have the same color in any legal coloring), and as such, it is not obvious how to extend these techniques to the setting of partially 3-colorable graphs.

In the semi-random model, we show how any efficient algorithm for exact 3-coloring that uses n^θ colors can be leveraged to obtain an efficient algorithm in this setting which uses the same number of colors with high probability and also does not remove too many vertices. An obvious next step would be to see if similar results can also be obtained for partially k-colorable graphs with $k > 3$. Another interesting question would be to see if one can design efficient approximation algorithms with similar guarantees, where the adversary can also delete the randomly sampled edges.

References

1 Amit Agarwal, Moses Charikar, Konstantin Makarychev, and Yury Makarychev. O(sqrtlog n)) approximation algorithms for min UnCut, min 2CNF deletion, and directed cut problems. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 573–581. ACM, 2005.

2 Noga Alon and Nabil Kahale. A spectral technique for coloring random 3-colorable graphs. *SIAM Journal on Computing*, 26(6):1733–1748, 1997.

3 Sanjeev Arora and Eden Chlamtac. New approximation guarantee for chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 215–224. ACM, 2006.

4 Sanjeev Arora and Rong Ge. New Tools for Graph Coloring. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, pages 1–12, 2011.

5 Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM (JACM)*, 56(2):5, 2009.

6 Nikhil Bansal and Subhash Khot. Optimal long code test with one free bit. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 453–462. IEEE, 2009.

7 Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding Semidefinite Programming Hierarchies via Global Correlation. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 472–481, 2011.

8 Avrim Blum. Some tools for approximate 3-coloring. In *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*, pages 554–562. IEEE, 1990.

9 Avrim Blum. New Approximation Algorithms for Graph Coloring. *J. ACM*, 41(3):470–516, 1994. doi:10.1145/176584.176586.

10 Avrim Blum and David Karger. An algorithm for 3-colorable graphs. *Information Processing Letters*, 61(1):49–53, 1997. doi:10.1016/s0020-0190(96)00190-1.

11 Avrim Blum and Joel Spencer. Coloring random and semi-random k-colorable graphs. *Journal of Algorithms*, 19(2):204–234, 1995.

12 Eden Chlamtac. Approximation Algorithms Using Hierarchies of Semidefinite Programming Relaxations. *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS07)*, 2007. doi:10.1109/focs.2007.72.

13 Roee David and Uriel Feige. On the effect of randomness on planted 3-coloring models. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 77–90. ACM, 2016.

14 Reinhard Diestel. Graph Theory, volume 173 of. *Graduate texts in mathematics*, page 7, 2012.

15 Uriel Feige and Joe Kilian. Heuristics for semirandom graph problems. *Journal of Computer and System Sciences*, 63(4):639–671, 2001.

16 Naveen Garg, Vijay V Vazirani, and Mihalis Yannakakis. Approximate max-flow min-(multi) cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.

17 David Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. *Journal of the ACM (JACM)*, 45(2):246–265, 1998.

18 Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

19 Ken-Ichi Kawarabayashi and Mikkel Thorup. Combinatorial Coloring of 3-Colorable Graphs. *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, 2012. doi:10.1109/focs.2012.16.

20 Ken-Ichi Kawarabayashi and Mikkel Thorup. Coloring 3-Colorable Graphs with Less than $n^{1/5}$ Colors. *Journal of the ACM*, 64(1):1–23, 2017. doi:10.1145/3001582.

21 Alexandra Kolla, Konstantin Makarychev, and Yury Makarychev. How to play unique games against a semi-random adversary: Study of semi-random models of unique games. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 443–452. IEEE, 2011.

22 Akash Kumar, Anand Louis, and Madhur Tulsiani. Finding Pseudorandom Colorings of Pseudorandom Graphs. In *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*, pages 37:1–37:12, 2017.

23 Daniel Lokshtanov, NS Narayanaswamy, Venkatesh Raman, MS Ramanujan, and Saket Saurabh. Faster parameterized algorithms using linear programming. *ACM Transactions on Algorithms (TALG)*, 11(2):15, 2014.

24 Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Approximation algorithms for semi-random partitioning problems. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 367–384, 2012.

25 Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Algorithms for Semi-random Correlation Clustering. *CoRR*, abs/1406.5667, 2014. arXiv:1406.5667.

26 Theo McKenzie, Hermish Mehta, and Luca Trevisan. A New Algorithm for the Robust Semi-random Independent Set Problem. *CoRR*, abs/1808.03633, 2018. arXiv:1808.03633.

27 Burkhard Monien and Ewald Speckenmeyer. Ramsey numbers and an approximation algorithm for the vertex cover problem. *Acta Informatica*, 22(1):115–123, 1985.

28 NS Narayanaswamy, Venkatesh Raman, MS Ramanujan, and Saket Saurabh. LP can be a cure for parameterized problems. In *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)*, volume 14, pages 338–349. LIPIcs, 2012.

29 Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 245–254, 2008.

30 Prasad Raghavendra and David Steurer. How to Round Any CSP. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, pages 586–594, 2009.

31 Bruce Reed, Kaleigh Smith, and Adrian Vetta. Finding odd cycle transversals. *Operations Research Letters*, 32(4):299–301, 2004.

32 Jacob Steinhardt. Does robustness imply tractability? A lower bound for planted clique in the semi-random model. *arXiv preprint*, 2017. arXiv:1704.05120.

33 Avi Wigderson. Improving the performance guarantee for approximate graph coloring. *Journal of the ACM*, 30(4):729–735, January 1983. doi:10.1145/2157.2158.

34 Mihalis Yannakakis. Node-and edge-deletion NP-complete problems. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 253–264. ACM, 1978.

A Proof of Claim 17

We begin by showing that the optimal of SDP-P3C is at most ϵn. Let $V = V_{\text{good}} \cup V_{\text{bad}}$ be any partition of the vertex sets into good and bad vertices such that (a) $G[V_{\text{good}}]$ is 3-colorable and (b) $|V_{\text{bad}}| \leq \epsilon n$. Using this partition we now construct a 2-dimensional feasible solution $(\widehat{v}, \widehat{w}, \widehat{z})$ to SDP-P3C as follows. We set the $\widehat{w}_i$ and $\widehat{z}_{ij}$ variables as

$$\widehat{w}_i = \begin{cases} 0, & \text{if } i \in V_{\text{good}} \\ 1, & \text{otherwise} \end{cases} \qquad \text{and} \qquad \widehat{z}_{ij} = \begin{cases} 0, & \text{if } i,j \in V_{\text{good}} \\ 1, & \text{otherwise} \end{cases}$$

Furthermore, we set $\{\widehat{v}_i\}_{i \in V_{\text{good}}}$ be a vector 3-coloring of $G[V_{\text{good}}]$, and for every $i \in V_{\text{bad}}$ we set $\widehat{v}_i = [1 \quad 0]$. We quickly verify that the $\widehat{v}, \widehat{w}$ and the $\widehat{z}$ variables constructed as above form a feasible solution to the SDP. By construction, for every $i \in V$ we have $\widehat{w}_i \in [0,1]$ and $\|\widehat{v}_i\|^2 = 1$, and for every edge $(i,j) \in E$ we have $z_{ij} \in [0,1]$. Furthermore, for any edge (i,j) we also have

$$\widehat{z}_{ij} = \mathbb{1}\Big(\{i \in V_{\text{bad}}\} \vee \{j \in V_{\text{bad}}\} \Big) \leq \mathbb{1}\big(\{i \in V_{\text{bad}}\} \big) + \mathbb{1}\big(\{j \in V_{\text{bad}}\} \big) = \widehat{w}_i + \widehat{w}_j$$

All that remains to verify is that the variables also satisfy the approximate vector coloring constraints. We look at two cases: if $i,j \in V_{\text{good}}$, then $\widehat{v}_i, \widehat{v}_j$ come from the vector 3-coloring of $G[V_{\text{good}}]$ and therefore they satisfy $\langle \widehat{v}_i, \widehat{v}_j \rangle \leq -\frac{1}{2} \leq -\frac{1}{2} + \widehat{z}_{ij}$. On the other hand if $i \in V_{\text{bad}}$ or $j \in V_{\text{bad}}$ then by construction we have $\widehat{z}_{ij} = 1$, and therefore $\langle \widehat{v}_i, \widehat{v}_j \rangle \leq \|\widehat{v}_i\| \|\widehat{v}_j\| = 1 = -\frac{1}{2} + \frac{3}{2}\widehat{z}_{ij}$.

Therefore, we have established that $(\widehat{z}, \widehat{w}, \widehat{v})$ are a feasible solution for SDP-P3C. Since by construction $\widehat{w}_i = \mathbb{1}\{i \in V_{\text{good}}\}$, and the $|V_{\text{bad}}| \leq \epsilon n$, it follows that the SDP optimal $\sum_{i \in V} w_i$ is at most $\sum_{i \in V} \widehat{w}_i \leq \epsilon n$. Therefore, using Markov's inequality, we get

$$|S| = n \cdot \Pr_{i \sim V}\Big[w_i \geq \gamma/3 \Big] \leq n \cdot \frac{3 \sum_{i \in V} w_i}{n\gamma} = \frac{3\epsilon n}{\gamma}$$

B Auxiliary Lemmas

In this section we give the proofs of Lemmas 12 and 13.

B.1 Proof of Lemma 12

The proof of this lemma follows along the lines of Lemma 4.3 from [17], which says that subgraphs induced by neighborhoods of vertices in vector 3-colorable graphs are vector 2-colorable. Without loss of generality, let $N_G(i) = \{1, 2, \ldots, r\}$ and let $\{v_1, v_2, \ldots, v_r\}$ be the set of vectors which are a $(3,\gamma)$-vector coloring of $N_G(i)$. For every $j \in [r]$, we can write $v_j = v_j^{\|} + v_j^{\perp}$ where $v_j^{\|}$ and $v_j^{\perp}$ are the projections of v_j along v_i and $(\text{span}(v_i))^{\perp}$ respectively. Finally, for every $j \in [r]$ we define $\tilde{v}_j := v_j^{\perp}/\|v_j^{\perp}\|$ to be unit vector given by the projection of v_j on the subspace $(\text{span}(v_i))^{\perp}$. It can be easily verified that $\tilde{v}_1, \tilde{v}_2, \ldots, \tilde{v}_r$ is a $(2, 4\gamma)$-vector coloring of the graph induced on $N(v)$. To see this, fix any $j \in V$. By construction, we have $\|v_j^{\|}\| = |\langle v_i, v_j \rangle| \geq \frac{1}{2} - \gamma$, and therefore $\|v_j^{\perp}\| = \sqrt{1 - \|v_j^{\|}\|^2} \leq \sqrt{\frac{3}{4} + \gamma - \gamma^2}$. Therefore for any $j, j' \in [r]$ such that $(j,j') \in E$, using the orthonormal decomposition of v_j and $v_{j'}$ we have

$$\begin{aligned}\langle \tilde{v}_j, \tilde{v}_{j'} \rangle = \left\langle \frac{v_j^\perp}{\|v_j^\perp\|}, \frac{v_{j'}^\perp}{\|v_{j'}^\perp\|} \right\rangle &= \frac{1}{\|v_j^\perp\|\|v_{j'}^\perp\|}\Big(\langle v_j, v_{j'}\rangle - \langle v_j^\parallel, v_{j'}^\parallel\rangle\Big) \\ &= \frac{1}{\|v_j^\perp\|\|v_{j'}^\perp\|}\Big(\langle v_j, v_{j'}\rangle - \langle v_i, v_j\rangle\langle v_i, v_{j'}\rangle\Big) \\ &\leq \frac{1}{\left(\frac{3}{4}+\gamma-\gamma^2\right)}\Big(-1/2+\gamma-\Big(\frac{1}{2}-\gamma\Big)^2\Big) \\ &\leq -1+4\gamma\end{aligned}$$

Since the above holds for any pair of vertices $j, j' \in [r]$ which forms an edge, the claim follows.

B.2 Proof of Lemma 13

Let $v_1, v_2, \ldots, v_n$ be the $(2,\gamma)$-vector coloring of G. For contradiction, let C be an odd cycle in G of length $r \leq 1/(8\sqrt{\gamma})$. Without loss of generality, let $C = \{1, 2, \ldots, r\}$, such that for every $i \in [r]$, the pair $\{i, (i \mod r) + 1\}$ forms an edge. Let $r = 2k+1$. Now for any $i \in [r]$, we have $-1 \leq \langle v_i, v_{i+1}\rangle \leq -1+\gamma$. Since v_i, v_{i+1} are unit vectors, we have

$$\|v_i + v_{i+1}\|^2 = \|v_i\|^2 + \|v_{i+1}\|^2 + 2\langle v_i, v_{i+1}\rangle \leq 2\gamma \tag{3}$$

which implies that $\|v_i + v_{i+1}\| \leq 2\sqrt{\gamma}$ i.e, any consecutive pair of vectors are *almost anti-podal*. Then, for any $i \in [r]$ we also get that

$$\|v_i - v_{i+2}\| \leq \|v_i + v_{i+1}\| + \|v_{i+1} + v_{i+2}\| \leq 4\sqrt{\gamma} \tag{4}$$

We shall now use the above observations to arrive at a contradiction. From the upper bound on r, we have $k \leq (r-1)/2 \leq 1/(16\sqrt{\gamma})$, and hence using Eq. 4 we get that

$$\|v_1 - v_r\| \leq \sum_{j=0}^{k-1} \|v_{1+2j} - v_{1+2(j+1)}\| \leq 4k\sqrt{\gamma} < 1/4 \tag{5}$$

But on the other hand, since v_1, v_r are consecutive vertices in the cycles C, we also have $\langle v_1, v_r\rangle \leq -1+\gamma$ which implies that $\|v_1 - v_r\| \geq \sqrt{4-4\gamma} > 1$, which give us the contradiction.

C Proof of Corollary 15

Consider Algorithm IndSetColoring for coloring by iteratively finding large independent sets.

Algorithm 3 IndSetColoring.

Input: Graph $G = (V, E)$
Initialize $t \leftarrow 1$ and $G_1 \leftarrow G$;
while $G_t \neq \phi$ **do**
  Let I_t be the independent set from Lemma 14 instantiated with G_t;
  Set $G_{t+1} \leftarrow G_t \setminus I_t$;
  Update $t \leftarrow t+1$;
end
Output coloring $I_1 \uplus I_2 \uplus \cdots \uplus I_t$;

In the above algorithm, we use Lemma 14 to iteratively remove independent sets $I_1, I_2, \ldots, I_t$, where each independent set forms a color class. Let $G_t = G[V \setminus (I_1 \cup I_2 \cup \cdots I_t)]$ denote the graph on the surviving vertices after t iterations. We claim that in every $T = n^{2\epsilon}$ applications of Lemma 14 at least a constant fraction of vertices are removed, i.e., for any iteration t, we have $|\text{Vert}(G_{t+T})| \leq (1 - 1/2^{1-2\epsilon})|\text{Vert}(G_t)|$.

This can be shown as follows. Let $n_t = |\text{Vert}(G_t)|$ denote the number of vertices in graph G_t. Then, we can assume that $|\text{vert}(G_{t+T})| > n_t/2$ (otherwise we are done). Then, in T iterations the number of vertices removed can be lower bounded by

$$\sum_{j=1}^{T} |I_{j+T}| \geq \sum_{j=1}^{T} |\text{Vert}(G_{t+j})|^{1-2\epsilon} \geq n^{2\epsilon}(n_t/2)^{1-2\epsilon} \geq n_t/2^{(1-2\epsilon)} \tag{6}$$

where the first inequality follows from the guarantee of Lemma 14. Therefore, in $\tilde{O}(n^{2\epsilon})$ iterations, all the vertices will be accounted for.

D Partial 2-Coloring in the Semi-random model

In this section, we give an efficient approximation algorithm for partial 2-coloring problem in the semi-random model with tighter guarantees. The following theorem formally states our guarantees for this setting.

▶ **Theorem 23** (Theorem 7 restated). *Let $\epsilon = \Omega(\log n/n)$ and $p > \sqrt{\epsilon}$. Then, there exists a polynomial time algorithm that takes as input a graph G sampled from $(1-\epsilon)$-P2C$^{\mathcal{R}}$ (n,p) , and with high probability, produces a set $S \subseteq V$ such that $|S| = O\left(\epsilon np^{-2}\right)$ and the induced subgraph on the remaining vertices $G[V \setminus S]$ is 2-colorable.*

The algorithm for the above theorem (described as Algorithm 4) is quite similar to P3C-Random algorithm, but overall, the algorithm and its analysis are much simpler. We begin by describing the algorithm.

Algorithm 4 P2C-Random.

For every vertex $v \in V$, compute a greedy triangle count as follows:
for $v \in V$ **do**
  Let $G_v = G[N_G(v)]$ be the graph induced on the neighborhood of v;
  Construct a maximal matching $T(v)$ in G_v using greedy algorithm;
  Set $t(v) \leftarrow |T(v)|$;
end
Let $S \leftarrow \{v \in V : t(v) \geq 2\epsilon n\}$;
Let $G_0 = G[V \setminus S]$;
Let $G_1 \subseteq G$ be the independent set obtained using the 2-factor approximation for Vertex Cover on G;
if $|\text{vert}(G_0)| \geq |\text{vert}(G_1)|$ *and* G_0 *is bipartite* **then**
  Output bipartite graph G_0;
end
else
  Output independent set G_1;
end

The key difference here is that the algorithm uses triangles as forbidden subgraphs for identifying bad vertices instead of neighborhoods with short odd cycles. As before, the algorithm broadly addresses two cases depending on the size of the maximum matching in $G[V_{\text{good}}]$. Suppose the subgraph $G[V_{\text{good}}]$ contains a linear sized matching M. Then, for every bad vertex $v \in V_{\text{bad}}$, with high probability, at least one of the matching edges from M will appear in the neighborhood of v, which together will form a triangle, which can then be used to identify the bad vertices. On the other hand, if the size of maximum matching in $G[V_{\text{good}}]$ is small, then the subgraph $G[V_{\text{good}}]$ and consequently G must admit a small sized vertex cover. Therefore, using the greedy approximation algorithm for vertex cover, we can find a small sized vertex cover, whose complement must be a large independent set (which is 1-colorable).

D.1 Proof of Theorem 7

Let $M \subseteq G[V_{\text{good}}]$ be a *fixed matching of maximum size* in $G[V_{\text{good}}]$, and let $m^* := |M|$ denote the size of the maximum matching. *We point out that the matching M^* is not affected by the realization of edges between V_{good} and V_{bad} (i.e, the E_0 and E_1 edges).* As before, we break the analysis into two cases depending on whether m^* is small or large.

Case (i): $m^* \geq (8\epsilon/p^2)n$. This case is similar to case (i) of the proof of Theorem 6. We begin by stating and proving two lemmas which say that the greedy triangle count $t(v)$ is small for all the good vertices, and large for all the bad vertices.

▶ **Lemma 24.** *For every good vertex $v \in V_{\text{good}}$, we have $t(v) \leq \epsilon n$*

Proof. Fix a good vertex $v \in V_{\text{good}}$, and let $T(v)$ be a set of edges as constructed in the algorithm. Observe that every edge $(a, b) \in T(v)$ along with vertex v induces a triangle in G. Furthermore, since $G[V_{\text{good}}]$ is bipartite (and hence triangle free), any triangle $T \subseteq G$ must contain at least one bad vertex. Therefore, as the vertex v is good, every edge $e \in T(v)$ must contain at least one bad vertex. Finally, we observe that the edges in $T(v)$ are vertex disjoint, and there are at most ϵn bad vertices, which together implies that $t(v) = |T(v)| \leq \epsilon n$. ◀

▶ **Lemma 25.** *With probability at least $1 - e^{-O(\epsilon n)}$, for every vertex $v \in V_{\text{bad}}$, we have $t(v) \geq 2\epsilon n$.*

Proof. Let G' be the subgraph on G consisting of edges from E_0 (i.e,. the randomly sampled set of edges). Recall that $M = \{(a_i, b_i)\}_{i \in [m^*]} \subseteq G[V_{\text{good}}]$ is the fixed maximum matching in $G[V_{\text{good}}]$ of size m^*. Let $Z_i := \mathbb{1}\Big(\{a_i, b_i \in N_{G'}(v)\}\Big)$ be the indicator variable for the event that a_i, b_i are neighbors of v in the graph G'. Then,

$$\mathbb{E}_{G'}\left[\sum_{i \in [m^*]} Z_i\right] = \sum_{i \in [m^*]} \Pr_{G'}\left[\{a_i, b_i \in N_{G'}(v)\}\right] = m^* p^2 \geq 8\epsilon n \tag{7}$$

Furthermore, since the edges in M are vertex disjoint, the random variables $Z_1, \ldots, Z_{m^*}$ are independent and identical. Therefore using Chernoff bound we get

$$\Pr_{G'}\left[\sum_{i \in [m^*]} Z_i \leq 4\epsilon n\right] \leq \Pr_{G'}\left[\sum_{i \in [m^*]} Z_i \leq \frac{1}{2}\mathbb{E}\sum_{i \in [m^*]} Z_i\right] \leq e^{-O(\epsilon n)} \tag{8}$$

Let $M_v = \{(a_i, b_i) : i \in [m^*], Z_i = 1\}$ be the set of matching edges from M^* appearing in the neighborhood of v in the graph G'. Furthermore, let $\tilde{M}_v$ be a maximum matching in the subgraph $G_V := G[N_G(v)]$ induced on the neighborhood of v (which contains both E_0 and E_1

edges). Then, by definition we have $|\tilde{M}_v| \geq |M_v|$. On the other hand, by construction, the set $T(v)$ is a maximal matching in the induced subgraph G_v. Since a maximal matching is a 2-approximation to the maximum matching, it follows that $|T(v)| \geq |\tilde{M}_v|/2 \geq |M_v|/2 \geq 2\epsilon n$.

Therefore, for a fixed bad vertex $v \in V_{\text{bad}}$, with probability at least $1 - e^{-O(\epsilon n)}$, we have $t(v) \geq 2\epsilon n$. The claim now follows by taking a union bound over all vertices $v \in V_{\text{bad}}$. ◀

Therefore, combining Lemmas 24 and 25, we know that with probability at least $1-e^{-O(\epsilon n)}$, we have $t(v) \leq \epsilon n$ if and only if $v \in V_{\text{good}}$. Conditioned on this event, the set S must exactly be the set of bad vertices, in which case $G[V \setminus S] = G[V_{\text{good}}]$ is bipartite.

Case (ii): $m^* \leq (8\epsilon/p^2)n$. Since the size of maximum matching in $G[V_{\text{good}}]$ is at most $(8\epsilon/p^2)n$, and $G[V_{\text{good}}]$ is bipartite, by König's theorem (Theorem 2.1.1 [14]), it follows that the minimum vertex cover of $G[V_{\text{good}}]$ has size at most $(8\epsilon/p^2)n$. Then G has a vertex cover of size at most $(8\epsilon/p^2)n + \epsilon n \leq (10\epsilon/p^2)n$. Therefore, the greedy approximation algorithm for vertex cover returns a vertex cover S' of size at most $(20\epsilon/p^2)n$, and consequently, $V \setminus S'$ will be an independent set of size at least $(1 - (20\epsilon/p^2))n$.

Putting things together. In case (i), the algorithm throws away at most ϵn vertices and returns a 2-colorable graph, with probability at least $1 - e^{-O(\epsilon n)}$. In case (ii), the algorithm throws away at most $O(\epsilon/p^2)n$ vertices, and returns an indpendent set. Combining the two cases gives us the guarantees for Theorem 7.

E Maximal and Maximum Short Odd Cycle sets

▶ **Proposition 26.** *For any graph $G := (V, E)$, and parameter $\delta \in (0, 1)$ the following holds. Let $\mathcal{C}$ be a maximal set of vertex disjoint odd cycles in G of length at most $1/\delta$, and let $\tilde{\mathcal{C}}$ be a set of largest cardinality of vertex disjoint odd cycles in G of length at most $1/\delta$. Then $|\mathcal{C}| \geq \delta|\tilde{\mathcal{C}}|$.*

Proof. Since $\mathcal{C}$ is a maximal set of vertex disjoint odd cycles of length at most $1/\delta$, for every odd cycle $\tilde{C} \in \tilde{\mathcal{C}}$, there exists an odd cycle $C \in \mathcal{C}$ such that $C \cap \tilde{C} \neq \emptyset$ i.e,. $\tilde{C}$ is hit by C. Now we observe that (i) the cycles in $\mathcal{C}$ are vertex disjoint and (ii) each cycle $C \in \mathcal{C}$ has size at most $1/\delta$. Hence, it follows that any cycle $C \in \mathcal{C}$ hits at most $1/\delta$ cycles in $\tilde{\mathcal{C}}$. Since every cycle in $\tilde{\mathcal{C}}$ is hit by some cycle in $\mathcal{C}$, we must have $|\mathcal{C}| \geq \frac{|\tilde{\mathcal{C}}|}{1/\delta} = \delta|\tilde{\mathcal{C}}|$. ◀

F Identifying the set of Good Vertices is NP-hard

▶ **Fact 27.** *For all $k \in \mathbb{N}$, given a graph α-partially k-colorable graph $G = (V, E)$ it is NP-Hard to identify a set $V_{\text{good}} \subset V$ of size at least αn such that $G[V_{\text{good}}]$ is k-colorable*

Proof. For $\alpha = 1 - 1/2n$, this is exactly the k-Coloring problem which is NP-Hard [18]. ◀

Towards Optimal Moment Estimation in Streaming and Distributed Models

Rajesh Jayaram
Carnegie Mellon University, Pittsburgh, PA, USA
http://rajeshjayaram.com/
rkjayara@cs.cmu.edu

David P. Woodruff
Carnegie Mellon University, Pittsburgh, PA, USA
http://www.cs.cmu.edu/~dwoodruf/
dwoodruf@cs.cmu.edu

Abstract

One of the oldest problems in the data stream model is to approximate the p-th moment $\|\mathcal{X}\|_p^p = \sum_{i=1}^n \mathcal{X}_i^p$ of an underlying non-negative vector $\mathcal{X} \in \mathbb{R}^n$, which is presented as a sequence of poly(n) updates to its coordinates. Of particular interest is when $p \in (0, 2]$. Although a tight space bound of $\Theta(\epsilon^{-2} \log n)$ bits is known for this problem when both positive and negative updates are allowed, surprisingly there is still a gap in the space complexity of this problem when all updates are positive. Specifically, the upper bound is $O(\epsilon^{-2} \log n)$ bits, while the lower bound is only $\Omega(\epsilon^{-2} + \log n)$ bits. Recently, an upper bound of $\tilde{O}(\epsilon^{-2} + \log n)$ bits was obtained under the assumption that the updates arrive in a *random order*.

We show that for $p \in (0, 1]$, the random order assumption is not needed. Namely, we give an upper bound for worst-case streams of $\tilde{O}(\epsilon^{-2} + \log n)$ bits for estimating $\|\mathcal{X}\|_p^p$. Our techniques also give new upper bounds for estimating the empirical entropy in a stream. On the other hand, we show that for $p \in (1, 2]$, in the natural coordinator and blackboard distributed communication topologies, there is an $\tilde{O}(\epsilon^{-2})$ bit max-communication upper bound based on a randomized rounding scheme. Our protocols also give rise to protocols for heavy hitters and approximate matrix product. We generalize our results to arbitrary communication topologies G, obtaining an $\tilde{O}(\epsilon^2 \log d)$ max-communication upper bound, where d is the diameter of G. Interestingly, our upper bound rules out natural communication complexity-based approaches for proving an $\Omega(\epsilon^{-2} \log n)$ bit lower bound for $p \in (1, 2]$ for streaming algorithms. In particular, any such lower bound must come from a topology with large diameter.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Streaming, Sketching, Message Passing, Moment Estimation

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.29

Category APPROX

Related Version A full version of the paper is available at https://arxiv.org/abs/1907.05816.

Funding The authors thank the partial support by the National Science Foundation under Grant No. CCF-1815840.

1 Introduction

The streaming and distributed models of computation have become increasingly important for the analysis of massive datasets, where the sheer size of the input imposes stringent restrictions on the resources available to algorithms. Examples of such datasets include internet traffic logs, sensor networks, financial transaction data, database logs, and scientific data streams (such as huge experiments in particle physics, genomics, and astronomy). Given


© Rajesh Jayaram and David P. Woodruff;
licensed under Creative Commons License CC-BY



their prevalence, there is a large body of literature devoted to designing extremely efficient algorithms for analyzing streams and enormous datasets. We refer the reader to [4, 52] for surveys of these algorithms and their applications.

Formally, the data stream model studies the evolution of a vector $\mathcal{X} \in \mathbb{Z}^n$, called the frequency vector. Initially, $\mathcal{X}$ is initialized to be the zero-vector. The frequency vector then receives a stream of m coordinate-wise updates of the form $(i_t, \Delta_t) \in [n] \times \{-M, \dots, M\}$ for some $M > 0$ and time step $t \in [m]$. Each update (i_t, Δ_t) causes the change $\mathcal{X}_{i_t} \leftarrow \mathcal{X}_{i_t} + \Delta_t$. If we restrict that $\Delta_t \geq 0$ for all $t \in [m]$, this is known as the *insertion-only* model. If the updates $\Delta_t \in \{-M, \dots, M\}$ can be both positive and negative, then this is known as the *turnstile*-model. The p-th frequency moment of the frequency vector at the end of the stream, F_p, is defined as $F_p = \sum_{i=1}^{n} |\mathcal{X}_i|^p$. For simplicity (but not necessity), it is generally assumed that $m, M = \text{poly}(n)$.

The study of frequency moments in the streaming model was initiated by the seminal 1996 paper of Alon, Matias, and Szegedy [1]. Since then, nearly two decades of research have been devoted to understanding the space and time complexity of this problem. An incomplete list of works which study frequency moments in data streams includes [16, 36, 6, 58, 35, 45, 12, 44, 11, 15, 11, 7, 13]. For $p > 2$, it is known that polynomial in n (rather than logarithmic) space is required for F_p estimation [16, 36]. In the regime of $p \in (0, 2]$, the space complexity of F_p estimation in the turnstile model is now understood, with matching upper and lower bounds of $\Theta(\epsilon^{-2} \log(n))$ bits to obtain a $(1 \pm \epsilon)$ approximation of F_p. Here, for $\epsilon > 0$, a $(1 \pm \epsilon)$ approximation means an estimate $\tilde{F}_p$ such that $(1 - \epsilon)F_p \leq \tilde{F}_p \leq (1 + \epsilon)F_p$. For insertion only streams, however, the best known lower bound is $\Omega(\epsilon^{-2} + \log(n))$ [58]. Moreover, if the algorithm is given query access to an arbitrarily long string of random bits (known as the random oracle model), then the lower bound is only $\Omega(\epsilon^{-2})$. On the other hand, the best upper bound is to just run the turnstile $O(\epsilon^{-2} \log(n))$-space algorithm.

In this work, we make progress towards resolving this fundamental problem. For $p < 1$, we resolve the space complexity by giving an $\tilde{O}(\epsilon^{-2} + \log n)$[1]-bits of space upper bound. In the random oracle model, our upper bound is $\tilde{O}(\epsilon^{-2})$[2], which also matches the lower bound in this setting. Prior to this work, an $\tilde{O}(\epsilon^{-2} + \log(n))$ upper bound for F_p estimation was only known in the restricted *random-order* model, where it is assumed that the stream updates are in a uniformly random ordering [13]. Our techniques are based on novel analysis of the behavior of the p-stable random variables used in the $O(\epsilon^{-2} \log(n))$ upper bound of [35], and also give rise to a space optimal algorithm for entropy estimation.

We remark that F_p estimation in the range $p \in (0, 1)$ is useful for several reasons. Firstly, for p near 1, F_p estimation is often used as a subroutine for estimating the empirical entropy of a stream, which itself is useful for network anomaly detection ([47], also see [31] and the references therein). Moment estimation is also used in weighted sampling algorithms for data streams [50, 42, 38] (see [23] for a survey of such samplers and their applications). Here, the goal is to sample an index $i \in [n]$ with probability $|\mathcal{X}_i|^p / F_p$. These samplers can be used to find heavy-hitters in the stream, estimate cascaded norms [2, 50], and design representative histograms of $\mathcal{X}$ on which more complicated algorithms are run [28, 27, 55, 29, 33, 24]. Furthermore, moment estimation for fractional p, such as $p = .5$ and $p = .25$, has been shown to be useful for data mining [22].

---

[1] the $\tilde{O}$ here suppresses a single $(\log \log n + \log 1/\epsilon)$ factor, and in general we use $\tilde{O}$ and $\tilde{\Omega}$ to hide $\log \log n$ and $\log 1/\epsilon$ terms.

[2] This space complexity is measured *between updates*. To read and process the $\Theta(\log(n))$-bit identity of an update, the algorithm will use an additional $O(\log(n))$-bit working memory tape during an update. Note that all lower bounds only apply to the space complexity between updates, and allow arbitrary space to process updates.

For the range of $p \in (1, 2]$, we prove an $\tilde{O}(\epsilon^{-2})$-bits of max-communication upper bound in the distributed models most frequently used to prove *lower bounds* for streaming. This result rules out a large and very commonly used class of approaches for proving lower bounds against the space complexity of streaming algorithms for F_p estimation. Our approach is based on a randomized rounding scheme for p-stable sketches. We show that our rounding scheme can be additionally applied to design improved protocols for the distributed heavy hitters and approximate matrix product problems. We now introduce the model in which all the aforementioned results hold.

1.1 Multi-Party Communication

In this work, we study a more general model than streaming, known as the message passing multi-party communication model. All of our upper bounds apply to this model, and our streaming algorithms are just the result of special cases of our communication protocols. In the message passing model, there are m players, each positioned at a unique vertex in a graph $G = (V, E)$. The i-th player is given as input an integer vector $X_i \in \mathbb{Z}^n$. The goal of the players is to work together to jointly approximate some function $f : \mathbb{R}^n \to \mathbb{R}$ of the aggregate vector $\mathcal{X} = \sum_{i=1}^n X_i$, such as the p-th moment $f(\mathcal{X}) = F_p = \|\mathcal{X}\|_p^p = \sum_{i=1}^n |\mathcal{X}_i|^p$. In the message passing model, as opposed to the *broadcast* model of communication, the players are only allowed to communicate with each other over the edges of G. Thus player i can send a message to player j only if $(i, j) \in E$, and this message will only be received by player j (and no other). At the end of the protocol, it is assumed that at least one player holds the approximation to $f(\mathcal{X})$. The goal of multi-party communication is to solve the approximation problem using small total communication between all the players over the course of the execution. More specifically, the goal is to design protocols that use small *max-communication*, which is the total number of bits sent over any edge of G. Our protocols hold in an even more restricted setting, known as the *one-shot* setting, where each player is allowed to communicate exactly once over the course of the entire protocol.

We now observe that data streams can be modeled as a special case of one-shot multi-party communication. Here, the graph G in question is the line graph on m vertices. If the updates to the data stream vector are $(i_1, \Delta_1), \ldots, (i_m, \Delta_m)$, then the t-th player has input $X_t \in \mathbb{Z}^n$, where $(X_t)_{i_t} = \Delta_t$ and $(X_t)_j = 0$ for $j \neq i_t$. The aggregate vector $\mathcal{X} = \sum_{i=1}^m X_i$ is just the frequency vector at the end of the stream, and the space complexity of any algorithm is just the max-communication used over any edge of the corresponding communication protocol. Since we are primarily interested in insertion only streams, in this work we will consider the *non-negative data* model, where $X_i \in \{0, 1, \ldots, M\}^n$ for all input vectors X_i, for some $M > 0$ (as in streaming, we assume $M = \text{poly}(n, m)$ for simplicity). Note that an equivalent condition is that each $X_i \in \mathbb{R}^n_{\geq 0}$ such that the entries of X_i can be stored in $O(\log M)$-bits.

We are now ready to introduce our results for moment estimation in the message passing model. Let d be the *diameter* of the communication graph G. Our first result is a protocol for F_p estimation when $p \in (1, 2]$ which uses a max communication of $\tilde{O}(\epsilon^{-2} \log d)$ bits. Using similar techniques, we also obtain a (optimal for $d = \Theta(1)$) bound of $\tilde{O}(\epsilon^{-2} \log n \log d)$ for the heavy hitters problem, which is to find the coordinates of $\mathcal{X}$ which contribute at least an ϵ fraction of the total $\sqrt{F_2} = \|\mathcal{X}\|_2$ of $\mathcal{X}$. For $p \in (0, 1)$, we give an $\tilde{O}(\epsilon^{-2})$ upper bound for F_p estimation. Notice that this is independent of the graph topology, and thus holds for the line graph, where we derive our $\tilde{O}(\epsilon^{-2})$ upper bound for F_p estimation in the random oracle streaming model. We then show how the streaming algorithm can be derandomized to not require a random oracle, now using an optimal $\tilde{O}(\epsilon^{-2} + \log(n))$-bits of space. Our techniques also result in an $\tilde{O}(\epsilon^{-2})$ upper bound for additively approximating the empirical entropy of the vector $\mathcal{X}$.

Our results for $p \in (1, 2]$ have interesting implications for any attempts to prove *lower-bounds* for streaming algorithms that estimate F_p, which we now describe. The link between streaming and communication complexity is perhaps one of the most fruitful sources of space lower bounds for algorithms in computer science. Namely, nearly all lower bounds for the space complexity of randomized streaming algorithms are derived via reductions from communication problems. For an incomplete list of such reductions, see [58, 61, 45, 42, 46, 10, 16, 57, 48, 49, 40] and the references therein. Now nearly all such lower bounds (and all of the ones that were just cited) hold in either the 2-party setting (G has 2 vertices), the coordinator model, or the black-board model. In the coordinator model there are m players, each with a single edge to a central coordinator (i.e., G is a star graph on $m + 1$ vertices). Note that the diameter d of the coordinator graph is 2. In the multi-player black-board model, every message that is sent is written to a shared blackboard that can be read by all players. Observe that any one-way protocol for the coordinator model immediately results in a protocol with the same communication for the blackboard model. Namely, each player simply writes what it would have sent to the coordinator on the blackboard, and at the end of the protocol the blackboard contains all the information that the coordinator would have had. For these three settings, our protocol gives an $\tilde{O}(\epsilon^{-2})$ max-communication upper bound for F_p estimation, $p \in (1, 2]$. This completely rules out the approach for proving lower bounds against F_p estimation in a stream via any of these three techniques. In particular, it appears that any lower bound for F_p estimation via communication complexity in this regime of p will need to use a graph with $\Omega(n)$ diameter, such as the line graph, without a black-board.

The coordinator and black-board models have also been studied in many other settings than for proving lower bounds against streaming. For instance, in the *Distributed Functional Monitoring* literature [25, 63, 60, 34, 56, 37], each player is receiving a continuous stream of updates to their inputs X_i, and the coordinator must continuously update its approximation to $f(\mathcal{X})$. The black-board model is also considered frequently for designing communication upper bounds, such as those for set disjointness [6, 16, 30]. Finally, there is substantial literature which considers numerical linear algebra and clustering problems in the coordinator model [61, 20, 5, 62]. Thus, our upper bounds can be seen as a new and useful contribution to these bodies of literature as well.

1.2 Our Contributions

As noted, the upper bounds in this paper all hold in the general multi-party message passing model, over an arbitrary topology G. Our algorithms also have the additional property that they are *one-shot*, meaning that each player is allowed to communicate exactly once. Our protocols pre-specify a central vertex $\mathcal{C} \in V$ of G. Specifically, $\mathcal{C}$ will be a *center* of G, which is a vertex with minimal max-distance to any other vertex. Our protocols then proceed in d rounds, where d is the diameter of G. Upon termination of the protocols, the central vertex $\mathcal{C}$ will hold the estimate of the protocol. We note that $\mathcal{C}$ can be replaced by any other vertex v, and d will then be replaced by the max distance of any other vertex to v. A summary of our results is given in Table 1.

We first formally state our general result for F_p estimation, $1 < p \leq 2$. Note that, while we state all our results for constant probability of success, by repeating $\log(1/\delta)$ times and taking the median of the estimates, this is boosted to $1 - \delta$ in the standard way.

▶ **Theorem 12.** *For $p \in (1, 2]$, there is a protocol for $(1 \pm \epsilon)$ approximating F_p which succeeds with probability $3/4$ in the message passing model. The protocol uses a max communication of $O(\frac{1}{\epsilon^2}(\log\log n + \log d + \log 1/\epsilon))$ bits, where d is the diameter of G.*

Table 1 For the communication problems above, the bounds are for the max-communication (in bits) across any edge. For the streaming problems, the bounds are for the space requirements of the algorithm. Here, d is the diameter of the communication network G. For all problems except point estimation, there is a matching $\Omega(\epsilon^{-2})$ lower bound. The problem of point estimation itself has a matching $\Omega(\epsilon^{-2}\log n)$ lower bound for graphs with constant d.

Problem	Prior best upper bound	Upper Bound (this work)	Notes
F_p, $1 < p \leq 2$	$O(\epsilon^{-2}\log(n))$ [45]	$\tilde{O}(\epsilon^{-2}\log(d))$	
F_p, $p < 1$	$O(\epsilon^{-2}\log(n))$[45]	$\tilde{O}(\epsilon^{-2})$	
F_p Streaming, $p < 1$	$O(\epsilon^{-2}\log(n))$[45]	$\tilde{O}(\epsilon^{-2})$	
Entropy	–	$\tilde{O}(\epsilon^{-2})$	
Entropy Streaming	$O(\epsilon^{-2}\log^2(n))$ [21]	$\tilde{O}(\epsilon^{-2})$	random oracle
Point Estimation	$O(\epsilon^{-2}\log^2(n))$ [18]	$\tilde{O}(\epsilon^{-2}\log(d)\log(n))$	
Approx Matrix Prod.	–	$\tilde{O}(1)$	per coordinate of sketch

For graphs with constant diameter, such as the coordinator model, our max communication bound of $\tilde{O}(\epsilon^{-2})$ matches the $\Omega(\epsilon^{-2})$ lower bound [58, 17], which follows from a 2-player reduction from the Gap-Hamming Distance problem. For $p = 2$, our *total communication* in the coordinator model matches the $\Omega(m^{p-1}/\epsilon^2)$ total communication lower bound (up to $\log\log(n)$ and $\log(1/\epsilon)$ terms) for non-one shot protocols [60]. For one shot protocols, we remark that there is an $\Omega(m/\epsilon^2)$ total communication lower bound for any $p \in (0, 2] \setminus \{1\}$ (see Appendix A). As discussed previously, our result also has strong implications for streaming algorithms, demonstrating that no $\Omega(\epsilon^{-2}\log n)$ lower bound for F_p estimation, $p \in (1, 2]$, can be derived via the common settings of 2-party, coordinator, or blackboard communication complexity.

Our main technique used to obtain Theorem 12 is a new randomized rounding scheme for p-stable sketches. We next show that this randomized rounding protocol can be applied to give improved communication upper bounds for the *point-estimation* problem. Here, the goal is to output a vector $\tilde{X} \in \mathbb{R}^n$ that approximates $\mathcal{X}$ well coordinate-wise. The result is formally given below in Theorem 14.

▶ **Theorem 14.** *Consider a message passing topology $G = (V, E)$ with diameter d, where the i-th player is given as input $X^i \in \mathbb{Z}^n_{\geq 0}$ and $\mathcal{X} = \sum_{i=1}^m X^i$. Then there is a communication protocol which outputs an estimate $\tilde{\mathcal{X}} \in \mathbb{R}^n$ of $\mathcal{X}$ such that $\|\tilde{\mathcal{X}} - \mathcal{X}\|_\infty \leq \epsilon\|\mathcal{X}_{tail(\epsilon^{-2})}\|_2$ with probability $1 - 1/n^c$ for any constant $c \geq 1$. Here $\mathcal{X}_{tail(\epsilon^{-2})}$ is $\mathcal{X}$ with the ϵ^{-2} largest (in absolute value) coordinates set equal to 0. The protocol uses a max communication of $O(\frac{1}{\epsilon^2}\log(n)(\log\log n + \log d + \log 1/\epsilon))$.*

For graphs with small diameter, our protocols demonstrate an improvement over the previously best known sketching algorithms, which use space $O(\epsilon^{-2}\log^2(n))$ to solve the point estimation problem [18]. Note that there is an $\Omega(\epsilon^{-2}\log n)$-max communication lower bound for the problem. This follows from the fact that point-estimation also solves the L_2 *heavy-hitters* problem. Here the goal is to output a set $S \subset [n]$ of size at most $|S| = O(\epsilon^{-2})$ which contains all $i \in [n]$ with $|\mathcal{X}_i| \geq \epsilon\|\mathcal{X}\|_2$ (such coordinates are called heavy hitters). The lower bound for heavy hitters is simply the result of the space required to store the $\log(n)$-bit identities of all possible ϵ^{-2} heavy hitters. Note that for the heavy hitters problem alone, there is an optimal streaming $O(\epsilon^{-2}\log(n))$-bits of space upper bound called BPTree [9]. However, BPTree cannot be used in the general distributed setting, since it crucially relies on the sequential natural of a stream.

Next, we demonstrate that F_p estimation for $p < 1$ is in fact possible with max communication independent of the graph topology. After derandomizing our protocol, this results in a optimal streaming algorithm for F_p estimation, $p < 1$, which closes a long line of research on the problem for this particular range of p [58, 35, 45, 44, 15, 13].

▶ **Theorem 21.** *For $p \in (0, 1)$, there is a protocol for F_p estimation in the message passing model which succeeds with probability $2/3$ and has max-communication of $O(\frac{1}{\epsilon^2}(\log\log n + \log 1/\epsilon))$.*

▶ **Theorem 22.** *There is a streaming algorithm for F_p estimation, $p \in (0, 1)$, which outputs a value $\tilde{R}$ such that with probability at least $2/3$, we have that $|\tilde{R} - \|X\|_p| \leq \epsilon\|X\|_p$. The algorithm uses $O((\frac{1}{\epsilon^2}(\log\log n + \log 1/\epsilon) + \frac{\log 1/\epsilon}{\log\log 1/\epsilon}\log n)$-bits of space. In the random oracle model, the space is $O(\frac{1}{\epsilon^2}(\log\log n + \log 1/\epsilon))$.*

The above bound matches the $\Omega(\epsilon^{-2})$ max communication lower bound of [58] in the shared randomness model, which comes from 2-party communication complexity. Moreover, our streaming algorithm matches the $\Omega(\log n)$ lower bound for streaming when a random oracle is not allowed. As an application of our protocol for F_p estimation, $p < 1$, we demonstrate a communication optimal protocol for additive approximation of the empirical *Shannon entropy* $H(\mathcal{X})$ of the aggregate vector $\mathcal{X}$. Here, $H = H(\mathcal{X})$ is defined by $H = \sum_{i=1}^{n} p_i \log(1/p_i)$ where $p_i = |\mathcal{X}_i|/\|\mathcal{X}\|_1$ for $i \in [n]$. The goal of our protocols is to produce an estimate $\tilde{H} \in \mathbb{R}$ of H such that $|\tilde{H} - H| \leq \epsilon$. Our result is as follows.

▶ **Theorem 26.** *There is a multi-party communication protocol in the message passing model that outputs a ϵ-additive error of the Shannon entropy H. The protocol uses a max-communication of $O(\frac{1}{\epsilon^2}(\log\log(n) + \log(1/\epsilon))$-bits.*

Note that for a *multiplicative* approximation of the Shannon entropy, there is a $\tilde{\Omega}(\epsilon^{-2})$ lower bound [14]. For additive estimation, [43] gives a $\Omega(\epsilon^{-2}\log(n))$ lower bound in the turnstile model. Using a similar reduction, we prove a matching $\Omega(\epsilon^{-2})$ lower bound for additive ϵ approximation in the insertion only model (see Appendix B for the proof). Furthermore, our protocol directly results in an $\tilde{O}(\epsilon^{-2})$-bits of space, insertion only *streaming* algorithm for entropy estimation in the random oracle model. Here, the random oracle model means that the algorithm is given query access to an arbitrarily long string of random bits. We note that many lower bounds in communication complexity (and all of the bounds discussed in this paper except for the $\Omega(\log n)$ term in the lower bound for F_p estimation) also apply to the random oracle model. Previously, the best known algorithm for the insertion only random oracle model used $O(\epsilon^{-2}\log(n))$-bits [47, 21], whereas the best known algorithm for the non-random oracle model uses $O(\epsilon^{-2}\log^2(n))$-bits (the extra factor of $\log(n)$ comes from a standard application of Nisan's pseudo-random generator [53]).

▶ **Theorem 27.** *There is a streaming algorithm for ϵ-additive approximation of the empirical Shannon entropy of an insertion only stream in the random oracle model, which succeeds with probability $3/4$. The space required by the algorithm is $O(\frac{1}{\epsilon^2}(\log\log(n) + \log(1/\epsilon))$ bits.*

Finally, we show how our techniques can be applied to the important numerical linear algebraic primitive of *approximate matrix product*, which we now define.

▶ **Definition 1.** *The multi-party approximate matrix product problem is defined as follows. Instead of vector valued inputs, each player is given $X_i \in \{0, 1, \ldots, M\}^{n\times t_1}$ and $Y_i \in \{0, 1, \ldots, M\}^{n\times t_2}$, where $\mathcal{X} = \sum_i X_i$ and $\mathcal{Y} = \sum_i Y_i$. Here, it is generally assumed that $n >> t_1, t_2$ (but not required). The players must work together to jointly compute a matrix $R \in \mathbb{R}^{t_1\times t_2}$ such that $\|R - \mathcal{X}^T\mathcal{Y}\|_F \leq \epsilon\|\mathcal{X}\|_F\|\mathcal{Y}\|_F$, where for a matrix $A \in \mathbb{R}^{n\times m}$, $\|A\|_F = (\sum_{i=1}^{n}\sum_{j=1}^{m} A_{i,j}^2)^{1/2}$ is the Frobenius norm of A.*

▶ **Theorem 29.** *There is a protocol which outputs, at the central vertex $\mathcal{C}$, a matrix $R \in \mathbb{R}^{t_1 \times t_2}$ which solves the approximate communication protocol with probability $3/4$ [3]. The max communication required by the protocol is $O\left(\epsilon^{-2}(t_1+t_2)(\log\log n + \log 1/\epsilon + \log d)\right)$, where d is the diameter of the communication topology G.*

We remark that an upper bound of $O\left(\epsilon^{-2}(t_1+t_2)\log n\right)$ was already well-known from sketching theory [59], and our main improvement is removing the $\log(n)$ factor for small diameter graphs, such as the coordinator model where distributed numerical linear algebra is usually considered.

1.3 Other Related Work

As mentioned, a closely related line of work is in the *distributed functional monitoring model*. Here, there are m machines connected to a central coordinator (the coordinator topology). Each machine then receives a stream of updates, and the coordinator must maintain at all time steps an approximation of some function, such as a moment estimation or a uniform sample, of the union of all streams. We note that there are two slightly different models here. One model is where the items (coordinates) being updated in the separate streams are considered disjoint, and each time an insertion is seen it is to a unique item. This model is considered especially for the problem of maintaining a uniform sample of the items in the streams [25, 34, 56, 37]. The other model, which is more related to ours, is where each player is receiving a stream of updates to a *shared* overall data vector $\mathcal{X} \in \mathbb{R}^n$. This can be seen as a distributed streaming setting, where the updates to a centralized stream are split over m servers, and is considered in [60, 25, 3]. For the restricted setting of *one-way* algorithms, which only transmit messages from the sites to the coordinators, any such algorithm can be made into a one-shot protocol for the multi-party message passing model. Here, each machine just simulates a stream on their fixed input vectors X_i, and sends all the messages that would have been sent by the functional monitoring protocol.

Perhaps the most directly related result to our upper bound for for F_p estimation, $p \in (1,2]$, is in the distributed functional monitoring model, where Woodruff and Zhang [60] show a $O(m^{p-1}\text{poly}(\log(n), 1/\epsilon) + m\epsilon^{-1}\log(n)\log(\log(n)/\epsilon))$[4] *total communication* upper bound. We remark here, however, that the result of [60] is incomparable to ours for several reasons. Firstly, their bounds are only for total communication, whereas their max communication can be substantially larger than $O(1/\epsilon^2)$. Secondly, while it is claimed in the introduction that the protocols are one way (i.e., only the players speak to the coordinator, and not vice versa), this is for their threshold problem and not for F_p estimation[5]. As remarked before, there is an $\Omega(m/\epsilon^2)$ total communication lower bound for one-way protocols, which demonstrates that their complexity could not hold in our setting (we sketch a proof of this in Appendix A).

[3] We remark that there are standard techniques to boost the probability of the matrix sketching results to $1-\delta$, using a blow-up of $\log(\delta)$ in the communication. See e.g. Section 2.3 of [59]

[4] We remark that the $\text{poly}(\log(n), 1/\epsilon)$ terms here are rather large, and not specified in the analysis of [60].

[5] The reason for this is as follows. Their algorithm reduces F_p estimation to the threshold problem, where for a threshold τ, the coordinator outputs 1 when the F_p first exceeds $\tau(1+\epsilon)$, and outputs 0 whenever the F_p is below $\tau(1-\epsilon)$. To solve F_p estimation, one then runs this threshold procedure for the $\log(mMn)/\epsilon$ thresholds $\tau = (1+\epsilon), (1+\epsilon)^2, \ldots, (mMn)^2$ in parallel. However, the analysis from [60] only demonstrates a total communication of $O(k^{1-p}\text{poly}(\log(n), \epsilon^{-1}))$ for the time steps *before* the threshold τ is reached. Once the threshold is reached, the communication would increase significantly, thus the coordinator must inform all players when a threshold τ is reached so that they stop sending messages for τ, violating the one-way property. This step also requires an additive k messages for each of the $O(\epsilon^{-1}\log(n))$ thresholds, which results in the $O(m\epsilon^{-1}\log(n)\log(\log(n)\epsilon)))$ term.

The message passing model itself has been the subject of significant research interest over the past two decades. The majority of this work is concerned with *exact* computation of Boolean functions of the inputs. Perhaps the canonical multi-party problem, and one which has strong applications to streaming, is set disjointness, where each player has a subset $S_i \subset [n]$ and the players want to know if $\cap_{i=1}^{m} S_i$ is empty. Bar-Yossef et al. [6] demonstrated strong bounds for this problem in the black-board model. This lower bound resulted in improved (polynomially sized) lower bounds for streaming F_p estimation for $p > 2$. These results for disjointness have since been generalized and improved using new techniques [16, 30, 41, 8]. Finally, we remark that while most results in the multi-party message passing model are not topology dependent, Chattopadhyay, Radhakrishnan, and Rudra have demonstrated that tighter topology-dependent lower bounds are indeed possible in the message passing model [19].

2 Preliminaries

Let f be a function $f : \mathbb{R}^n \to \mathbb{R}$. Let $G = (V, E)$ be a connected undirected graph with m vertices, i.e. $V = \{1, \ldots, m\}$. In the message passing model on the graph topology G, there are m players, each placed at a unique vertex of G, with unbounded computational power. Player i is given as input only a vector $X_i \in \mathbb{Z}^n$, which is known as the Number in Hand (NIH) model of communication. Let $\mathcal{X} = \sum_{i=1}^{n} X_i$ be the aggregate vector of the players inputs. The goal of the players is to jointly compute or approximate the function $f(\mathcal{X})$ by carrying out some previously unanimously agreed upon communication protocol. It is assumed that the graph topology of G is known to all players.

In this paper, we are concerned with the non-negative input model. Namely, the inputs X_i satisfy $X_i \in \{0, 1, \ldots, M\}^n$ for all players i. Note an equivalent assumption to is that $(X_i)_j \geq 0$ for all i, and that the $(X_i)_j$'s can be specified in $O(\log(M))$ bits.

▶ Remark 2. For ease of presentation, we assume that $m, M = O(n^c)$ for some constant c. This allows us to simplify complexity bounds and write $\log(nmM) = O(\log n)$. This is a common assumption in the streaming literature, where m corresponds to the length of the stream. We remark, however, that all our results hold for general m, n, M, by replacing each occurrence of n in the communication complexity with (mnM).

During execution of the protocol, a player $i \in V$ is only allowed to send a message to a player j if $(i, j) \in E$. Thus, players may only communicate directly with their neighbors in the graph G. In contrast to the *broadcast* and *blackboard* models of communication, in the message passing model the message sent by player i to player j is only received by player j, and no other player. Upon termination of the protocol, at least one player must hold an approximation of the value $f(\mathcal{X})$. For the protocols considered in this paper, this player will be fixed and specified by the protocol beforehand. We use $\mathcal{C} \in V$ to denote the distinguished player specified by the protocol to store the approximation at the end of the execution.

Every such communication protocol in this model can be divided into rounds, where on the j-th round some subset $S_j \subseteq V$ of the players simultaneously send a message across one of their edges. Although it is not a restriction in the message passing model, our protocols satisfy the additional property that each player communicates *exactly once*, across one of its edges, and that each player will receive messages from its neighbors in exactly one round. Specifically, for each player i, there will be exactly one round j where some subset of its neighbors send player i a message, and then player i will send a single message in round $j + 1$, and never again communicate. Such protocols are called *one-shot* protocols.

The *total communication* cost of a protocol is the total number of bits sent in all the messages during its execution. The *max-communication* of a protocol is the maximum number of bits sent across any edge over the execution of the protocol. Communication protocols can be either deterministic or randomized. In this paper we consider the standard *public-coin* model of communication, where each player is given shared access to an arbitrarily long string of random bits. This allows players to jointly utilize the same source of randomness without having to communicate it.

Our protocols for F_p estimation will utilize the p-stable distribution, D_p, which we will now introduce. For $p = 2$, the distribution D_2 is the just standard Gaussian distribution. Note for $p < 2$, the distributions have heavy tails – they decay like x^{-p}. Thus, for $p < 2$, the variance is infinite, and for $p \leq 1$, the expectation is undefined.

▶ **Definition 3.** *For $0 < p \leq 2$, there exists a probability distribution D_p called the p-stable distribution. If $Z \sim D_p$, $p < 2$, then the characteristic function of D_p is given by $\mathbb{E}[e^{itZ}] = e^{-|t|^p}$. For $p = 2$, D_2 is the standard Gaussian distribution. Moreover, for any n, and any $x \in \mathbb{R}^n$, if $Z_1, \ldots, Z_n \sim D_p$ are independent, then $\sum_{i=1}^n Z_i x_i \sim \|x\|_p Z$, where $Z \sim D_p$, and $\sim$ means distributed identically to.*

Standard methods for generating p-stable random variables are discussed in [54]. Note that all protocols in this paper will generate these variables only to precision $1/\text{poly}(n)$. For a distribution D_p, we write D_p^n to denote the product distribution of D_p. Thus $Z \sim D_p^n$ means $Z \in \mathbb{R}^n$ and $Z_1, \ldots, Z_n$ are drawn i.i.d. from D_p. For reals $a, b \in \mathbb{R}$, we write $a = (1 \pm \epsilon)b$ to denote the containment $a \in [(1-\epsilon)b, (1+\epsilon)b]$. For an integer $t \geq 0$, we write $[t]$ to denote the set $\{1, 2, \ldots, t\}$.

3 Message Passing F_p Estimation, $p > 1$

In this section, we provide our algorithm for F_p estimation, $1 \leq p \leq 2$, in the message passing model. We begin by specifying the distinguished vertex $\mathcal{C} \in V$ which will hold and output the F_p approximation at the end of the protocol. For a vertex $v \in G$, define its eccentricity $\texttt{ecc}(v) = \max_{u \in V} d(v, u)$, where $d(v, u)$ is the graph distance between v, u. We then set $\mathcal{C} \in V$ to be any vertex with minimal eccentricity. Such a vertex is known as a center of G. We now fix a shortest path spanning tree T for G, rooted at the distinguished player $\mathcal{C}$. The spanning tree T has the property that the path between $\mathcal{C}$ and any vertex $v \in V$ in the tree T is also a shortest path between $\mathcal{C}$ and v in G. Thus the distance between $\mathcal{C}$ and any vertex $v \in V$ is the same in T as it is in G. The fact that the depth of T is at most d, where d is the diameter of G, now follows naturally. Such a shortest path spanning tree T can be easily obtained via a breath first search. First, we will need a technical Lemma about the behavior of p-stables. To prove it, we first use the following fact about the tails of p stables, which can be found in [54].

▶ **Proposition 4.** *If $Z \sim D_p$ for $0 < p < 2$, then $\Pr[|Z| \geq \lambda] \leq O(\frac{1}{\lambda^p})$.*

Also, we use the straightforward fact that $\|X_i\|_p^p \leq \|\sum_{i=1}^m X_i\|_p^p$ for non-negative vectors X_i and $p \geq 1$.

▶ **Fact 5.** *If $X_1, \ldots, X_m \in \mathbb{R}^n$ are entry-wise non-negative vectors and $1 \leq p \leq 2$, then $\sum_{i=1}^m \|X_i\|_p^p \leq \|\sum_{i=1}^m X_i\|_p^p$.*

▶ **Lemma 6.** *Fix $1 \le p \le q \le 2$, and let $Z = (Z_1, Z_2, \ldots, Z_n) \sim D_p^m$. Suppose $X_1, \ldots, X_m \in \mathbb{R}^n$ are non-negative vectors, with $\mathcal{X} = \sum_j X_j$. Then for any $\lambda \ge 1$, if either $q - p \ge c > 0$ for some constant c independent of m, or if $p = 2$, we have*

$$\Pr\left[\sum_{j=1}^{m} |\langle Z, X_j\rangle|^q \ge C\lambda^q \|\mathcal{X}\|_p^q\right] \le \frac{1}{\lambda^p}$$

Otherwise, we have $\Pr[\sum_{j=1}^m |\langle Z, X_j\rangle|^q \ge C\log(\lambda m)\lambda^q \|\mathcal{X}\|_p^q] \le \frac{1}{\lambda^p}$, where C is some constant (depending only on c in the first case).

▶ **Corollary 7.** *Suppose $Z = (Z_1, \ldots, Z_m)$ where the Z_i's are uniform over $\{1, -1\}$ and pairwise independent, and let $X_1, \ldots, X_m$ be non-negative vectors with $\mathcal{X} = \sum_j X_j$. Then for any $\lambda \ge 1$, we have $\Pr[\sum_{j=1}^m |\langle Z, X_j\rangle|^2 \ge \lambda \|\mathcal{X}\|_2^2] \le \frac{1}{\lambda}$*

▶ **Corollary 8.** *Let $Z = (Z_1, Z_2, \ldots, Z_n) \sim D_2^m$ be i.i.d. Gaussian. Suppose $X_1, \ldots, X_m \in \mathbb{R}^n$ are non-negative vectors, with $\mathcal{X} = \sum_j X_j$. Then for any $\lambda \ge c\log(m)$ for some sufficiently large constant c, we have $\Pr[\sum_{j=1}^m |\langle Z, X_j\rangle| \ge \lambda \|\mathcal{X}\|_2^2] \le \exp(-C\lambda)$, where C is some universal constant.*

3.1 Randomized Rounding of Sketches

We now introduce our randomized rounding protocol. Consider non-negative integral vectors $X_1, X_2, \ldots, X_m \in \mathbb{Z}_{\ge 0}^n$, with $\mathcal{X} = \sum_{i=1}^n X_i$. Fix a message passing topology $G = (V, E)$, where each player $i \in V$ is given as input X_i. Fix any vertex $\mathcal{C}$ that is a center of G, and let T be a shortest path spanning tree of G rooted at $\mathcal{C}$ as described at the beginning of the section. Let d be the depth of T. The players use shared randomness to choose a random vector $Z \in \mathbb{R}^n$, and their goal is to approximately compute $\langle Z, \mathcal{X}\rangle = \langle Z, \sum_{i=1}^m X_i\rangle$. The goal of this section is to develop a d-round randomized rounding protocol, so that at the end of the protocol the approximation to $\langle Z, \mathcal{X}\rangle$ is stored at the vertex $\mathcal{C}$.

We begin by introducing the rounding primitive which we use in the protocol. Fix $\epsilon > 0$, and let $\gamma = (\epsilon\delta/\log(nm))^C$, for a sufficiently large constant $C > 1$. For any real value $r \in \mathbb{R}$, let $i_r \in \mathbb{Z}$ and $\alpha_i \in \{1, -1\}$ be such that $(1+\gamma)^{i_r} \le \alpha_i r \le (1+\gamma)^{i_r+1}$. Now fix p_r such that: $\alpha_i r = p_r(1+\gamma)^{i_r+1} + (1-p_r)(1+\gamma)^{i_r}$. We then define the rounding random variable $\Gamma(r)$ by

$$\Gamma(r) = \begin{cases} 0 & \text{if } r = 0 \\ \alpha_i(1+\gamma)^{i_r+1} & \text{with probability } p_r \\ \alpha_i(1+\gamma)^{i_r} & \text{with probability } 1 - p_r \end{cases}$$

The following proposition is clear from the construction of p_r and the fact that the error is deterministically bounded by $\gamma|r|$.

▶ **Proposition 9.** *For any $r \in R$, We have $\mathbb{E}[\Gamma(r)] = r$ and $\boldsymbol{Var}[\Gamma(r)] \le r^2\gamma^2$*

We partition T into d layers, so that all nodes at distance $d - t$ from $\mathcal{C}$ in T are put in layer t. Define $L_t \subset [n]$ to be the set of players at layer t in the tree. For any vertex $u \in G$, let T_u be the subtree of T rooted at u (including the vertex u). For any player i, let $C_i \subset [n]$ be the set of children of i in the tree T. The procedure for all players $j \in V$ is then given as Algorithm 1.

Algorithm 1 Recursive Randomized Rounding.

Procedure for node j in layer i:

1. Choose random vector $Z \in \mathbb{R}^n$ using shared randomness.
2. Receive rounded sketches $r_{j_1}, r_{j_2}, \ldots, r_{j_{t_j}} \in \mathbb{R}$ from the t_j children of node j in the prior layer (if any such children exist).
3. Compute $x_j = \langle X_j, Z \rangle + r_{j_1} + r_{j_2} + \cdots + r_{j_t} \in \mathbb{R}$.
4. Compute $r_j = \Gamma(x_j)$. If player $j \neq \mathcal{C}$, then send r_j it to the parent node of j in T. If $j = \mathcal{C}$, then output r_j as the approximation to $\langle Z, \mathcal{X} \rangle$.

For each player i in layer 0, they take their input X_i, and compute $\langle Z, X_i \rangle$. They then round their values as $r_i = \Gamma(\langle Z, X_i \rangle)$, where the randomness used for the rounding function Γ is drawn independently for each call to Γ. Then player i sends r_i to their parent in T. In general, consider any player i at depth $j > 0$ of T. At the end of the j-th round, player i will receive a rounded value r_ℓ for every child vertex $\ell \in C_i$. They then compute $x_i = \langle Z, X_i \rangle + \sum_{\ell \in C_i} r_\ell$, and $r_i = \Gamma(x_i)$, and send r_i to their parent in T. This continues until, on round d, the center vertex $\mathcal{C}$ receives r_ℓ for all children $\ell \in C_{\mathcal{C}}$. The center $\mathcal{C}$ then outputs $r_{\mathcal{C}} = \langle Z, X_{\mathcal{C}} \rangle + \sum_{\ell \in C_{\mathcal{C}}} r_\ell$ as the approximation.

For any player i, let $Q_i = \sum_{u \in T_i} X_u$, and $y_i = \langle Z, Q_i \rangle$. Then define the error e_i at player i as $e_i = y_i - r_i$. We first prove a proposition that states the expectation of the error e_i for any player i is zero, and then the main lemma which bounds the variance of e_i. The error bound of the protocol at $\mathcal{C}$ then results from an application of Chebyshev's inequality.

▶ **Proposition 10.** *For any player i, we have $\mathbb{E}[e_i] = 0$. Moreover, for any players i, j such that $i \notin T_j$ and $j \notin T_i$, the variables e_i and e_j are statistically independent.*

▶ **Lemma 11.** *Fix $p \in [1, 2]$, and let $Z = (Z_1, Z_2, \ldots, Z_n) \sim D_p^n$. Then the above procedure when run on $\gamma = (\epsilon\delta/(d \log(nm)))^C$ for a sufficiently large constant C, produces an estimate $r_{\mathcal{C}}$ of $\langle Z, \mathcal{X} \rangle$, held at the center vertex $\mathcal{C}$, such that $\mathbb{E}[r_{\mathcal{C}}] = \langle Z, \mathcal{X} \rangle$. Moreover, over the randomness used to draw Z, with probability $1 - \delta$ for $p < 2$, and with probability $1 - e^{-1/\delta}$ for Gaussian Z, we have $\mathbb{E}[(r_{\mathcal{C}} - \langle Z, \mathcal{X} \rangle)^2] \leq (\epsilon/\delta)^2 \|\mathcal{X}\|_p$. Thus, with probability at least $1 - O(\delta)$, we have $|r_{\mathcal{C}} - \langle Z, \mathcal{X} \rangle| \leq \epsilon \|\mathcal{X}\|_p$. Moreover, if $Z = (Z_1, Z_2, \ldots, Z_n) \in \mathbb{R}^n$ where each $Z_i \in \{1, -1\}$ is a 4-wise independent Rademacher variable, then the above bound holds with $p = 2$ (and with probability $1 - \delta$).*

▶ **Theorem 12.** *For $p \in (1, 2]$, there is a protocol for F_p estimation which succeeds with probability $3/4$ in the message passing model, which uses a total of $O(\frac{m}{\epsilon^2}(\log(\log(n)) + \log(d) + \log(1/\epsilon)))$ communication, and a max communication of $O(\frac{1}{\epsilon^2}(\log(\log(n)) + \log(d) + \log(1/\epsilon)))$, where d is the diameter of the communication network.*

3.2 Heavy Hitters and Point Estimation

In this section, we show how our randomized rounding protocol can be used to solve the L_2 heavy hitters problem. For a vector $\mathcal{X} \in \mathbb{R}^n$, let $\mathcal{X}_{\text{tail}(k)}$ be $\mathcal{X}$ with the k largest (in absolute value) entries set equal to 0. Formally, given a vector $\mathcal{X} \in \mathbb{R}^n$, the heavy hitters problem is to output a set of coordinates $H \subset [n]$ of size at most $|H| = O(\epsilon^{-2})$ that contains all $i \in [n]$ with $|\mathcal{X}_i| \geq \epsilon \|\mathcal{X}_{tail(1/\epsilon^2)}\|_2$. Our protocols solve the strictly harder problem of *point-estimation*. The point estimation problem is to output a $\tilde{\mathcal{X}} \in \mathbb{R}^n$ such that $\|\tilde{\mathcal{X}} - \mathcal{X}\|_\infty \leq \epsilon \|\mathcal{X}_{tail(1/\epsilon^2)}\|_2$. Our protocol uses the well-known *count-sketch* matrix S [18], which we now introduce.

▶ **Definition 13.** *Given a precision parameter ϵ and an input vector $\mathcal{X} \in \mathbb{R}^n$, count-sketch stores a table $A \in \mathbb{R}^{\ell \times 6/\epsilon^2}$, where $\ell = \Theta(\log(n))$. Count-sketch first selects pairwise independent hash functions $h_j : [n] \to [6/\epsilon^2]$ and 4-wise independent $g_j : [n] \to \{1, -1\}$, for $j = 1, 2, \ldots, \ell$. Then for all $i \in [\ell]$, $j \in [6/\epsilon^2]$, it computes the following linear function $A_{i,j} = \sum_{k \in [n], h_i(k) = j} g_i(k)\mathcal{X}_k$, and outputs an approximation $\tilde{\mathcal{X}}$ of $\mathcal{X}$ given by $\tilde{\mathcal{X}}_k = median_{i \in [\ell]}\{g_i(k)A_{i,h_i(k)}\}$*

Observe that the table $A \in \mathbb{R}^{\ell \times 6/\epsilon^2}$ can be flattened into a vector $A \in \mathbb{R}^{6\ell/\epsilon^2}$. Given this, A can be represented as $A = S\mathcal{X}$ for a matrix $S \in \mathbb{R}^{6\ell/\epsilon^2 \times n}$. For any $i \in [\ell]$, $j \in [6/\epsilon^2]$, and $\ell \in [n]$, the matrix S is given by $S_{(i-1)(6/\epsilon^2)+j,\ell} = \delta_{i,j,\ell} g_j(\ell)$, where $\delta_{i,j,\ell}$ indicates the event that $h_i(\ell) = j$. Given $S\mathcal{X}$, one can solve the point-estimation problem as described in Definition 13 [18]. In order to reduce the communication from sending each coordinate of $S\mathcal{X}$ exactly, we can use our rounding procedure to approximately compute the sketch $S\mathcal{X}$, which will give us the following theorem.

▶ **Theorem 14.** *Consider a message passing topology $G = (V, E)$ with diameter d, where the i-th player is given as input $X_i \in \mathbb{Z}^n_{\geq 0}$ and $\mathcal{X} = \sum_{i=1}^m X_i$. Then there is a communication protocol which outputs an estimate $\tilde{\mathcal{X}} \in \mathbb{R}^n$ of $\mathcal{X}$ such that $\|\tilde{\mathcal{X}} - \mathcal{X}\|_\infty \leq \epsilon \|\mathcal{X}_{tail(1/\epsilon^2)}\|_2$ with probability $1 - 1/n^c$ for any constant $c \geq 1$. The protocol uses $O(\frac{m}{\epsilon^2}\log(n)(\log(\log(n)) + \log(d) + \log(1/\epsilon)))$ total communication, and a max communication of $O(\frac{1}{\epsilon^2}\log(n)(\log(\log(n)) + \log(d) + \log(1/\epsilon)))$.*

4 F_p Estimation for $p < 1$

In this section, we develop algorithms for F_p estimation for $p < 1$ in the message passing model, and in the process obtain improved algorithms for entropy estimation. We begin by reviewing the fundamental sketching procedure used in our estimation protocol. The algorithm is known as a Morris counter [51, 26]. The algorithm first picks a base $1 < b \leq 2$, and initializes a counter $C \leftarrow 0$. Then, every time it sees an insertion, it increments the counter $C \leftarrow C + \delta$, where $\delta = 1$ with probability b^{-C}, and $\delta = 0$ otherwise (in which case the counter remains unchanged). After n insertions, the value n can be estimated by $\tilde{n} = (b^C - b)/(b - 1) + 1$.

▶ **Definition 15.** *The approximate counting problem is defined as follows. Each player i is given a positive integer value $x_i \in \mathbb{Z}_{\geq 0}$, and the goal is for some player at the end to hold an estimate of $x = \sum_i x_i$.*

▶ **Proposition 16** (Proposition 5 [26])**.** *If C_n is the value of the Morris counter after n updates, then $\mathbb{E}[\tilde{n}] = n$, and $\boldsymbol{Var}[\tilde{n}] = (b-1)n(n+1)/2$.*

▶ **Corollary 17.** *If C_n is the value of a Morris counter run on a stream of n insertions with base $b = (1 + (\epsilon\delta)^2)$, then with probability at least $1 - \delta$, we have $\tilde{n} = (1 \pm \epsilon)n$ with probability at least $1 - \delta$. Moreover, with probability at least $1 - \delta$, the counter C_n requires $O(\log\log(n) + \log(1/\epsilon) + \log(1/\delta))$-bits to store.*

▶ **Lemma 18.** *Given Morris counters X, Y run on streams of length n_1, n_2 respectively, There is a merging procedure that produces a Morris counter Z which is distributed identically to a Morris counter that was run on a stream of $n_1 + n_2$ insertions.*

▶ **Corollary 19.** *There is a protocol for F_1 estimation of non-negative vectors, equivalently for the approximate counting problem, in the message passing model which succeeds with probability $1 - \delta$ and uses a max-communication of $O((\log\log(n) + \log(1/\epsilon) + \log(1/\delta))$-bits.*

We now note that Morris counters can easily used as approximate counters for streams with both insertions and deletions (positive and negative updates), by just storing a separate Morris counter for the insertions and deletions, and subtracting the estimate given by one from the other at the end.

▶ **Corollary 20.** *Using two Morris counters separately for insertions and deletions, on a stream of I insertions and D deletions, there is an algorithm, called a signed Morris counter, which produces $\tilde{n}$ with $|\tilde{n} - n| \leq \epsilon(I + D)$, where $n = I - D$, with probability $1 - \delta$, using space $O(\log\log(I + D) + \log(1/\epsilon) + \log(1/\delta))$.*

Hereafter, when we refer to a Morris counter that is run on a stream which contains both positive and negative updates as a *signed* Morris counter. Therefore, the guarantee of Corollary 20 apply to such signed Morris counters, and moreover such signed Morris counters can be Merged as in Lemma 18 with the same guarantee.

Algorithm 2 Multi-party F_p estimation protocol, $p < 1$.

Procedure for player j

$k \leftarrow \Theta(1/\epsilon^2)$, $\epsilon' \leftarrow \Theta(\epsilon \frac{\delta^{1/p}}{\log(n/\delta)})$, $\delta \leftarrow 1/(200k)$

1. Using shared randomness, choose sketching matrix $S \in \mathbb{R}^{k \times n}$ of i.i.d. p-stable random variables, with $k = \Theta(1/\epsilon)$. Generate S up to precision $\eta = \text{poly}(1/(n, m, M))$, so that $\eta^{-1}S$ has integral entries.
2. For each $i \in [k]$, receive signed Morris counters $y_{j_1,i}, y_{j_2,i}, \ldots, y_{j_t,i}$ from the $t \in \{0, \ldots, m\}$ children of node j in the prior layer.
3. Compute $\eta^{-1}\langle S_i, X_j\rangle \in \mathbb{Z}$, where S_i is the i-th row of S, and run a new signed Morris counter C on $\eta^{-1}\langle S_i, X_j\rangle$ with parameters (ϵ', δ').
4. Merge the signed Morris counters $y_{j_1,i}, y_{j_2,i}, \ldots, y_{j_t,i}, C$ into a counter $y_{j,i}$.
5. Send the merged signed Morris counter $y_{j,i}$ to the parent of player j. If player j is the root node $\mathcal{C}$, then set C_i to be the estimate of the signed Morris counter $y_{j,i}$, and return the estimate $\eta \cdot \text{median}\,\{\frac{|C_1|}{\theta_p}, \ldots, \frac{|C_k|}{\theta_p}\}$, where θ_p is the median of the distribution $\mathcal{D}_p$.

We now provide our algorithm for F_p estimation in the message passing model with $p \leq 1$. Our protocol is similar to our algorithm for $p \geq 1$. We fix a vertex $\mathcal{C}$ which is a center of the communication topology. We then consider the shortest path tree T rooted at $\mathcal{C}$, which has depth at most d, where d is the diameter of G. The players then choose random vectors $S_i \in \mathbb{R}^n$ for $i \in [k]$, and the j-th player computes $\langle S_i, X_j\rangle$, and adds this value to a Morris counter. Each player receives Morris counters from their children in T, and thereafter merges these Morris counters with its own. Finally, it sends this merged Morris counter, containing updates from all players in the subtree rooted at j, to the parent of j in T. At the end, the center $\mathcal{C}$ holds a Morris counter C_i which approximates $\sum_j \langle S_i, X_j\rangle$. The main algorithm for each player j is given formally as Algorithm 2.

▶ **Theorem 21.** *For $p \in (0, 1)$, there is a protocol for F_p estimation in the message passing model which succeeds with probability $2/3$ and uses a total communication of $O(\frac{m}{\epsilon^2}(\log\log(n) + \log(1/\epsilon))$-bits, and a max-communication of $O(\frac{1}{\epsilon^2}(\log\log(n) + \log(1/\epsilon))$-bits. The protocol requires a total of at most d rounds, where d is the diameter of the communication topology G.*

4.1 The Streaming Algorithm for F_p Estimation, $p < 1$

As discussed earlier, the insertion-only streaming model of computation is a special case of the above communication setting, where the graph in question is the line graph, and each player receives vector $X_i \in \mathbb{R}^n$ which is the standard basis vector $e_j \in \mathbb{R}^n$ for some $j \in [n]$. The only step remaining to fully generalize the result to the streaming setting is an adequate derandomization of the randomness required to generate the matrix S. Our derandomization will follow from the results of [45], which demonstrate that, using a slightly different estimator known as the log-cosine estimator, the entries of each row S_i can be generated with only $\Theta(\log(1/\epsilon)/\log\log(1/\epsilon))$-wise independence, and the seeds used to generate separate rows of S_i need only be pairwise independent. Thus, storing the randomness used to generate S requires only $O(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\log(n))$-bits of space.

We now discuss the estimator of [45] precisely. The algorithm generates a matrix $S \in \mathbb{R}^{k\times n}$ and $S' \in \mathbb{R}^{k'\times n}$ with $k = \Theta(1/\epsilon^2)$ and $k' = \Theta(1)$, where each entry of S, S' is drawn from $\mathcal{D}_p$. For a given row i of S, the entries $S_{i,j}$ are $\Theta(\log(1/\epsilon)/\log\log(1/\epsilon))$-wise independent, and for $i \neq i'$, the seeds used to generate $\{S_{i,j}\}_{j=1}^n$ and $\{S_{i',j}\}_{j=1}^n$ are pairwise independent. S' is generated with only $\Theta(1)$-wise independence between the entries in a given row in S', and pairwise independence between rows. The algorithm then maintains the vectors $y = S\mathcal{X}$ and $y' = S'\mathcal{X}$ throughout the stream, where $\mathcal{X} \in \mathbb{Z}^n_{\geq 0}$ is the stream vector. Define $y'_{med} = \text{median}\{|y'_i|\}_{i=1}^{k'}/\theta_p$, where θ_p is the median of the distribution $\mathcal{D}_p$ ([45] discusses how this can be approximated to $(1 \pm \epsilon)$ efficiently). The log-cosine estimator R of $\|\mathcal{X}\|_p$ is then given by $R = y'_{med} \cdot \left(-\ln\left(\frac{1}{k}\sum_{i=1}^k \cos\left(\frac{y_i}{y'_{med}}\right)\right)\right)$

▶ **Theorem 22.** *There is a streaming algorithm for insertion only F_p estimation, $p \in (0,1)$, outputs a value $\tilde{R}$ such that with probability at least $2/3$, we have that $|\tilde{R} - \|\mathcal{X}\|_p| \leq \epsilon\|\mathcal{X}\|_p$ where $\mathcal{X} \in \mathbb{R}^n$ is the state of the stream vector at the end of the stream. The algorithm uses $O((\frac{1}{\epsilon^2}(\log\log(n) + \log(1/\epsilon)) + \frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\log(n))$-bits of space.*

5 Entropy Estimation

In this section, we show how our results imply improved algorithms for entropy estimation in the message-passing model. Here, for a vector $\mathcal{X} \in \mathbb{R}^n$, the Shannon entropy is given by $H = \sum_{i=1}^n \frac{|\mathcal{X}_i|}{\|\mathcal{X}\|_1}\log(\frac{\|\mathcal{X}\|_1}{|\mathcal{X}_i|})$. We follow the approach taken by [21, 47, 31, 32] for entropy estimation in data streams, which is to use sketched of independent *maximally-skewed stable random variables*. While we introduced p-stable random variables in Definition 3 as the distribution with characteristic function $\mathbb{E}[e^{itZ}] = e^{-|t|^p}$, we remark now that the p-stable distribution is also parameterized by an additional *skewness* parameter $\beta \in [-1,1]$. Up until this point, we have assumed $\beta = 0$. In this section, however, we will be using maximally skewed, meaning $\beta = -1$, $p = 1$-stable random variables. We introduce these now

▶ **Definition 23** (Stable distribution, general)**.** *There is a distribution $F(p,\beta,\gamma,\delta)$ called the p-stable distribution with skewness parameter $\beta \in [-1,1]$, scale γ, and position δ. The characteristic function of a $Z \sim F(p,\beta,\gamma,\delta)$ variable Z is given by:*

$$\mathbb{E}[e^{-itZ}] = \begin{cases} \exp\left(-\gamma^p|t|^p\left[1 - i\beta\tan(\frac{\pi p}{2})\text{sign}(t)\right] + i\delta t\right) & \text{if } p \in (0,2]\setminus\{1\} \\ \exp\left(-\gamma|t|\left[1 + i\beta\frac{2}{\pi}\text{sign}(t)\log(|t|)\right] + i\delta t\right) & \text{if } p = 1 \end{cases}$$

where sign$(t) \in \{1,-1\}$ is the sign of a real $t \in \mathbb{R}$. Moreover, if $Z \sim F(p,\beta,\gamma,0)$ for any $\beta \in [-1,1]$ and $0 < p < 2$, for any $\lambda > 0$ we have $\Pr[|Z| > C\lambda] \leq (\frac{\gamma}{\lambda})^p$, where C is some universal constant. We refer the reader to [54] for a further discussion on the parameterization and behavior of p-stable distributions with varying rates.

Algorithm 3 Entropy Estimation algorithm of [21].

Sketching algorithm for Entropy Estimation

Input: $\mathcal{X} \in \mathbb{R}^n$

1. Generate $S \in \mathbb{R}^{k \times n}$ for $k = \Theta(1/\epsilon^2)$ of i.i.d. $F(1, -1, \pi/2, 0)$ random variables to precision $\eta = 1/\text{poly}(M, n)$.
2. Compute $S\mathcal{X} \in \mathbb{R}^k$.
3. Set $y_i \leftarrow (S\mathcal{X})_i / \|\mathcal{X}\|_1$ for $i \in [k]$
4. Return $\tilde{H} = -\log\left(\frac{1}{k}\sum_{i=1}^k e^{y_i}\right)$

The algorithm of [21] is given formally as Algorithm 3. The guarantee of the algorithm is given in Theorem 24.

▶ **Theorem 24** ([21]). *The above estimate $\tilde{H}$ satisfies $|\tilde{H} - H| < \epsilon$ with probability at least $9/10$.*

▶ **Lemma 25.** *Fix $0 < \epsilon_0 < \epsilon$. Let $S \in \mathbb{R}^{k \times n}$ with $k = \Theta(1/\epsilon^2)$ be a matrix of i.i.d. $F(1, -1, \pi/2, 0)$ random variables to precision $\eta = 1/poly(M, n)$. Then there is a protocol in the message passing model that outputs $Y \in \mathbb{R}^k$ at a centralized vertex with $\|Y - S\mathcal{X}\|_\infty \leq \epsilon_0 \|\mathcal{X}\|_1$ with probability $9/10$. The protocol uses a total communication of $O(\frac{m}{\epsilon^2}(\log\log(n) + \log(1/\epsilon_0))$-bits, and a max-communication of $O(\frac{1}{\epsilon^2}(\log\log(n) + \log(1/\epsilon_0)))$-bits.*

▶ **Theorem 26.** *There is a multi-party communication protocol in the message passing model that outputs a ϵ-additive error of the Shannon entropy H. The protocol uses a max-communication of $O(\frac{1}{\epsilon^2}(\log\log(n) + \log(1/\epsilon))$-bits.*

Since our protocol does not depend on the topology of G, a direct corollary is that we obtain a $\tilde{O}(\epsilon^{-2})$-bits of space *streaming* algorithm for entropy estimation in the random oracle model. Recall that the random oracle model allows the streaming algorithm query access to an arbitrarily long tape of random bits. This fact is used to store the random sketching matrix S.

▶ **Theorem 27.** *There is a streaming algorithm for ϵ-additive approximation of the empirical Shannon entropy of an insertion only stream in the random oracle model, which succeeds with probability $3/4$. The space required by the algorithm is $O(\frac{1}{\epsilon^2}(\log\log(n) + \log(1/\epsilon))$ bits.*

6 Approximate Matrix Product in the Message Passing Model

In this section, we consider the approximate regression problem in the message passing model over a topology $G = (V, E)$. Here, instead of vector valued inputs, each player is given as input two integral matrices $X_i \in \{0, 1, 2, \ldots, M\}^{n \times t_1}$, $Y_i \in \{0, 1, 2, \ldots, M\}^{n \times t_2}$. It is generally assumed that $n >> t_1, t_2$, so the matrices X_i, Y_i are rectangular. Let $\mathcal{X} = \sum_{i=1}^m X_i$ and $\mathcal{Y} = \sum_i Y_i$. The goal of the players is to approximate the matrix product $\mathcal{X}^T\mathcal{Y} \in \mathbb{R}^{t_1 \times t_2}$. Specifically, at the end of the protocol one player must output a matrix $R \in \mathbb{R}^{t_1 \times t_2}$ such that $\|R - \mathcal{X}^T\mathcal{Y}\|_F \leq \epsilon \|\mathcal{X}\|_F \|\mathcal{Y}\|_F$, where for a matrix A, $\|A\|_F = (\sum_{i,j} A_{i,j}^2)^{1/2}$ is the Frobenius norm of A.

We now describe a classic sketching algorithm which can be used to solve the approximate regression problem. The algorithm picks a $S \in \mathbb{R}^{k \times n}$ of i.i.d. Gaussian variables with variance $1/k$. It then computes $S\mathcal{X}$ and $S\mathcal{Y}$, and outputs $(S\mathcal{X})^T S\mathcal{Y}$. The following fact about such sketches will demonstrate correctness.

▶ **Lemma 28** ([43]). *Fix matrices $\mathcal{X} \in \mathbb{R}^{n\times t_1}, \mathcal{Y} \in \mathbb{R}^{n\times t_2}$ and $0 < \epsilon_0$. Let $S \in \mathbb{R}^{k\times n}$ be a matrix of i.i.d. Gaussian random variables with variance $1/k$, for $k = \Theta(1/(\delta\epsilon_0^2))$. Then we have $\Pr[\|\mathcal{X}^T S^T S\mathcal{Y} - \mathcal{X}^T\mathcal{Y}\|_F \le \epsilon_0\|\mathcal{X}\|_F\|\mathcal{Y}\|_F] \ge 1-\delta$. Moreover, with the same probability we have $\|S\mathcal{X}\|_F = (1\pm\epsilon_0)\|\mathcal{X}\|_F$ and $\|S\mathcal{Y}\|_F = (1\pm\epsilon_0)\|\mathcal{Y}\|_F$*

Now by Lemma 11, the central vertex $\mathcal{C}$ can recover a value $r_{\mathcal{C}}^{i,j}$ such that $\mathbb{E}[r_{\mathcal{C}}^{i,j}] = (S\mathcal{X})_{i,j}$ and $\mathbf{Var}[r_{\mathcal{C}}^{i,j}] \le \epsilon^2\|\mathcal{X}_{*,j}\|_2$ (after setting δ sufficiently small), where $\mathcal{X}_{*,j}$ is the j-th *column* of $\mathcal{X}$. Thus, the central vertex can obtain a random matrix $R^{\mathcal{X}} \in \mathbb{R}^{k\times t_1}$ such that $\mathbb{E}[R^{\mathcal{X}}] = (S\mathcal{X})$ and $\mathbb{E}[\|R^{\mathcal{X}} - S\mathcal{X}\|_F^2] \le k\epsilon^2\sum_{j=1}^{t_1}\|\mathcal{X}_{*,j}\|_2$. Setting $\epsilon = \text{poly}(1/k) = \text{poly}(1/\epsilon_0)$ small enough, we obtain $\mathbb{E}[\|R^{\mathcal{X}} - S\mathcal{X}\|_F^2] \le \epsilon_0^2\|\mathcal{X}\|_F$. Similarly, we can obtain a $R^{\mathcal{Y}}$ at the central vertex $\mathcal{C}$, and output the estimate $R = (R^{\mathcal{X}})^T R^{\mathcal{Y}}$. Utilizing the error guarantees of Lemma 11 as well as Lemma 28, we obtain the following theorem.

▶ **Theorem 29.** *Given inputs $\mathcal{X} = \sum_{i=1}^m X_i, \mathcal{Y} = \sum_{i=1}^m Y_i$ as described above, there is a protocol which outputs, at the central vertex $\mathcal{C}$, a matrix $R \in \mathbb{R}^{t_1\times t_2}$ such that with probability $3/4$ we have $\|R - \mathcal{X}^T\mathcal{Y}\|_F \le \epsilon\|\mathcal{X}\|_F\|\mathcal{Y}\|_F$ The max communication required by the protocol is $O\left(\epsilon^{-2}(t_1+t_2)(\log\log n + \log 1/\epsilon + \log d)\right)$, where d is the diameter of the communication topology G.*

References

1. Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 20–29. ACM, 1996.
2. Alexandr Andoni, Robert Krauthgamer, and Krzysztof Onak. Streaming algorithms from precision sampling. *arXiv preprint*, 2010. `arXiv:1011.1263`.
3. Chrisil Arackaparambil, Joshua Brody, and Amit Chakrabarti. Functional monitoring without monotonicity. In *International Colloquium on Automata, Languages, and Programming*, pages 95–106. Springer, 2009.
4. Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and issues in data stream systems. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–16. ACM, 2002.
5. Maria Florina Balcan, Yingyu Liang, Le Song, David Woodruff, and Bo Xie. Communication efficient distributed kernel principal component analysis. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 725–734. ACM, 2016.
6. Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.
7. Jarosław Błasiok, Jian Ding, and Jelani Nelson. Continuous monitoring of lp norms in data streams. *arXiv preprint*, 2017. `arXiv:1704.06710`.
8. Mark Braverman, Faith Ellen, Rotem Oshman, Toniann Pitassi, and Vinod Vaikuntanathan. A tight bound for set disjointness in the message-passing model. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 668–677. IEEE, 2013.
9. Vladimir Braverman, Stephen R Chestnut, Nikita Ivkin, Jelani Nelson, Zhengyu Wang, and David P Woodruff. BPTree: an L2 heavy hitters algorithm using constant memory. *arXiv preprint*, 2016. `arXiv:1603.00759`.
10. Vladimir Braverman, Stephen R Chestnut, David P Woodruff, and Lin F Yang. Streaming space complexity of nearly all functions of one variable on frequency vectors. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 261–276. ACM, 2016.

11 Vladimir Braverman, Jonathan Katzman, Charles Seidell, and Gregory Vorsanger. An Optimal Algorithm for Large Frequency Moments Using O (nˆ(1-2/k)) Bits. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

12 Vladimir Braverman and Rafail Ostrovsky. Recursive sketching for frequency moments. *arXiv preprint*, 2010. `arXiv:1011.2571`.

13 Vladimir Braverman, Emanuele Viola, David Woodruff, and Lin F Yang. Revisiting frequency moment estimation in random order streams. *arXiv preprint*, 2018. `arXiv:1803.02270`.

14 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. A near-optimal algorithm for estimating the entropy of a stream. *ACM Transactions on Algorithms (TALG)*, 6(3):51, 2010.

15 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Robust Lower Bounds for Communication and Stream Computation. *Theory of Computing*, 12(1):1–35, 2016.

16 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *18th IEEE Annual Conference on Computational Complexity, 2003. Proceedings.*, pages 107–117. IEEE, 2003.

17 Amit Chakrabarti and Oded Regev. An optimal lower bound on the communication complexity of gap-hamming-distance. *SIAM Journal on Computing*, 41(5):1299–1317, 2012.

18 Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. *Automata, languages and programming*, pages 784–784, 2002.

19 Arkadev Chattopadhyay, Jaikumar Radhakrishnan, and Atri Rudra. Topology matters in communication. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 631–640. IEEE, 2014.

20 Jiecao Chen, He Sun, David Woodruff, and Qin Zhang. Communication-optimal distributed clustering. In *Advances in Neural Information Processing Systems*, pages 3727–3735, 2016.

21 Peter Clifford and Ioana Cosma. A simple sketching algorithm for entropy estimation over streaming data. In *Artificial Intelligence and Statistics*, pages 196–206, 2013.

22 Graham Cormode, Piotr Indyk, Nick Koudas, and S Muthukrishnan. Fast mining of massive tabular data via approximate distance computations. In *Proceedings 18th International Conference on Data Engineering*, pages 605–614. IEEE, 2002.

23 Graham Cormode and Hossein Jowhari. L p Samplers and Their Applications: A Survey. *ACM Computing Surveys (CSUR)*, 52(1):16, 2019.

24 Graham Cormode, S Muthukrishnan, and Irina Rozenbaum. Summarizing and mining inverse distributions on data streams via dynamic inverse sampling. In *Proceedings of the 31st international conference on Very large data bases*, pages 25–36. VLDB Endowment, 2005.

25 Graham Cormode, S Muthukrishnan, and Ke Yi. Algorithms for distributed functional monitoring. *ACM Transactions on Algorithms (TALG)*, 7(2):21, 2011.

26 Philippe Flajolet. Approximate counting: a detailed analysis. *BIT Numerical Mathematics*, 25(1):113–134, 1985.

27 Phillip B Gibbons and Yossi Matias. New sampling-based summary statistics for improving approximate query answers. In *ACM SIGMOD Record*, volume 27, pages 331–342. ACM, 1998.

28 Phillip B Gibbons, Yossi Matias, and Viswanath Poosala. Fast Incremental Maintenance of Approximate Histograms. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 466–475. Morgan Kaufmann Publishers Inc., 1997.

29 Anna C Gilbert, Yannis Kotidis, S Muthukrishnan, and Martin J Strauss. How to summarize the universe: Dynamic maintenance of quantiles. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 454–465. Elsevier, 2002.

30 Andre Gronemeier. Asymptotically optimal lower bounds on the nih-multi-party information. *arXiv preprint*, 2009. `arXiv:0902.1609`.

31 Nicholas JA Harvey, Jelani Nelson, and Krzysztof Onak. Sketching and streaming entropy via approximation theory. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 489–498. IEEE, 2008.

32 Nicholas JA Harvey, Jelani Nelson, and Krzysztof Onak. Streaming algorithms for estimating entropy. In *2008 IEEE Information Theory Workshop*, pages 227–231. IEEE, 2008.

33 Ling Huang, XuanLong Nguyen, Minos Garofalakis, Joseph M Hellerstein, Michael I Jordan, Anthony D Joseph, and Nina Taft. Communication-efficient online detection of network-wide anomalies. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 134–142. IEEE, 2007.

34 Zengfeng Huang, Ke Yi, and Qin Zhang. Randomized algorithms for tracking distributed count, frequencies, and ranks. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*, pages 295–306. ACM, 2012.

35 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *Journal of the ACM (JACM)*, 53(3):307–323, 2006.

36 Piotr Indyk and David Woodruff. Optimal approximations of the frequency moments of data streams. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 202–208. ACM, 2005.

37 Rajesh Jayaram, Gokarna Sharma, Srikanta Tirthapura, and David P. Woodruff. Weighted Reservoir Sampling from Distributed Streams. In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, SIGMOD/PODS '19, 2019.

38 Rajesh Jayaram and David P Woodruff. Perfect lp sampling in a data stream. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 544–555. IEEE, 2018.

39 Thathachar S Jayram, Ravi Kumar, and D Sivakumar. The One-Way Communication Complexity of Hamming Distance. *Theory of Computing*, 4(1):129–135, 2008.

40 Thathachar S Jayram and David P Woodruff. The data stream space complexity of cascaded norms. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 765–774. IEEE, 2009.

41 TS Jayram. Hellinger strikes back: A note on the multi-party information complexity of AND. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 562–573. Springer, 2009.

42 Hossein Jowhari, Mert Sağlam, and Gábor Tardos. Tight Bounds for Lp Samplers, Finding Duplicates in Streams, and Related Problems. In *Proceedings of the Thirtieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '11, pages 49–58, New York, NY, USA, 2011. ACM. `doi:10.1145/1989284.1989289`.

43 Daniel M Kane and Jelani Nelson. Sparser johnson-lindenstrauss transforms. *Journal of the ACM (JACM)*, 61(1):4, 2014.

44 Daniel M Kane, Jelani Nelson, Ely Porat, and David P Woodruff. Fast moment estimation in data streams in optimal space. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 745–754. ACM, 2011.

45 Daniel M Kane, Jelani Nelson, and David P Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1161–1178. SIAM, 2010.

46 Michael Kapralov, Jelani Nelson, Jakub Pachocki, Zhengyu Wang, David P Woodruff, and Mobin Yahyazadeh. Optimal lower bounds for universal relation, and for samplers and finding duplicates in streams. *arXiv preprint*, 2017. `arXiv:1704.00633`.

47 Ping Li and Cun-Hui Zhang. A new algorithm for compressed counting with applications in shannon entropy estimation in dynamic data. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 477–496, 2011.

48 Yi Li and David P Woodruff. A tight lower bound for high frequency moment estimation with small error. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 623–638. Springer, 2013.

49 Andrew McGregor, A Pavan, Srikanta Tirthapura, and David P Woodruff. Space-Efficient Estimation of Statistics Over Sub-Sampled Streams. *Algorithmica*, 74(2):787–811, 2016.

50 Morteza Monemizadeh and David P Woodruff. 1-pass relative-error lp-sampling with applications. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1143–1160. SIAM, 2010.

51 Robert Morris. Counting large numbers of events in small registers. *Communications of the ACM*, 21(10):840–842, 1978.

52 Shanmugavelayutham Muthukrishnan et al. Data streams: Algorithms and applications. *Foundations and Trends® in Theoretical Computer Science*, 1(2):117–236, 2005.

53 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

54 J. P. Nolan. *Stable Distributions - Models for Heavy Tailed Data*. Birkhauser, Boston, 2018. In progress, Chapter 1 online at http://fs2.american.edu/jpnolan/www/stable/stable.html.

55 Frank Olken. *Random sampling from databases*. PhD thesis, University of California, Berkeley, 1993.

56 Srikanta Tirthapura and David P Woodruff. Optimal random sampling from distributed streams revisited. In *International Symposium on Distributed Computing*, pages 283–297. Springer, 2011.

57 Omri Weinstein and David P Woodruff. The simultaneous communication of disjointness with applications to data streams. In *International Colloquium on Automata, Languages, and Programming*, pages 1082–1093. Springer, 2015.

58 David Woodruff. Optimal space lower bounds for all frequency moments. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 167–175. Society for Industrial and Applied Mathematics, 2004.

59 David P Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

60 David P Woodruff and Qin Zhang. Tight bounds for distributed functional monitoring. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 941–960. ACM, 2012.

61 David P Woodruff and Qin Zhang. Distributed Statistical Estimation of Matrix Products with Applications. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 383–394. ACM, 2018.

62 David P Woodruff and Peilin Zhong. Distributed low rank approximation of implicit functions of a matrix. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 847–858. IEEE, 2016.

63 Ke Yi and Qin Zhang. Optimal tracking of distributed heavy hitters and quantiles. *Algorithmica*, 65(1):206–223, 2013.

A Proof Sketch of $\Omega(m/\epsilon^2)$ Lower Bound for F_p estimation in the One-Way Coordinator Model

We now sketch the proof of the $\Omega(m/\epsilon^2)$ lower bound that was remarked upon in the introduction. First, consider the following problem Alice is given a vector $x \in \mathbb{R}^t$, and bob $y \in \mathbb{R}^t$, such that $x_i \geq 0, y_i \geq 0$ for all $i \in [t]$. Alice and Bob both send a message to Eve, who must then output a $(1 \pm \epsilon)$ approximation to $\|x + y\|_p$, for $p \in (0, 2] \setminus \{1\}$. Via a reduction from the Gap-Hamming communication problem, there is a $\Omega(1/\epsilon^2)$-bit communication lower bound for this problem [58]. More specifically, there is a distribution $\mathcal{D}$ over inputs $(x, y) \in \mathbb{R}^t \times \mathbb{R}^t$, such that any communication protocol that solves the above problem on these inputs correctly with probability $3/4$ must send $\Omega(1/\epsilon^2)$ bits.

Now consider the one-way coordinator model, where there are m players connected via an edge to a central coordinator. They are given inputs $x_1, \ldots, x_m$, and must each send a single message to the coordinator, who them must estimate $\|x\|_p = \|x_1 + x_2 + \cdots + x_m\|_p$. Consider

two distribution, P_1, P_2 over the inputs $(x_1, \ldots, x_m)$. In the first, two players i, j are chosen uniformly at random, and given as inputs $(x, y) \sim \mathcal{D}$, and the rest of the players are given the 0 vector. In P_2, we draw $(x, y) \sim \mathcal{D}$, and every player is given either x or y at random. The players are then either given input from P_1 or P_2, with probability $1/2$ for each. In the first case, if the two players with the input do not send $\Omega(1/\epsilon^2)$ bits, then they will not be able to solve the estimation problem via the 2-party lower bound. However, given only their input, the distributions P_1 and P_2 are indistinguishable to a given player. So the players cannot tell if the input is from P_1 or P_2, so any player that gets an non-zero input must assume they are in case P_1 if they want to solve the communication problem with sufficiently high constant probability, and send $\Omega(1/\epsilon^2)$ bits of communication. This results in $\Omega(m/\epsilon^2)$ total communication when the input is from P_2, which is the desired lower bound.

B $\Omega(1/\epsilon^2)$ Lower Bound for additive approximation of Entropy in Insertion-Only Streams

We now prove the $\Omega(1/\epsilon^2)$-bits of space lower bound for any streaming algorithm that produces an approximation $\tilde{H}$ such that $|\tilde{H} - H| < \epsilon$ with probability $3/4$. Here H is the empirical entropy of the stream vector $\mathcal{X}$, namely $H = H(\mathcal{X}) = -\sum_{i=1}^{n} \frac{|\mathcal{X}_i|}{F_1} \log \frac{|\mathcal{X}_i|}{F_1}$. To prove the lower bound, we must first introduce the GAP-HAMDIST problem. Here, there are two players, Alice and Bob. Alice is given $x \in \{0,1\}^t$ and Bob receives $y \in \{0,1\}^t$. Let $\Delta(x, y) = |\{i \mid x_i \neq y_i\}|$ be the Hamming distance between two binary strings x, y. Bob is promised that either $\Delta(x, y) \leq t/2 - \sqrt{t}$ (NO instance) or $\Delta(x, y) \geq t/2 + \sqrt{t}$ (YES instance), and must decide which holds. Alice must send a single mesage to Bob, from which he must decide which case the inputs are in. It is known that any protocol which solves this problem with constant probability must send $\Omega(t)$-bits in the worst case (i.e. the maximum number of bits sent, taken over all inputs and random bits used by the protocol).

▶ **Proposition 30** ([58, 39]). *Any protocol which solves the* GAP-HAMDIST *problem with probability at least* $2/3$ *must send* $\Omega(t)$*-bits of communication in the worst case.*

We remark that while a $\Omega(1/\epsilon^2)$ lower bound is known for *multiplicative-approximation* of the entropy, to the best of our knowledge there is no similar lower bound written in the literature for additive approximation.

▶ **Theorem 31.** *Any algorithm for* ϵ*-additive approximation of the entropy* H *of a stream, in the insertion-only model, which succeeds with probability at least* $2/3$*, requires space* $\Omega(\epsilon^{-2})$

Proof. Given a $x, y \in \{0,1\}^t$ instance of GAP-HAMDIST, for $t = \Theta(1/\epsilon^2)$, Alice constructs a stream on $2t$ items. Let x' be the result of flipping all the bits of x, and let $x'' = x \circ 0^t + 0^t \circ x' \in \{0,1\}^{2t}$ where $\circ$ denotes concatenation. Define y', y'' similarly. Alice then inserts updates so that the stream vector $\mathcal{X} = x''$, and then sends the state of the streaming algorithm to Bob, who inserts his vector, so that now $\mathcal{X} = x'' + y''$. We demonstrate that the entropy of H differs by an additive term of at least ϵ between the two cases. In all cases case, we have

$$\begin{aligned} H &= \frac{t-\Delta}{t}\log(t) + \frac{\Delta}{2t}\log(2t) \\ &= \log(t) + \Delta\left(\frac{2\log(t) - \log 2t}{2t}\right) \end{aligned} \tag{1}$$

We can assume $t \geq 4$, and then $2\log(t) - \log(2t) = C > 0$, where C is some fixed value known to both players that is bounded away from 0. So as Δ increases, the entropy increases. Thus in a YES instance, the entropy is at least

$$\begin{aligned} H &\geq \log(t) + (t/2 + \sqrt{t})\frac{C}{2t} \\ &= \log(t) + (1/4 + 1/2\sqrt{t})C \\ &= \log(t) + C/4 + \Theta(\epsilon) \end{aligned} \tag{2}$$

In addition, in the NO instance, the entropy is maximized when $\Delta = t/2 - \sqrt{T}$. so we have

$$\begin{aligned} H &\leq \log(t) + (t/2 - \sqrt{t})\frac{C}{2t} \\ &= \log(t) + C/4 - \Theta(\epsilon) \end{aligned} \tag{3}$$

Therefore, the entropy differs between YES and NO instances by at least an additive $\Theta(\epsilon)$ term. After sufficient rescaling of ϵ by a constant, we obtain our $\Omega(t) = \Omega(1/\epsilon^2)$ lower bound for additive entropy estimation via the linear lower bound for GAP-HAMDIST from Proposition 30. ◀

The Complexity of Partial Function Extension for Coverage Functions

Umang Bhaskar
Tata Institute of Fundamental Research, Mumbai, India
umang@tifr.res.in

Gunjan Kumar
Tata Institute of Fundamental Research, Mumbai, India
gunjan.kumar@tifr.res.in

Abstract

Coverage functions are an important subclass of submodular functions, finding applications in machine learning, game theory, social networks, and facility location. We study the complexity of partial function extension to coverage functions. That is, given a partial function consisting of a family of subsets of $[m]$ and a value at each point, does there exist a coverage function defined on all subsets of $[m]$ that extends this partial function? Partial function extension is previously studied for other function classes, including boolean functions and convex functions, and is useful in many fields, such as obtaining bounds on learning these function classes.

We show that determining extendibility of a partial function to a coverage function is NP-complete, establishing in the process that there is a polynomial-sized certificate of extendibility. The hardness also gives us a lower bound for learning coverage functions. We then study two natural notions of approximate extension, to account for errors in the data set. The two notions correspond roughly to multiplicative point-wise approximation and additive L_1 approximation. We show upper and lower bounds for both notions of approximation. In the second case we obtain nearly tight bounds.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Coverage Functions, PAC Learning, Approximation Algorithm, Partial Function Extension

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.30

Category APPROX

Related Version A full version of the paper is available at `https://arxiv.org/abs/1907.07230`.

Funding *Umang Bhaskar*: Supported in part by a Ramanujan fellowship and an Early Career Research award.

1 Introduction

When can a *partial function* – given as a set of points from a domain, and a value at each point – be extended to a *total function* on the domain, that lies in some particular class of functions? This is the basic question of *partial function extension*, and is studied both independently (such as in convex analysis) and as a recurring subproblem in many areas in combinatorial optimization, including computational learning and property testing.

In this paper we study the computational complexity of partial function extension for *coverage functions*. Coverage functions are a natural and widely-studied subclass of submodular functions that find many applications, including in machine learning [18], auctions [6, 19], influence maximization [8, 22], and plant location [11]. For a natural number m, let $[m]$ denote the set $\{1, 2, \ldots, m\}$. A set function $f : 2^{[m]} \to \mathbb{R}_+$ is a coverage function if there exists a universe U of elements with non-negative weights and m sets $A_1, \ldots, A_m \subseteq U$ such that for all $S \subseteq [m]$, $f(S)$ is the total weight of elements in $\cup_{j \in S} A_j$. A coverage function is succinct if $|U|$ is at most a fixed polynomial in m.


© Umang Bhaskar and Gunjan Kumar;
licensed under Creative Commons License CC-BY



The complexity of partial function extension has been studied earlier for other function classes, with a number of important applications shown. For boolean functions, Boros et al. present complexity results for extension to a large number of boolean function classes, as well as results on approximate extension [9]. Pitt and Valiant show a direct relation between the complexity of partial function extension problem and proper PAC-learning. Informally, a class $\mathcal{F}$ of (boolean) functions on $2^{[m]}$ is said to be properly PAC-learnable if for any distribution μ on $2^{[m]}$ and any small enough $\epsilon > 0$, any function $f^* \in \mathcal{F}$ can be learned by a polynomial-time algorithm that returns a function $f \in \mathcal{F}$ with a polynomial number of samples that differs from f^* with probability at most ϵ. Pitt and Valiant show that if partial function extension for a class $\mathcal{F}$ of functions is NP-hard, then the class $\mathcal{F}$ cannot be PAC-learned unless NP = RP [21].[1] They show computational lower bounds for various classes of boolean functions, thereby obtaining lower bounds on the complexity for learning these classes. In this paper, we show lower bounds on partial function extension for coverage functions, which by this relation give lower bounds on proper PAC learning as well. In separate work, we present results on the computational complexity of partial function extension for submodular, subadditive, and convex functions, and show further connections with learning and property testing [5].

Characterizing partial functions extendible to convex functions is widely studied in convex analysis. Here a partial function is given defined on a non-convex set of points, and is required to be extended to a convex function on the convex hull or some other convex domain. Characterizations for extendible partial functions are given in various papers, such as [12, 26]. This finds many applications, including mechanism design [14], decision making under risk [20], and quantum computation [25].

Another example of the ubiquity of partial function extension is in property testing. Given oracle access to a function f, the goal of property testing is to determine by querying the oracle if the function f lies in some class $\mathcal{F}$ of functions of interest, or is far from it, i.e., differs from any function in $\mathcal{F}$ at a large number of points. Partial function extension is a natural step in property testing, since at any time the query algorithm has a partial function consisting of the points queried and the values at those points. If at any time the partial function thus obtained is not extendible to a function in $\mathcal{F}$, the algorithm should reject, and should accept otherwise. Partial function extension is used to give both upper and lower bounds for property testing [5, 23]. Partial function extension is thus a basic problem that finds application in a wide variety of different fields.

Our Contribution

Our input is a partial function $H = \{(T_1, f_1), \ldots, (T_n, f_n)\}$ with $T_i \subseteq [m]$ and $f_i \geq 0$, and the goal is to determine if there exists a coverage function $f : 2^{[m]} \to \mathbb{R}_{\geq 0}$ such that $f(T_i) = f_i$ for all $i \in [n]$. This is the Coverage Extension problem. Throughout the paper we use $[m]$ for the ground set, n for the number of defined sets in the partial function, and $\mathcal{D}$ for the set of defined sets $\{T_1, \ldots, T_n\}$. We also use $d = \max_{i \in [n]} |T_i|$ to denote the maximum size of sets in $\mathcal{D}$, and $F := \sum_{i \in [n]} f_i$.

Our first result shows that Coverage Extension is NP-hard. Interestingly, we show if there exists a coverage function extending the given partial function then there is an extension by a coverage function for which the size of the universe $|U|$ is at most n. This shows that Coverage

[1] Randomized Polynomial (RP) is the class of problems for which a randomized algorithm runs in polynomial time, always answers correctly if the input is a "no" instance, and answers correctly with probability at least 1/2 if the input is a "yes" instance.

Extension is in NP. In contrast, it is known that minimal certificates for non-extendibility may be of exponential size [10]. Also, unlike property testing, this shows that Coverage Extension does not become easier when restricted to succinct coverage functions.

▶ **Theorem 1.** *Coverage Extension is NP-complete.*

For the hardness, we show a reduction from fractional graph colouring, a problem studied in fractional graph theory. Our hardness for extension also shows the following result for proper learning of succinct coverage functions.

▶ **Theorem 2.** *Unless RP = NP, the class of succinct coverage functions cannot be PAC-learned (i.e., cannot be PMAC-learned with approximation factor $\alpha = 1$).*

These are the first hardness results for learning coverage functions based on standard complexity assumptions. Earlier results showed a reduction from learning disjoint DNF formulas to learning coverage functions [13], however as far as we are aware, there are no known lower bounds for learning disjoint DNF formulas. The following theorem is shown in the appendix.

Given the hardness result for Coverage Extension, we study approximation algorithms for two natural optimization versions of the extension problem. In both of these problems, the goal is to determine the distance between the given partial function and the class of coverage functions. Based on the notion of the distance, we study the following two problems.

In *Coverage Approximate Extension*, the goal is to determine minimum value of $\alpha \geq 1$ such that there exists a coverage function $f : 2^{[m]} \to \mathbb{R}_{\geq 0}$ satisfying $f_i \leq f(T_i) \leq \alpha f_i$ for all $i \in [n]$.

In *Coverage Norm Extension*, the goal is to determine the minimum L_1 distance from a coverage function, i.e., minimize $\sum_{i\in[n]} |\epsilon_i|$ where $\epsilon_i = f(T_i) - f_i$ for all $i \in [n]$ for some coverage function f.

The two notions of approximation we study thus roughly correspond to the two prevalent notions of learning real-valued functions. Coverage Approximate Extension corresponds to PMAC learning, where we look for point-wise multiplicative approximations. Coverage Norm Extension corresponds to minimizing the L_1 distance in PAC learning.

Throughout this paper, the minimum value of α in Coverage Approximate Extension will be denoted by α^* and minimum value of $\sum_{i\in[n]} |\epsilon_i|$ in Norm Extension will be denoted by OPT. As both of these problems are generalisations of Coverage Extension, they are NP-hard. We give upper and lower bounds for approximation for both of these problem.

▶ **Theorem 3.** *There is a $\left(\min\{d, m^{2/3}\} \log d\right)$-approximation algorithm for Coverage Approximate Extension. If d is a constant then there is a d-approximation algorithm.*

In Coverage Norm Extension, $OPT = 0$ iff the partial function is extendible and hence no multiplicative approximation is possible for OPT unless P = NP (because of Theorem 1). We hence consider additive approximations for Coverage Norm Extension. An algorithm for Coverage Norm Extension is called an α-approximation algorithm if for all instances (partial functions), the value β returned by the algorithm satisfies $OPT \leq \beta \leq OPT + \alpha$. We show nearly tight upper and lower bounds on the hardness of approximation. As defined before $F = \sum_{i\in[n]} f_i$. Note that an F-approximation algorithm is trivial, since the function $f = 0$ is coverage and satisfies $\sum_{i\in[n]} |f(T_i) - f_i| \leq F$.

▶ **Theorem 4.** *There is a $(1-1/d)F$-approximation algorithm for Coverage Norm Extension. Moreover, a coverage function f can be efficiently computed such that $\sum_{i\in[n]} |f(T_i) - f_i| \leq OPT + (1 - 1/d)F$.*

▶ **Theorem 5.** *It is NP-hard to approximate Coverage Norm Extension by a factor* $\alpha = 2^{poly(n,m)}F^{\delta}$ *for any fixed* $0 \leq \delta < 1$. *This holds even when* $d = 2$.

Our lower bound is roughly based on the equivalence of *validity* and *membership*, where given a convex, compact set K, the validity problem is to determine the optimal value of $c^T x$ given a vector c over all $x \in K$, while the membership problem seeks to determine if a given point x is in K or not. The equivalence of optimization and separation is a widely used tool. The reduction from optimization to separation is particularly useful for, e.g., solving linear programs with exponential constraints. Our work is unusual in both the use of validity and membership rather than optimization and separation, and because of the direction – we use the equivalence to show hardness of the validity problem. We hope that our techniques may be useful in future work as well.

Related Work

We focus here on work related to partial function extension and coverage functions. In a separate paper, we study partial function extension to submodular, subadditive, and convex functions, showing results on the complexity as well as applications to learning and property testing [5]. Previously, Seshadri and Vondrak [23] introduce the problem of extending partial functions to a submodular function, and note its usefulness in analyzing property testing algorithms. For submodular functions, partial function extension is also useful in optimization [24]. The problem of extending a partial function to a convex function is also studied in convex analysis [26, 12]. As mentioned earlier, both characterizing extendible partial functions, and the complexity of partial function extension has been studied for large classes of Boolean functions [9, 21].

Chakrabarty and Huang study property testing for coverage functions [10]. Here, the goal is to determine whether the input function (given by an oracle) is coverage or far from coverage by querying an oracle, where distance is measured by the number of points at which the function must be changed for it to be coverage. They show that succinct coverage functions can be reconstructed with a polynomial number of queries and hence can be efficiently tested. However, they conjecture that testing general coverage functions requires $2^{\Omega(m)}$ queries, and prove this lower bound under a different notion of distance. They present a particular characterization of coverage functions in terms of the W-transform that we use as well.

There has also been interest in sketching and learning coverage functions. Badanidiyuru et al. [1] showed that coverage functions admit a $(1+\epsilon)$-sketch, i.e., given any coverage function, there exists a succinct coverage function (of size polynomial in m and $1/\epsilon$) that approximates the original function within $(1+\epsilon)$ factor with high probability. Feldman and Kothari [13] gave a fully polynomial time algorithm for learning succinct coverage functions in the PMAC model if the distribution is uniform. However, if the distribution is unknown, they show learning coverage functions is as hard as learning polynomial size DNF formulas for which no efficient algorithm is known.

Balkanski et al [3] study whether coverage functions can be optimized from samples. They consider a scenario where random samples $\{(S_i, f(S_i))\}$ of an unknown coverage function f are provided and ask if it is possible to optimize f under a cardinality constraint, i.e., solve $\max_{S:|S|\leq k|} f(S)$. They prove a negative result: no algorithm can achieve approximation ratio better than $2^{\Omega(\sqrt{\log m})}$ with a polynomial number of sampled points.

2 Preliminaries

As earlier, for $m \in \mathbb{Z}_+$, define $[m] := \{1, 2, \ldots, m\}$. A set function f over a ground set $[m]$ is a coverage function if there exists a universe U of elements with non-negative weights and m sets $A_1, ..., A_m \subseteq U$, such that for all $S \subseteq [m]$, $f(S)$ is the total weight of elements in $\cup_{j \in S} A_j$. A coverage function is *succinct* if $|U|$ is at most a fixed polynomial in m.

Chakrabarty and Huang [10] characterize coverage functions in terms of their W-transform, which we use as well. For a set function $f : 2^{[m]} \to \mathbb{R}_{\geq 0}$, the W-transform $w : 2^{[m]} \setminus \emptyset \to \mathbb{R}$ is defined as

$$\forall S \in 2^{[m]} \setminus \emptyset, \quad w(S) = \sum_{T : S \cup T = [m]} (-1)^{|S \cap T|+1} f(T)\,. \tag{1}$$

The set $\{w(S) | S \in 2^{[m]} \setminus \emptyset\}$ is called the set of W-coefficients of f. We can also recover the function f from its W-coefficients.

$$\forall T \subseteq [m], \quad f(T) = \sum_{S \subseteq [m] : S \cap T \neq \emptyset} w(S)\,. \tag{2}$$

If f is a coverage function induced by the universe U and sets $A_1, \ldots, A_m$, then the W-transform $w(S)$ is precisely the weight of the set $\{(\cap_{i \in S} A_i) \setminus \cup_{j \notin S} A_j\}$, and is hence non-negative. The converse is also true. The set $\{S | w(S) > 0\}$ is the called the *support* of the coverage function, and the elements are exactly the elements of the universe U.

▶ **Theorem 6** ([10]). *A set function $f : 2^{[m]} \to \mathbb{R}_{\geq 0}$ is a coverage function iff all of its W-coefficients are non-negative.*

From Theorem 6, given a partial function H, there exists a coverage function f satisfying $f(T_i) = f_i$ for all $i \in [n]$ iff the following linear program is feasible, where the variables are the W-coefficients $w(S)$ for all $S \in 2^{[m]} \setminus \emptyset$.

$$\text{Extension-P:} \qquad \sum_{S : S \cap T_i \neq \emptyset} w(S) = f_i \quad \forall i \in [n]\,, \qquad w(S) \geq 0 \quad \forall S \in 2^{[m]} \setminus \emptyset.$$

All missing proofs are in the appendix.

3 Coverage Extension and PAC-Learning

Our first observation is that there is a polynomial-sized certificate of extendibility to a coverage function. This is obtained by observing that at a vertex of the feasible set in Extension-P, at most n of the variables are non-zero. It is interesting to compare this with Chakrabarty and Huang [10], who give an example to show that minimal certificates of nonextendibility may be of exponential size.

▶ **Proposition 7.** *If a partial function is extendible to a coverage function, then it is also extendible to a coverage function with support size $\leq n$. Hence, Coverage Extension is in NP.*

We show the NP-hardness of Coverage Extension by reduction from *fractional chromatic number*, defined as follows. Given a graph $G = (V, E)$, a set $I \subseteq V$ is called an *independent set* if no two vertices in I are adjacent. Let $\mathcal{I}$ be the set of all independent sets. The fractional chromatic number $\chi^*(G)$ of a graph G is the optimal value of the following linear program.

$$\chi^*(G) := \left\{ \min \sum_{I\in\mathcal{I}} x_I : \sum_{I\in\mathcal{I}:v\in I} x_I \geq 1 \quad \forall v \in V(G), 0 \leq x_I \leq 1 \quad \forall I \in \mathcal{I} \right\}$$

Note that if $x_I \in \{0,1\}$ then the optimal value is just the chromatic number of the graph.[2]

▶ **Theorem 8** ([15]). *For graph $G = (V,E)$, there exist nonnegative weights $\{x_I\}_{I\in\mathcal{I}}$ on independent sets such that $\chi^*(G) = \sum_{I\in\mathcal{I}} x_I$ and $\sum_{I\in\mathcal{I}:v\in I} x_I = 1 \quad \forall v \in V$.*

▶ **Corollary 9.** *For graph $G = (V,E)$ and for any value of t such that $\chi^*(G) \leq t \leq |V|$, there exist nonnegative weights $\{z_I\}_{I\in\mathcal{I}}$ on independent sets such that $\sum_{I\in\mathcal{I}} z_I = t$ and $\sum_{I\in\mathcal{I}:v\in I} z_I = 1 \quad \forall v \in V$.*

▶ **Theorem 10** ([17]). *Given graph $G = (V,E)$ and $1 \leq k \leq |V|$, it is NP-hard to determine if $\chi^*(G) \leq k$.*

We now show the NP-hardness of Coverage Extension.

Proof of Theorem 1. Since membership in NP was shown earlier, we give the reduction from fractional chromatic number. The input is a graph $G = (V,E)$ and $1 \leq k \leq |V|$.

We identify $[n']$ with the set of vertices V, and therefore $E(G) \subseteq \{\{i,j\}|i,j \in [n']\}$, and any set $S \subseteq [n']$ can be viewed as a set of vertices. The partial function construction is as follows. The ground set is $[n']$ and therefore $m = n'$. The partial function is defined at all vertices, all edges, and the set consisting of all vertices. Hence $\mathcal{D}$, the set of defined points for the partial function, is $\{\{i\}|i \in [n']\} \cup E(G) \cup \{[n']\}$ and $|\mathcal{D}| = n' + |E(G)| + 1$. The value of the partial function h at these defined sets is given by

$$h(S) = \begin{cases} 1 & \text{if} \quad S = \{i\},\, i \in [n']\,, \\ 2 & \text{if} \quad S \in E(G)\,, \\ k & \text{if} \quad S = \{[n']\}\,. \end{cases}$$

Intuitively, the function $h(S)$ can be interpreted as the (fractional) number of colours used to colour the subset S.

We claim that $\chi^*(G) \leq k$ iff the above partial function is extendible. Suppose $\chi^*(G) \leq k$. Therefore by Corollary 9, there exist nonnegative weights $\{x_I\}_{I\in\mathcal{I}}$ such that $\sum_{I\in\mathcal{I}} x_I = k$ and $\sum_{I\in\mathcal{I}:v\in I} x_I = 1 \quad \forall v \in V(G)$. For all $S \in 2^{[m]} \setminus \emptyset$, define the function $w(S)$ as x_S if $S \in \mathcal{I}$ and 0 otherwise. Since $w(S) \geq 0$, this defines the W-transform for a coverage function g. We have, for any $i \in [n']$,

$$g(\{i\}) = \sum_{S:S\cap\{i\}\neq\emptyset} w(S) = \sum_{I\in\mathcal{I}:i\in I} x_I = 1\,,$$

for any $\{i,j\} \in E(G)$,

$$g(\{i,j\}) = \sum_{S:S\cap\{i,j\}\neq\emptyset} w(S) = \sum_{I\in\mathcal{I}:i\in I} x_I + \sum_{I\in\mathcal{I}:j\in I} x_I = 2$$

as no independent set I can contain both i and j; and finally $g(\{[n']\}) = \sum_{S:S\cap\{[n']\}\neq\emptyset} w(S) = \sum_{I\in\mathcal{I}} x_I = k$. Therefore g is an extension of the above partial function h.

[2] The chromatic number of a graph is the minimum number of colours required to colour the vertices so that no two adjacent vertices get the same colour.

Now suppose there is an extension, i.e., there exists $w(S) \geq 0$ for all $S \in 2^{[m]} \setminus \emptyset$ such that for any $i \in [n']$, $\sum_{S:S\cap\{i\}\neq\emptyset} w(S) = 1$; for any $\{i,j\} \in E(G), \sum_{S:S\cap\{i,j\}\neq\emptyset} w(S) = 2$; and finally $\sum_{S:S\cap\{[n']\}\neq\emptyset} w(S) = k$. For any $\{i,j\} \in E(G)$, we have

$$\sum_{S:S\cap\{i,j\}\neq\emptyset} w(S) = \sum_{S:S\cap\{i\}\neq\emptyset} w(S) + \sum_{S:S\cap\{j\}\neq\emptyset} w(S) - \sum_{S:S\supseteq\{i,j\}} w(S)\,.$$

Therefore,$\sum_{S:S\supseteq\{i,j\}} w(S) = 0$, i.e., if $w(S) > 0$ then S must be an independent set. It now follows that $\chi^*(G) \leq \sum_{S:S\cap\{[n']\}\neq\emptyset} w(S) = k$. ◀

Proper PAC-learning of Coverage functions

We now prove Theorem 2. We first recall the definition of proper PAC-learning.

▶ **Definition 11** ([2]). *An algorithm $\mathcal{A}$ properly PAC-learns a family of functions $\mathcal{F}$, if for* any *distribution μ (on $2^{[m]}$) and* any *target function $f^* \in \mathcal{F}$, and for* any *sufficiently small $\epsilon, \delta > 0$:*

1. *$\mathcal{A}$ takes the sequence $\{(S_i, f^*(S_i))\}_{1\leq i\leq l}$ as input where l is poly$(m, 1/\delta, 1/\epsilon)$ and the sequence $\{S_i\}_{1\leq i\leq l}$ is drawn i.i.d. from the distribution μ,*
2. *$\mathcal{A}$ runs in poly$(m, 1/\delta, 1/\epsilon)$ time, and*
3. *$\mathcal{A}$ returns a function $f : 2^{[m]} \to \mathbb{R} \in \mathcal{F}$ such that*

$$Pr_{S_1,\ldots,S_l\sim\mu}\left[Pr_{S\sim\mu}[f(S) = f^*(S)] \geq 1-\epsilon\right] \geq 1-\delta$$

We use the reconstruction algorithm for coverage functions given by Chakrabarty and Huang [10] in our proof. Given a coverage function f as an input, this reconstruction algorithm terminates in $O(m\,s)$ steps where s is the support size of f, i.e., the number of non-zero W-coefficients of f, and returns these non-zero W-coefficients.

Recall the reduction from fractional chromatic number to Coverage Extension (Theorem 1). Given an instance of fractional chromatic number (graph $G = (V, E)$ and rational k' with $|V| = n'$), the instance of Coverage Extension is a set of defined points $\mathcal{D} = \{\{i\}|i \in [n']\} \cup E(G) \cup \{[n']\}$ and a function h on $\mathcal{D}$. Let $k = |\mathcal{D}| = |V| + |E| + 1$. From Theorem 1 and Proposition 7, $\chi^*(G) \leq k'$ iff h is extendible to a coverage function with support size at most k.

Let $\mathcal{F}$ be a family of coverage functions with support size at most k. Let $\epsilon = 1/k^3$ (and hence $\epsilon < 1/|\mathcal{D}|$) and μ be a uniform distribution over $\{(S, h(S))|S \in \mathcal{D}\}$. Now suppose a (randomized) algorithm A properly PAC-learns $\mathcal{F}$. We will show that in this case, we can determine efficiently if the partial function is extendible to a coverage function, and hence RP = NP.

Suppose the algorithm A returns a function g. If the partial function is extendible then there exists a function in $\mathcal{F}$ that has the same value on samples seen by A. Therefore, if the partial function is extendible then $g(S)$ must be equal to $h(S)$ for all $S \in \mathcal{D}$ (since $\epsilon < 1/|\mathcal{D}|$ and A must satisfy$Pr_{S\sim D^*}[f(S) = f^*(S)] \geq 1-\epsilon$). We run the reconstruction algorithm on input g. If the partial function is extendible then g must be in $\mathcal{F}$ and hence the reconstruction algorithm must terminate in $O(mk)$ steps. Further, if $\{w(S)\}_{S\in\mathcal{S}}$ is the output of the algorithm then (i) $w(S) > 0$ for all $S \in \mathcal{S}$, (ii) $|\mathcal{S}| \leq k$ (iii) the coverage function f' given by the W-coefficients $w'(S) = w(S)$ if $S \in \mathcal{S}$ and 0 otherwise is an extension of the partial function h. Condition (iii) should hold because f' must be the same as g which we have shown earlier is an extension of h.

The converse is also true – if the reconstruction algorithm terminates and (i), (ii), (iii) hold then clearly h is extendible (by f'). Since all the steps require polynomial time to check, we can efficiently determine if the partial function is extendible.

4 Coverage Approximate Extension

We now build the framework for Theorem 3. We start with the following lemma.

▶ **Lemma 12.** *Given a partial function H and $\alpha \geq 1$, there is no coverage function f satisfying $f_i \leq f(T_i) \leq \alpha f_i$ for all $i \in [n]$ iff the following program, with variables l_i for all $i \in [n]$ is feasible:*

$$-\alpha \sum_{i:l_i<0} f_i l_i < \sum_{i:l_i>0} f_i l_i \tag{3}$$

$$\sum_{i:S\cap T_i\neq\emptyset} l_i \leq 0 \quad \forall S \subseteq [m] \tag{4}$$

Thus the optimal approximation ratio α^* is the minimum value of α for which (3) and (4) are not feasible together.

A natural representation of the partial function $H = \{(T_1, f_1), \ldots, (T_n, f_n)\}$ is as a weighted bipartite graph $H = (A \cup [m], E)$ with $|A| = n$, and an edge between $a_i \in A$ and $j \in [m]$ if the set T_i contains element $j \in [m]$. Each vertex $a_i \in A$ also has weight f_i. Then $d = \max_i |T_i|$ is the maximum degree of any vertex in A. For the remainder of this section, we will use this representation of partial functions.

We use the following notation given a bipartite graph $H = (A \cup [m], E)$. For any $S \subseteq [m]$, let $N(S) = \{v \in A : (v, j) \in E \quad \text{for some} \quad j \in S\}$ be the set of neighbours of set S. Similarly for set $R \subseteq A$, $N(R) = \{j \in [m] : (v, j) \in E \quad \text{for some} \quad v \in R\}$ be the set of neighbours of set R. For any vertex v in H, we use $N(v)$ for $N(\{v\})$. In this bipartite graph representation, the inequality (4) is equivalent to $\sum_{i\in N(S)} l_i \leq 0$ for all $S \subseteq [m]$.

We now define a parameter κ called the *replacement ratio* for a partial function H.

▶ **Definition 13.** *Let $H = (A \cup [m], E)$ be a bipartite graph with weights f_v on each $v \in A$. For $v \in A$, let $\mathcal{F}_v = \{R \subseteq A \setminus \{v\} \mid N(R) \supseteq N(v)\}$ be the set of all subsets of $A \setminus \{v\}$ that cover all the neighbours of v. We call each $R \in \mathcal{F}_v$ a* replacement *for v. The* replacement ratio *κ is then the minimum of $\frac{\sum_{w\in R} f_w}{f_v}$ over all vertices $v \in A$ and replacements $R \in \mathcal{F}_v$.*

The proof of the upper bound in Theorem 3 will follow from the bounds on α^* shown in Lemma 14, 16 and 18.

▶ **Lemma 14.** *For any partial function H, $\alpha^* \geq \frac{1}{\kappa}$.*

Proof. By definition of κ, there exists a vertex $v \in A$ and a replacement R for v such that $\sum_{w\in R} f_w = c f_v$. Note that setting $l_w = -1 \quad \forall w \in R$, $l_v = 1$ and all other l_w's to be zero results in feasibility of the inequalities $\sum_{w\in N(S)} l_w \leq 0$ for all $S \in 2^{[m]} \setminus \emptyset$. From the definition of α^* and Lemma 12, $\alpha^* \sum_{w\in R} f_w \geq f_v$, and hence $\alpha^* \geq 1/\kappa$. ◀

Let $\beta = \frac{\min\{d, m^{2/3}\}}{\kappa}$. Given values $\{l_v\}_{v\in A}$ on the vertices in A such that $\sum_{v\in N(S)} l_v \leq 0$ for all $S \subseteq [m]$, we will show that $\beta \sum_{v:l_v<0} f_v l_v \geq \sum_{v:l_v>0} f_v l_v$ and hence $\alpha^* \leq \beta$. If $l_v = 0$ for any vertex, we simply ignore such a vertex, since it does not affect either (4) or (3).

By scaling, we can assume that $l_v \in \mathbb{Z}$ for all $v \in A$. At some point, we will use Hall's theorem to show a perfect matching. To simplify exposition, we replace each $v \in A$ with $|l_v|$ identical copies, each of which is adjacent to the same vertices as v. Each such new vertex v' has $l_{v'} = 1$ if $l_v > 0$ and $l_{v'} = -1$ if $l_v < 0$, and $f_{v'} = f_v$. Let the new bipartite graph be $H' = (A' \cup [m], E')$. It is easy to check that in the new bipartite graph, the degree of vertices in A' and the values κ, $\sum_{v\in A':l_v>0} f_v l_v$, $\sum_{v\in A':l_v<0} f_v l_v$ and $\sum_{v\in N(S)} l_v$ remain unchanged for all $S \subseteq [m]$.

Let $\mathcal{N} = \{v \in A' | l_v = -1\}$ and $\mathcal{P} = \{v \in A' | l_v = 1\}$, and let E^- be the set of edges with one end-point in $\mathcal{N}$, while E^+ are the edges with one end-point in $\mathcal{P}$. For any $S \subseteq [m]$, let $N^-(S) = N(S) \cap \mathcal{N}$ and $N^+(S) = N(S) \cap \mathcal{P}$ (so $N(S) = N^+(S) \cup N^-(S)$). Finally, define $E^+(S)$ ($E^-(S)$) as the set of edges with one end-point in S and the other end-point in $\mathcal{P}$ ($\mathcal{P}$). If $S = \{j\}$, we abuse notation slightly and use $N^-(j)$, $N^+(j)$, $E^-(j)$ and $E^+(j)$. Note that $|N^-(S)| \geq |N^+(S)|$ for all $S \subseteq [m]$ in H', since in H, $\sum_{v \in N(S)} l_v \leq 0$ for all $S \subseteq [m]$. Our goal is to show $\beta \sum_{v \in \mathcal{N}} f_v \geq \sum_{v \in \mathcal{P}} f_v$.

▶ **Lemma 15.** *Suppose for some $\beta' \geq 1$, $\beta'|N^-(S)| \geq \sum_{j \in S} |N^+(j)|$ for all $S \subseteq [m]$. Then for each vertex $v \in \mathcal{P}$, there exists a replacement $F_v \subseteq \mathcal{N}$ such that each vertex in $\mathcal{N}$ is contained in F_v for at most β' vertices $v \in \mathcal{P}$. Hence, $\beta' \sum_{v \in \mathcal{N}} f_v \geq \kappa \sum_{v \in \mathcal{P}} f_v$ and so $\alpha^* \leq \frac{\beta'}{\kappa}$.*

Proof. By Hall's theorem, there exists a set of edges $M \subseteq E^-$ such that (i) the degree in M of each vertex $j \in [m]$ is at least $|N^+(j)|$, and (ii) the degree in M of each vertex $v \in \mathcal{N}$ is at most β'. Because of (i), for each $j \in [m]$ there is an injection h_j from edges in $E^+(j)$ to edges in $E^-(j) \cap M$, i.e., each edge in $E^+(j)$ maps to a distinct edge in $E^-(j) \cap M$. Now for a vertex $v \in \mathcal{P}$, consider a neighbouring vertex $j \in N(v)$. Each such edge (v, j) is in $E^+(j)$, and is hence mapped by h_j to an edge in $E^-(j) \cap M$. Let F_v be the end-points in $\mathcal{N}$ of these mapped edges. That is, $w \in F_v$ iff there exists $j \in N(v)$ such that $(w, j) = h_j(v, j)$. Then F_v is a replacement for v, and hence, $\sum_{w \in F_v} f_w \geq \kappa f_v$. Further, because of (ii), and since each h_j is an injection, each vertex in $\mathcal{N}$ is contained in F_v for at most β' vertices $v \in \mathcal{P}$. Then summing the inequality $\sum_{w \in F_v} f_w \geq \kappa f_v$ over all $v \in \mathcal{P}$, we get that $\beta' \sum_{v \in \mathcal{N}} f_v \geq \kappa \sum_{v \in \mathcal{P}} f_v$ as required. ◀

▶ **Lemma 16.** *For any partial function H, $\alpha^* \leq \frac{d}{\kappa}$.*

Proof. Fix $S \subseteq [m]$. Since $|N^-(j)| \geq |N^+(j)|$ for all $j \in [m]$, $\sum_{j \in S} |N^-(j)| \geq \sum_{j \in S} |N^+(j)|$, and since d is the maximum degree of any vertex in A', $d|N^-(S)| \geq \sum_{j \in S} |N^-(j)|$. The proof follows from Lemma 15. ◀

If we can show $m^{2/3}|N^-(S)| \geq \sum_{j \in S} |N^+(j)|$ for all $S \subseteq [m]$ then by Lemma 15, $\alpha^* \leq \frac{m^{2/3}}{\kappa}$. Unfortunately this may not be true. Let $\mathcal{N} = \{v_1\}, \mathcal{P} = \{v_2\}, E^- = \{(v_1, j) | j \in [m]\}$ and $E^+ = \{(v_2, j) | j \in [m]\}$. Note that $\sum_{j \in [m]} |N^+(j)| = m$ whereas $|N^-([m])| = 1$. Notice that in this bad example, the bipartite graph contains a 4-cycle v_1, j_1, v_2, j_2, v_1 where $v_1 \in \mathcal{N}$ and $v_2 \in \mathcal{P}$. We now define a subgraph called a *diamond* which generalises such a 4-cycle. A diamond (v_p, v_n, J) of size k is a subgraph of H' where $v_p \in \mathcal{P}, v_n \in \mathcal{N}, J \subseteq [m]$ ($|J| = k$) such that for all $j \in J$, both (v_p, j) and (v_n, j) are contained in E'. Note that a 4-cycle is a diamond of size two (and the bad example considered above is a diamond of size m).

Let $k_{max} = m^a$ ($0 \leq a \leq 1$) be the maximum size of any diamond in H'.

▶ **Lemma 17.** *For all $S \subseteq [m]$, $m^{\frac{1+a}{2}}|N^-(S)| \geq \sum_{j \in S} |N^+(j)|$, where m^a is the size of the largest diamond in H'.*

Proof. Recall that for all $j \in [m]$, $|N^+(j)| \leq |N^-(j)|$, hence there is an injection h_j from $N^+(j)$ to $N^-(j)$, i.e, h_j maps each vertex in $N^+(j)$ to a unique vertex in $N^-(j)$. Fix $S \subseteq [m]$ and vertex $v \in \mathcal{P}$, and let $S_v := N(v) \cap S$ be the neighbourhood of v in S. We will consider $N^+(S_v)$ and $N^-(S_v)$, the negative and positive neighbourhoods of S_v. Note that since all vertices in S_v are adjacent to $v \in \mathcal{P}$, a vertex in $N^-(S_v)$ is adjacent to at most m^a vertices in S_v, by definition of a. Thus for a vertex $v' \in N^-(S_v)$, there are at most m^a different vertices $j \in S_v$ for which h_j maps a vertex in $N^+(j)$ to v', and hence $m^a|N^-(S_v)| \geq \sum_{j \in S_v} |N^+(j)|$.

Now if there is a vertex $v \in \mathcal{P}$ such that $m^{\frac{1-a}{2}} \sum_{j \in S_v} |N^+(j)| \geq \sum_{j \in S} |N^+(j)|$ then we are done, since

$$|N^-(S)| \geq |N^-(S_v)| \geq \frac{\sum_{j \in S_v} |N^+(j)|}{m^a} \geq \frac{\sum_{j \in S} |N^+(j)|}{m^{\frac{1+a}{2}}}.$$

So assume that for all $v \in \mathcal{P}$, $\sum_{j \in S_v} |N^+(j)| \leq \frac{\sum_{j \in S} |N^+(j)|}{m^{\frac{1-a}{2}}}$. In this case, note that by reversing the order of summation,

$$\sum_{j \in S} |N^+(j)|^2 = \sum_{j \in S} \sum_{v \in N^+(j)} |N^+(j)| = \sum_{v \in N^+(S)} \sum_{j \in S_v} |N^+(j)| \leq |N^+(S)| \frac{\sum_{j \in S} |N^+(j)|}{m^{\frac{1-a}{2}}}.$$

Therefore, using the above inequality for $|N^+(S)|$,

$$|N^-(S)| \geq |N^+(S)| \geq m^{\frac{1-a}{2}} \frac{\sum_{j \in S} |N^+(j)|^2}{\sum_{j \in S} |N^+(j)|} \geq \frac{m^{\frac{1-a}{2}}}{|S|} \frac{\left(\sum_{j \in S} |N^+(j)\right)^2}{\sum_{j \in S} |N^+(j)|} \geq \frac{\sum_{j \in S} |N^+(j)|}{m^{\frac{1+a}{2}}}$$

as required by the lemma. The third inequality follows from Cauchy-Schwarz. ◀

From Lemmas 15 and 17, if $a \leq 1/3$ then $\alpha^* \leq \frac{m^{2/3}}{\kappa}$. Next we show this is true in general.

▶ **Lemma 18.** *For any partial function H, $\alpha^* \leq \frac{m^{2/3}}{\kappa}$.*

Proof. If $k_{max} \leq m^{1/3}$ then by Lemma 17 and 15, $\alpha^* \leq \frac{m^{2/3}}{\kappa}$. So we assume $k_{max} > m^{1/3}$. In this case, we pick a diamond (v_p, v_n, J) of size $> m^{1/3}$. We remove, for all $j \in J$, the edges (v_p, j) and (v_n, j). We repeat the above procedure (in the new graph) until we are left with a bipartite graph where all diamonds are of size at most $m^{1/3}$. Note that if a diamond (v_p, v_n, J) of size k is removed then the degree of v_n decreases by k. Hence, for a fixed vertex v_n, number of removed diamonds is at most $m^{2/3}$ (as at any step we remove diamonds of size at least $m^{1/3}$). It is easy to see that after every step, $|N^-(S)| \geq |N^+(S)|$ (for all $S \in 2^{[m]} \setminus \emptyset$) still holds in the bipartite graph. Let H^* be the bipartite graph at the end (all diamonds of size at most $m^{1/3}$).

Note that we do not remove any vertex in the above procedure. Fix vertex $v \in \mathcal{P}$. By Lemmas 17 and 15 with $a = 1/3$, there exists $F_v \subseteq \mathcal{N}$ such that F_v covers all neighbours of v in H^* and each vertex in $\mathcal{N}$ is contained in F_v for at most $m^{2/3}$ vertices $v \in \mathcal{P}$. Since we have removed edges, F_v may not cover all the neighbours of v in H'. Let $v^1, \ldots, v^s \in \mathcal{N}$ be the set of all vertices such that for each $i \in [s]$, a diamond (v, v^i, J^i) was removed in a removal step. Clearly $\{v^1, \ldots, v^s\} \cup F_v$ cover all the neighbour of v in H'. Therefore, we have $\sum_{i=1}^{s} f_{v^i} + \sum_{w \in F_v} f_w \geq \kappa f_v$. Since any v^i $(1 \leq i \leq s)$ is a part of at most $m^{2/3}$ removed diamonds and each vertex in $\mathcal{N}$ is contained in F_v for at most $m^{2/3}$ vertices $v \in \mathcal{P}$, summing the above inequality for each $v \in \mathcal{P}$, we get $m^{2/3} \sum_{v \in \mathcal{N}} f_v \geq \kappa \sum_{v \in \mathcal{P}} f_v$ as required. ◀

It follows from Lemmas 14, 16 and 18 that an algorithm that returns $\frac{\min\{d, m^{2/3}\}}{\kappa}$ is a $\min\{d, m^{2/3}\}$-approximation algorithm. However, computing κ corresponds to solving a general set cover instance, and is NP-hard. This connection however allows us to show the following result.

▶ **Lemma 19.** *Given a partial function, the replacement ratio κ can be efficiently approximated by κ' such that $\kappa \leq \kappa' \leq \kappa \log d$. If d is a constant, the replacement ratio κ can be determined efficiently.*

This completes the proof of the upper bound in Theorem 3. In the full version of the paper, we show there exist partial functions such that (i) $\alpha^* = 1/\kappa$ for any value of κ, and (ii) with $d = \sqrt{m}$ and $\alpha^* = \Omega(\frac{\sqrt{m}}{\kappa \log m})$. The bounds shown on α^* thus cannot be substantially improved.

5 Coverage Norm Extension

From Theorem 6, the Norm Extension problem can be stated as the convex program Norm-P. It can be equivalently transformed to a linear program whose dual is Norm-D.

Norm-P: $\min \sum_{i=1}^{n} |\epsilon_i|$

$$\sum_{S: S \cap T_i \neq \emptyset} w(S) = f_i + \epsilon_i \quad \forall i \in [n]$$

$$w(S) \geq 0 \quad \forall S \in 2^{[m]} \setminus \emptyset$$

Norm-D: $\max \sum_{i=1}^{n} f_i y_i$

$$\sum_{i: S \cap T_i \neq \emptyset} y_i \leq 0 \quad \forall S \in 2^{[m]} \setminus \emptyset \tag{5}$$

$$-1 \leq y_i \leq 1 \quad \forall i \in [n] \tag{6}$$

Both Norm-P and Norm-D are clearly feasible. We use OPT for the optimal value of Norm-P (and Norm-D). As stated earlier, no multiplicative approximation is possible for OPT unless P = NP. Therefore, we consider additive approximations for Norm Extension.

An algorithm for Norm Extension is called an α-approximation algorithm if for all instances (partial functions), the value β returned by the algorithm satisfies $OPT \leq \beta \leq OPT + \alpha$. First we prove our upper bound in Theorem 4. Recall that $d = \max_{i \in [n]} |T_i|$ and $F = \sum_{i \in [n]} f_i$. As noted earlier, the function $f(\cdot) = 0$ is trivially an F-approximation algorithm for Norm Extension, since $\sum_{i \in [n]} |f(T_i) - f_i| = F$.

Proof of Theorem 4. Consider the linear programs obtained by restricting Norm-P to variables $w(S)$ for $S \in [m]$, and similarly restricting the constraints (5) in Norm-D to sets $S \in [m]$ only. They are clearly the primal and dual of each other. The optimal values of these modified problems (say OPT^R, w^R and y^R) can be computed in polynomial time. We will show that $OPT \leq OPT^R \leq OPT + (1 - 1/d)F$ for the proof of the theorem. The first inequality is obvious, since OPT^R is the optimal solution to a relaxed (dual) linear program.

For the second inequality, define $y^A = (y_1^A, \ldots, y_n^A)$ as the vector such that for all $i \in [n]$, $y_i^A = y_i^R$ if $y_i^R \leq 0$ and y_i^R/d otherwise. Then note that

$$OPT^R = \sum_{i \in [n]} f_i y_i^R = \sum_{i \in [n]} f_i y_i^A + (1 - 1/d) \sum_{i: y_i^R \geq 0} f_i y_i^R \leq \sum_{i \in [n]} f_i y_i^A + (1 - 1/d)F\,, \tag{7}$$

where the last inequality is because each $y_i^R \leq 1$. We now show that y^A is a feasible solution for Norm-D, and hence $\sum_{i \in [n]} f_i y_i^A \leq OPT$. Together with (7) this completes the proof.

Clearly y^A satisfies the constraints (6). We will show that y^A also satisfies the constraints (5) for all $S \in 2^{[m]} \setminus \emptyset$. Consider any $S \in 2^{[m]} \setminus \emptyset$. Let $P = \{i \in [n] | S \cap T_i \neq \emptyset \quad \text{and} \quad y_i^R > 0\}$ and $N = \{i \in [n] | S \cap T_i \neq \emptyset \quad \text{and} \quad y_i^R \leq 0\}$. Thus $P \cup N$ are all sets in $\mathcal{D}$ that have nonempty intersection with S. We have for any $j \in S$ that $\sum_{i: j \in T_i} y_i^R \leq 0$. Summing these inequalities over $j \in S$, we obtain $\sum_{i \in P \cup N} |T_i \cap S| y_i^R \leq 0$. Thus $\sum_{i \in P} y_i^R + d \sum_{i \in N} y_i^R \leq 0$. From the definition of y_i^A, we get $\sum_{i: S \cap T_i \neq \emptyset} y_i^A \leq 0$, as required. ◀

We now prove the lower bound in Theorem 4. We start with an outline of the proof. In a nutshell, the proof shows the following reductions (for brevity, WM stands for Weak Membership and WV for Weak Validity):

$$\text{Densest-Cut} \leq_p \text{Cut WM} \leq_p \text{Span WM} \equiv \text{Coverage WM} \leq_p \text{Coverage WV} \leq_p \text{Norm Extension}\,.$$

Given a graph $G = (V, E)$ and a positive rational M, the *Densest-Cut* problem asks if there is a cut $S \subset V$ such that $\frac{|\delta(S)|}{|S|\,|V \setminus S|} > M$. The Densest-Cut problem is known to be NP-hard [7], and ultimately we reduce the Densest-Cut problem to the problem of approximating the optimal value for Norm-P. We formally define the other problems later. However, to show this reduction, we need to utilize the equivalence of optimization (or validity) over a polytope and membership in the polytope. Typically optimization algorithms use the equivalence of optimization and separation to show upper bounds, e.g., that a linear program with an exponential number of constraints can be optimized. Our work is unique in that we use the less-utilized equivalence of validity and membership; and secondly, we use it to show hardness. In fact, since we are looking for hardness of approximation algorithms, our work is complicated further by the need to use *weak* versions of this equivalence.

Given a convex and compact set K and a vector c, the *Strong Validity* problem, given a vector c, is to find the maximum value of $c^T x$ such that $x \in K$ (the x which obtains this maximum is not required). In the *Strong Membership* problem, the goal is to determine if a given vector y is in K or not. The *Weak Validity* and *Weak Membership* problems are weaker versions of the Strong Validity and Strong Membership problems respectively, formally defined later. Then Theorem 4.4.4 in [16] says that for a convex and compact body K, there is an oracle polynomial time reduction from the Weak Membership problem for K to the Weak Validity problem for K.

To formally state Theorem 4.4.4 from [16], which will form the basis of our reduction, we need the following notations and definitions.

We use $||.||$ for the Euclidean norm. Let $K \subseteq \mathbb{R}^{n'}$ be a convex and compact set. A ball of radius $\epsilon > 0$ around K is defined as

$$S(K, \epsilon) := \{x \in \mathbb{R}^{n'} \,|\, ||x - y|| \leq \epsilon \text{ for some } y \text{ in } K\}\,.$$

Thus, for $x \in \mathbb{R}^{n'}$, $S(x, \epsilon)$ is the ball of radius ϵ around x. The interior ϵ-ball of K is defined as

$$S(K, -\epsilon) := \{x \in K | S(x, \epsilon) \subseteq K\}$$

Thus $S(K, -\epsilon)$ can be seen as points deep inside K.

▶ **Definition 20** ([16]). *Given a vector $c \in \mathbb{Q}^{n'}$, a rational number γ and a rational number $\epsilon > 0$, the Weak Validity problem is to assert either (1) $c^T x \leq \gamma + \epsilon$ for all $x \in S(K, -\epsilon)$, or (2) $c^T x \geq \gamma - \epsilon$ for some $x \in S(K, \epsilon)$. Note that the vector x satisfying the second inequality is not required.*

▶ **Definition 21** ([16]). *Given a vector $y \in \mathbb{R}^{n'}$ and $\delta > 0$, the Weak Membership problem is to assert either (1) $y \in S(K, \delta)$, or (2) $y \notin S(K, -\delta)$.*

Intuitively, in the Weak Membership problem, it is required to distinguish between the cases when the given point y is far from the polyhedron K (in which case, the algorithm should return $y \notin S(K, -\delta)$) and y is deep inside K (which case the algorithm should return $y \in S(K, \delta)$). If y is near the boundary of K, then either output can be returned. Our reduction crucially uses the following result.

▶ **Theorem 22** (Theorem 4.4.4 of [16]). *Given a weak validity oracle for $K \subseteq \mathbb{R}^{n'}$ that runs in polynomial time and a positive R such that $K \subseteq S(0,R)$, the Weak Membership problem for the polyhedron K can be solved in polynomial time.*

For our problem K is the polytope of linear program Norm-D.

$$K := \left\{ y \in \mathbb{R}^n : \sum_{i:S\cap T_i \neq \emptyset} y_i \leq 0 \quad \forall S \subseteq [m], \quad ||y||_\infty \leq 1 \right\}. \tag{8}$$

Coverage WM $\leq_p$ Coverage WV $\leq_p$ Coverage Norm Extension

Coverage Weak Membership is the Weak Membership problem for polytope K (8). Given a set $\mathcal{D} = \{T_1, \ldots, T_n\}$ (where $T_i \subseteq [m]$) with weights $\hat{y}_i$ ($\hat{y}_i \in \mathbb{R}$) associated with T_i for all $i \in [n]$ and a $\delta > 0$, the goal in this problem is to assert either $(\hat{y}_1, \ldots, \hat{y}_n) \in S(K, \delta)$ or $(\hat{y}_1, \ldots, \hat{y}_n) \notin S(K, -\delta)$.

Note that Coverage Norm Extension is the Strong Validity problem for K with $c_i = f_i$. We show the following lemma (Coverage WV $\leq_p$ Coverage Norm Extension).

▶ **Lemma 23.** *If there is an $\alpha = 2^{poly(n,m)}F^\delta$ efficient approximation algorithm (for any fixed $0 \leq \delta < 1$) for Coverage Norm Extension then there is an efficient algorithm for Weak Validity problem for K.*

Theorem 22 immediately gives Coverage WM $\leq_p$ Coverage WV.

Span WM $\equiv$ Coverage WM

In fact, we show that Coverage Weak Membership is NP-hard even for the case when $|T_i| = 2$ for all $i \in [n]$.[3] The restriction $|T_i| = 2$ gives us a graphical representation of the membership problems. We first introduce some notations, which will be used in the remainder. Given a weighted graph $G = (V, E)$ and a set $S \subseteq V$, the span $E^+_G(S)$ and cut $\delta_G(S)$ of set S are the set of edges with at least one endpoint and exactly one endpoint in S respectively. We use $w(E^+_G(S)), w(\delta_G(S))$ and $w(E_G(S))$ for the sum of weight of edges with at least one endpoint, exactly one endpoint and both endpoints in S respectively. If the set S is a single vertex v then we use v instead of $\{v\}$. If the graph G is understood from the context we drop the subscript G.

Given a set $\mathcal{D} = \{T_1, \ldots, T_n\}$ ($T_i \subseteq [m]$) with the property that $|T_i| = 2$ for all $i \in [n]$, we construct a weighted graph $G = (V, E)$ as follows: vertex set $V = [m]$ and $\{i, j\} \in E$ ($i, j \in [m]$) iff there exists a $T_k \in \mathcal{D}$ such that $T_k = \{i, j\}$. The weight $\hat{y}_k$ associated with $T_k = \{i, j\}$ is now associated to the edge $\{i, j\}$. Now the constraint $\sum_{i:T_i\cap S\neq\emptyset} y_i \leq 0$ (in the polyhedron K) translates to $\sum_{e\in E^+(S)} y_e \leq 0$ for all $S \subseteq V$. Thus Coverage-Weak-Membership for $|T_i| = 2$ case is equivalent to following problem, which we call *Span Weak Membership*.

Given a weighted graph $G = (V, E)$ with weights $\hat{y}_e$ on the edges and $\delta > 0$, assert either $\hat{y} = (\hat{y}_e)_{e\in E}$ is in $S(K_s, \delta)$ or $\hat{y}$ is not in $S(K_s, -\delta)$, where

$$K_s = \left\{ \sum_{e\in E^+(S)} y_e \leq 0 \quad \forall S \subseteq V, ||y||_\infty \leq 1 \right\}. \tag{9}$$

[3] There is a relatively easier proof for unrestricted d by reduction from Set Cover, which we show in the full version.

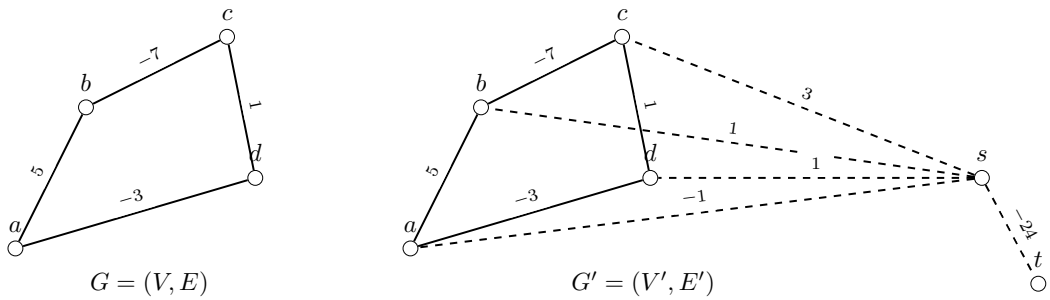


Figure 1 Reduction from Cut Strong Membership to Span Strong Membership. The number shown on the edges in E is the weight y_e, while on edges in E' is the product of $L = 24$ and weight y'_e.

Densest-Cut $\leq_p$ Cut WM $\leq_p$ Span WM

We now show that the Span Weak Membership is NP-Hard thereby showing Coverage Weak Membership is also NP-Hard for the restricted setting with $|T_i| = 2$ for all $i \in [n]$. We first define Cut Weak Membership.

Given a weighted graph $G = (V, E)$ with weights $\hat{y}_e$ on the edges and $\delta > 0$, the goal in Cut Weak Membership is to assert either $\hat{y} = (\hat{y}_e)_{e \in E}$ is in $S(K_c, \delta)$ or $\hat{y}$ is not in $S(K_c, -\delta)$ where

$$K_c = \left\{ \sum_{e \in \delta(S)} y_e \leq 0 \quad \forall S \in 2^V \setminus \emptyset, ||y||_\infty \leq 1 \right\}. \tag{10}$$

Note that in the Cut Weak membership problem, we have constraints $\sum_{e \in \delta(S)} y_e \leq 0$ instead of $\sum_{e \in E^+(S)} y_e \leq 0$ for all S.

▶ **Lemma 24.** *There is a reduction from Densest-Cut to Cut Weak Membership and from Cut Weak Membership to Span Weak Membership. Therefore, Coverage Weak Membership is NP-hard even when $d = 2$.*

We can now complete the proof of Theorem 5.

Proof of Theorem 5. Suppose there is an efficient α-approximation algorithm for Coverage Norm Extension. Then by Lemma 23 there is an efficient algorithm for Weak Validity problem for polytope K (8) and then by Theorem 22 we have an efficient algorithm for Coverage Weak Membership. But by Lemma 24, this is not possible unless $P = NP$. ◀

We here prove Lemma 25, which is a weaker statement than Lemma 24 to convey the main ideas. Recall that in Strong Membership problem, the goal is to decide if given vector y is in polyhedron K. Following our nomenclature, we define the following Strong Membership problems.

An instance of Span Strong Membership and Cut Strong Membership is given by a weighted graph $G = (V, E)$ with weights $\hat{y}_e$ on the edges, and the goal is to decide if vector $y = (y_e)_{e \in E}$ is in K_s and K_c respectively, with K_s and K_c as defined in (9), (10).

▶ **Lemma 25.** *There is a reduction from Densest-Cut to Cut Strong Membership, and from Cut Strong Membership to Span Strong Membership.*

Proof. For the second reduction, the instance of Cut Strong Membership is weighted graph $G = (V, E)$ with weights y_e on the edges. We assume $||y||_\infty \leq 1$ as otherwise clearly $y \notin K_c$.

Let $L = 2|E| + |V||E|$. We construct an instance of Span Strong Membership (see Figure 1), i.e., graph $G' = (V', E')$ and weights y'_e as follows:

$$V' = V \cup \{s, t\}\,,\ E' = E \cup \{s, t\} \cup \{v, s\} \quad \forall v \in V\,,\ y'_e = \begin{cases} \frac{y_e}{L} & \text{if } e \in E(G) \\ -\frac{1}{2L} w(\delta_G(v)) & \text{if } e = \{v, s\}, v \neq t \\ -1 & \text{if } e = \{s, t\}. \end{cases}$$

Then $||y'||_\infty \leq 1$.

Assume $y \notin K_c$, i.e., there exists $S \subseteq V$ s.t. $w(\delta_G(S)) > 0$. We need to show there exists $S' \subseteq V'$ s.t. $\sum_{e \in E^+(S')} y'_e > 0$. For $S' = S$, $L \sum_{e \in E^+(S')} y'_e = w(E_G(S)) + w(\delta_G(S)) + \sum_{v \in S} -\frac{1}{2} \cdot w(\delta_G(v)) = w(E_G(S)) + w(\delta_G(S)) - \frac{1}{2} \cdot (2w(E_G(S)) + w(\delta_G(S))) = \frac{w(\delta_G(S))}{2} > 0$.

Now assume $y \in K_c$, i.e., $\forall S \subseteq V, w(\delta_G(S)) \leq 0$. We need to show $\forall S' \subseteq V'$, $\sum_{e \in E^+(S')} y'_e \leq 0$. Since $y'_{\{s,t\}} = -1$ (and L is sufficiently large), we need to consider only those S' which do not contain either s or t. But we have shown that for such S', $\sum_{e \in E^+(S')} y'_e = \frac{w(\delta_G(S'))}{2L} \leq 0$.

Now we finish the proof by giving a reduction from Densest-Cut to Cut Strong Membership. Given an undirected graph $G = (V, E)$ and rational M, we want to know if there exists $S \subset V$ s.t. $\frac{\delta_G(S)}{|S||V \setminus S|} > M$. Consider the complete graph $G' = (V, E')$ where the weight of an edge is $\frac{1-M}{L}$ if it existed in E, and is $-\frac{M}{L}$ otherwise (note that edges may now have positive, negative, or zero weight). Let $L' = 2 \max\{M, |1 - M|\}$ be a sufficiently large quantity so that $||\hat{y}||_\infty < 1$. It is easy to see that $L\, w(\delta_{G'}(S)) = |\delta_G(S)| - M|S||V \setminus S|$. Therefore, $\exists S \subset V$ s.t. $w(\delta_{G'}(S)) > 0 \Leftrightarrow \exists S \subset V$ s.t. $\frac{|\delta_G(S)|}{|S||V \setminus S|} > M$. ◀

References

1. Ashwinkumar Badanidiyuru, Shahar Dobzinski, Hu Fu, Robert Kleinberg, Noam Nisan, and Tim Roughgarden. Sketching valuation functions. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1025–1035. Society for Industrial and Applied Mathematics, 2012.
2. Maria-Florina Balcan and Nicholas J. A. Harvey. Learning submodular functions. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 793–802, 2011.
3. Eric Balkanski, Aviad Rubinstein, and Yaron Singer. The limitations of optimization from samples. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1016–1027. ACM, 2017.
4. Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA, 1997.
5. Umang Bhaskar and Gunjan Kumar. Partial Function Extension with Applications to Learning and Property Testing. *arXiv preprint*, 2018. `arXiv:1812.05821`.
6. Liad Blumrosen and Noam Nisan. Combinatorial auctions. *Algorithmic game theory*, 267:300, 2007.
7. Paul S. Bonsma, Hajo Broersma, Viresh Patel, and Artem V. Pyatkin. The Complexity Status of Problems Related to Sparsest Cuts. In *Combinatorial Algorithms - 21st International Workshop, IWOCA 2010, London, UK, July 26-28, 2010, Revised Selected Papers*, pages 125–135, 2010.
8. Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. Maximizing social influence in nearly optimal time. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 946–957. SIAM, 2014.
9. Endre Boros, Toshihide Ibaraki, and Kazuhisa Makino. Error-Free and Best-Fit Extensions of Partially Defined Boolean Functions. *Inf. Comput.*, 140(2):254–283, 1998.

10 Deeparnab Chakrabarty and Zhiyi Huang. Recognizing Coverage Functions. *SIAM J. Discrete Math.*, 29(3):1585–1599, 2015.

11 Gerard Cornuejols, Marshall L Fisher, and George L Nemhauser. Exceptional paper—Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management science*, 23(8):789–810, 1977.

12 F Dragomirescu and C Ivan. The smallest convex extensions of a convex function. *Optimization*, 24(3-4):193–206, 1992.

13 Vitaly Feldman and Pravesh Kothari. Learning coverage functions and private release of marginals. In *Conference on Learning Theory*, pages 679–702, 2014.

14 Rafael M. Frongillo and Ian A. Kash. General Truthfulness Characterizations via Convex Analysis. In *Web and Internet Economics - 10th International Conference, WINE 2014, Beijing, China, December 14-17, 2014. Proceedings*, pages 354–370, 2014.

15 Chris Godsil and Gordon F Royle. *Algebraic graph theory*, volume 207. Springer Science & Business Media, 2013.

16 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

17 Subhash Khot. Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In *Proceedings 2001 IEEE International Conference on Cluster Computing*, pages 600–609. IEEE, 2001.

18 Andreas Krause, H Brendan McMahan, Carlos Guestrin, and Anupam Gupta. Robust submodular observation selection. *Journal of Machine Learning Research*, 9(Dec):2761–2801, 2008.

19 Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55(2):270–296, 2006.

20 Hans JM Peters and Peter P Wakker. Convex functions on non-convex domains. *Economics letters*, 22(2-3):251–255, 1986.

21 Leonard Pitt and Leslie G. Valiant. Computational limitations on learning from examples. *J. ACM*, 35(4):965–984, 1988.

22 Lior Seeman and Yaron Singer. Adaptive seeding in social networks. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 459–468. IEEE, 2013.

23 C. Seshadhri and Jan Vondrák. Is Submodularity Testable? *Algorithmica*, 69(1):1–25, 2014.

24 Donald M Topkis. Minimizing a submodular function on a lattice. *Operations research*, 26(2):305–321, 1978.

25 Armin Uhlmann. Roofs and convexity. *Entropy*, 12(7):1799–1832, 2010.

26 Min Yan. Extension of convex function. *arXiv preprint*, 2012. To appear in the Journal of Convex Analysis. `arXiv:1207.0944`.

A Appendix

Proof of Proposition 7

Consider the polyhedron Extension-P. If the partial function is extendible, then Extension-P is nonempty. Since the variables are non-negative, the polyhedron must have a vertex [4], and in particular there is a vertex in which at most n variables $w(S)$ are non-zero. This is because the dimension of the problem is 2^m, hence at a vertex at least 2^m constraints must be tight. But then at least $2^m - n$ of constraints $w(S) \geq 0$ must be tight.

Proof of Corollary 9

Consider the polytope $P = \{\sum_{I \in \mathcal{I}: v \in I} x_I = 1 \quad \forall v \in V(G), 0 \leq x_I \leq 1 \quad \forall I \in \mathcal{I}\}$. By the Theorem 8, there exists $x = \{x_I\}_{I \in \mathcal{I}}$ in P such that $\chi^*(G) = \sum_{I \in \mathcal{I}} x_I$. Consider $y = \{y_I\}_{I \in \mathcal{I}}$ given by $y_{\{v\}} = 1$ for all $v \in V(G)$ and 0 otherwise. Therefore, $y \in P$ and $|V(G)| = \sum_{I \in \mathcal{I}} y_I$. Consider $z = \lambda x + (1 - \lambda)y$ where $\lambda = \frac{|V(G)| - t}{|V(G)| - \chi^*(G)}$. Therefore, $z \in P$ and $\sum_{I \in \mathcal{I}} z_I = \lambda \sum_{I \in \mathcal{I}} x_I + (1 - \lambda) \sum_{I \in \mathcal{I}} y_I = t$.

Proof of Lemma 12

From Theorem 6, given a partial function H and $\alpha \geq 1$, there exists a coverage function f satisfying $f_i \leq f(T_i) \leq \alpha f_i$ for all $i \in [n]$ iff the following linear program is feasible, where the variables are the W-coefficients $w(S)$ for all $S \in 2^{[m]} \setminus \emptyset$:

$$f_i \leq \sum_{S:S\cap T_i \neq \emptyset} w(S) \leq \alpha f_i \quad \forall i \in [n]$$

$$w(S) \geq 0 \quad \forall S \in 2^{[m]} \setminus \emptyset.$$

By Farkas' Lemma, it follows that the above linear program is feasible iff the following linear program is infeasible, with variables y_i and z_i for all $i \in [n]$:

$$\alpha \sum_{i=1}^{n} f_i y_i < \sum_{i=1}^{n} f_i z_i \tag{11}$$

$$\sum_{i:S\cap T_i \neq \emptyset} y_i \geq \sum_{i:S\cap T_i \neq \emptyset} z_i \quad \forall S \in 2^{[m]} \setminus \emptyset \tag{12}$$

$$y_i, z_i \geq 0.$$

Now we proceed towards proving the claim. Suppose l_i's satisfy (4) and (3). Set y_i and z_i as follows: If $l_i \leq 0$ then let $y_i = -l_i$ and $z_i = 0$. Else if $l_i > 0$ then let $y_i = 0$ and $z_i = l_i$. It is easy to see that $y_i, z_i \geq 0$ and $l_i = z_i - y_i$ and hence (12) is satisfied by y_i's and z_i's. Further, $\alpha \sum_{i=1}^{n} f_i y_i = \alpha(\sum_{i:l_i \leq 0} f_i y_i + \sum_{i:l_i>0} f_i y_i) = -\alpha \sum_{i:l_i \leq 0} f_i l_i$ and similarly $\sum_{i=1}^{n} f_i z_i = \sum_{i:l_i>0} f_i l_i$. Thus (11) is also satisfied by y_i's and z_i's.

For the other direction observe that if the vector $y = (y_1, .., y_n), z = (z_1, ..., z_n) \geq 0$ satisfy (11) and (12) then wlog we can assume for any i, the minimum of y_i and z_i is 0 (otherwise we can decrease both y_i and z_i by the minimum of y_i and z_i, and $\alpha \geq 1$ allows (11) to remain true). Note that $\sum_i f_i y_i = \sum_{i:y_i \leq z_i} f_i y_i + \sum_{i:y_i>z_i} f_i y_i = \sum_{i:y_i>z_i} f_i y_i$, since $\min\{y_i, z_i\} = 0$ by the previous observation. Now suppose $y, z \geq 0$ satisfy (11) and (12). We thus have $\alpha \sum_{i=1}^{n} f_i y_i < \sum_{i=1}^{n} f_i z_i \Leftrightarrow \alpha \sum_{y_i>z_i} f_i y_i < \sum_{z_i>y_i} f_i z_i$. Now let $l_i = z_i - y_i$. This makes both (4) and (3) true.

Proof of Lemma 19

Suppose we are given a weighted bipartite graph $G = (A \cup [m], E)$ with weight f_v on each $v \in A$. Recall that κ is the minimum of $\frac{\sum_{w\in R} f_w}{f_v}$ over vertices $v \in A$ and $R \in \mathcal{F}_v$ where $\mathcal{F}_v = \{R \subseteq A \setminus \{v\} | N(R) \supseteq N(v)\}$ is the set of all $R \subseteq A \setminus \{v\}$ that covers all the neighbours of v.

We will use $f(R)$ $(R \subseteq A)$ to denote the summation $\sum_{v\in R} f_v$. If d is a constant then for each $v \in A$, we can find minimum of $f(R)$ over all $R \subseteq \mathcal{F}_v$ in $O(n^d)$ time where $n = |A|$. Therefore, by taking the minimum of the above minimum value over all vertices $v \in A$, we get the value of κ. For general d, we use an approximation algorithm for Set-Cover to find, for each vertex $v \in A$, a set $R'_v \in \mathcal{F}_v$ such that $f(R'_v) \leq f(R_v) \log d$ where R_v is the optimal set. It can be seen that $\kappa' = \min_{v\in A} \frac{f(R'_v)}{f_v}$ has the property $\kappa' \leq \kappa \log d$.

Proof of Lemma 23

The instance of weak validity problem is given by a vector $c \in \mathbb{Q}^n$ and rational numbers γ and $\epsilon > 0$. We show that there is a reduction from general Weak Validity to Weak Validity with instances satisfying $c_i \geq 0$ for all $i \in [n]$.

Let $N = \{i \in [n] | c_i \leq 0\}$. Consider a vector c' such that $c'_i = 0$ for $i \in N$ and c_i otherwise and $\gamma' = \gamma - \sum_{i \in N} |c_i|$. If x is in $S(K, \epsilon)$ then clearly $\bar{x}$ defined as $\bar{x}_i = -1$ if $i \in N$ and x_i otherwise, is also in $S(K, \epsilon)$. If for some x in $S(K, \epsilon)$, we have $(c')^T x \geq \gamma' - \epsilon$ then for $\bar{x} \in S(K, \epsilon)$, we have $c^T \bar{x} = \sum_{i \in N} |c_i| + (c')^T x \geq \gamma - \epsilon$. Also if for all $x \in S(K, -\epsilon)$, we have $(c')^T x \leq \gamma' + \epsilon$ then $c^T x \leq \sum_{i \in N} |c_i| + (c')^T x \leq \gamma + \epsilon$. This shows the reduction and hence we assume $c_i \geq 0$ in the instance of Weak Validity problem.

Let OPT and OPT' be the optimal value of Norm-P for $(f_1, \ldots, f_n) = (c_1, \ldots, c_n)$ and $(f_1, \ldots, f_n) = (Lc_1, \ldots, Lc_n)$ respectively (L will be chosen later). Obviously $OPT' = L \cdot OPT$. Let the approximation algorithm for Norm-P return β for instance $(f_1, \ldots, f_n) = (Lc_1, \ldots, Lc_n)$. Let $C = \sum_i c_i$. Therefore, $OPT' \leq \beta \leq OPT' + 2^{poly(n,m)}(LC)^\delta = L \cdot OPT + 2^{poly(n,m)}(LC)^\delta$ and hence $\beta/L \leq OPT + \frac{2^{poly(n,m)}(C)^\delta}{L^{1-\delta}}$. We set $L := \left(\frac{2^{poly(n,m)}(C)^\delta}{2\epsilon}\right)^{1/1-\delta}$ so that $\frac{2^{poly(n,m)}(C)^\delta}{L^{1-\delta}} = 2\epsilon$. Note that the number of bits to specify L is polynomial in $\langle c \rangle, \langle \epsilon \rangle, n, m$, where $\langle c \rangle$, $\langle \epsilon \rangle$ denote the number of bits required to represent these quantities. Thus, $OPT \leq \beta/L \leq OPT + 2\epsilon$. Now if $\gamma + \epsilon \leq \beta/L$ then for the optimal solution $x^* \in K$, $c^T x^* = OPT \geq \frac{\beta}{L} - 2\epsilon \geq \gamma - \epsilon$. If $\gamma + \epsilon \geq \beta/L$ then for all x in K (and hence $S(K, -\epsilon)$), we have $c^T x \leq OPT \leq \beta/L \leq \gamma + \epsilon$. Since at least one of these two conditions must hold, the conditions of weak validity problem can be correctly asserted.

Proof of Lemma 24

In the proof, for any vector y, recall that we use $||y||_\infty$ for $\max_i |y_i|$ and $||\hat{y} - y||$ for the Euclidean distance between $\hat{y}$ and y. We will frequently use the fact that the distance of a point x_0 from the hyperplane $w^T x + b = 0$ is equal to $\frac{|w^T x_0 + b|}{||w||}$.

Recall the definitions of Span Weak Membership, Cut Weak Membership and Densest Cut:

1. Given a weighted graph $G = (V, E)$ with weights $\hat{y}_e$ on the edges and $\delta > 0$,
   a. The goal in Span Weak Membership is to assert either $\hat{y} = (\hat{y}_e)_{e \in E}$ is in $S(K_s, \delta)$ or $\hat{y}$ is not in $S(K_s, -\delta)$ where

$$K_s = \left\{ \sum_{e \in E^+(S)} y_e \leq 0 \quad \forall S \in 2^V \setminus \emptyset, ||y||_\infty \leq 1 \right\},$$

   b. The goal in Cut Weak Membership is to assert either $\hat{y} = (\hat{y}_e)_{e \in E}$ is in $S(K_c, \delta)$ or $\hat{y}$ is not in $S(K_c, -\delta)$ where

$$K_c = \left\{ \sum_{e \in \delta(S)} y_e \leq 0 \quad \forall S \in 2^V \setminus \emptyset, ||y||_\infty \leq 1 \right\}.$$

   Note that in the Cut Weak membership, we have constraints $\sum_{e \in \delta(S)} y_e \leq 0$ instead of $\sum_{e \in E^+(S)} y_e \leq 0$ for all S.
2. In the Densest-Cut problem, given a graph $G = (V, E)$ and a positive rational M, the goal is to decide if there exist a set $S \subset V$ s.t. $\frac{|\delta(S)|}{|S||V \setminus S|} \geq M$.

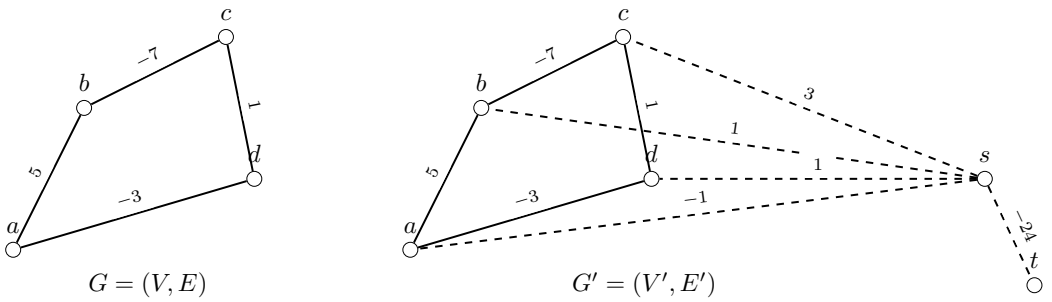


Figure 2 Reduction from Cut-Weak-Membership to Span-Weak-Membership. The number shown on the edges in E is the weight y_e, while on edges in E' is product of $L = 48$ and weight y'_e.

The Densest-Cut is known to be NP-Hard [7]. Note that $\frac{|\delta(S)|}{|S||V\setminus S|}$ called the density of cut $(S, V \setminus S)$ can take values only from $\left\{\frac{r}{s(|V|-s)} | 1 \le r \le |E|, 1 \le s \le |V|-1, r, s \in \mathbb{Z}_+\right\}$. Thus there are only polynomially many possible values of cut densities. We will use this fact in our proof.

▶ **Lemma 26.** *There is a reduction from Cut Weak Membership to Span Weak Membership.*

Proof. Our goal in Cut Weak Membership, given a graph a $G = (V, E)$ with weights $\hat{y}_e$ on edges and $\delta > 0$, is to assert either $\hat{y} = (\hat{y}_e)_{e\in E}$ is in $S(K_c, \delta)$ or $\hat{y}$ is not in $S(K_c, -\delta)$. If the point $\hat{y}$ violates the constraint $||y||_\infty \le 1$ of K_c then it can be asserted that $\hat{y}$ is not in $S(K_c, -\delta)$. So we assume $||\hat{y}||_\infty \le 1$. Given this assumption, we have $w(\delta_G(v)) \le |E|$.

We construct an instance of Span-Weak-Membership (see Figure 2), i.e., graph $G' = (V', E')$, $\hat{y}'_e$ and δ' as follows (the values of B and L will be set later):

$V' = V \cup \{s, t\}$

$E' = E \cup \{\{s, t\}\} \cup \{\{v, s\}\} \quad \forall v \in V', v \ne \{s, t\}$

$$\hat{y}'_e = \begin{cases} \frac{y_e}{L} & \text{if } e \in E \\ -\frac{\frac{1}{2}w(\delta_G(v))}{L} & \text{if } e = \{v, s\}, v \ne t \\ -\frac{B}{L} & \text{if } e = \{s, t\}. \end{cases}$$

The value of B is set to $2|E| + |V||E|$ so that $\sum_{e\in E^+_{G'}(S)} \hat{y}'_e \le \frac{-B+|E|+1/2|V||E|}{L} \le 0$ for all S containing either s or t. Further, $L = 2B$ so that $||\hat{y}'||_\infty = 1/2$ where $\hat{y}' = (\hat{y}'_e)_{e\in E'}$. Finally we choose $\delta' = \frac{1}{2}\min\left\{\frac{\sqrt{|E|}\delta}{2L\sqrt{|E'|}}, \frac{|E|+1/2|V||E|}{\sqrt{|E'|}L}, \frac{1}{2}\right\}$.

▷ Claim 27. For all $S \subseteq V$, $w(E^+_{G'}(S)) = \frac{w(\delta_G(S))}{2L}$.

Proof. This is because

$$L\, w(E^+_{G'}(S)) = L \sum_{e\in E^+_{G'}(S)} w'_e = w(E_G(S)) + w(\delta_G(S)) + \sum_{v\in S} -\frac{1}{2}\, w(\delta_G(v))\,,$$

and since $w(\delta_G(v))$ counts edges in $E_G(S)$ twice and edges in $\delta_G(S)$ once,

$$L\, w(E^+_{G'}(S)) = w(E_G(S)) + w(\delta_G(S)) - \frac{1}{2}\cdot(2w(E_G(S)) + w(\delta_G(S))) = \frac{w(\delta_G(S))}{2}\,. \quad \triangleleft$$

Suppose the algorithm for Span Weak Membership asserts that the point $\hat{y}'$ is in $S(K_s, \delta')$. If $\hat{y}'$ satisfies all the constraints $\sum_{e\in E^+_{G'}(S)} y_e \le 0$ for all $S \in 2^V \setminus \emptyset$ then the point $\hat{y}$ must satisfy all the constraints $\sum_{e\in\delta_G(S)} y_e \le 0$ for all $S \in 2^V \setminus \emptyset$ (because by the Claim 27

$w(\delta_G(S)) = 2L \cdot w(E^+_{G'}(S)))$ and hence $\hat{y} \in K_c$. Thus $\hat{y} \in S(K_c, \delta)$. Now suppose $\hat{y}'$ violates a constraint $\sum_{e \in E^+_{G'}(R)} y_e \leq 0$ for some $R \in 2^V \setminus \emptyset$. Since $\hat{y}' \in S(K_s, \delta')$, it is at most δ' distance away from the hyperplanes corresponding to the violated constraints. Therefore, we have $w(E^+_{G'}(R)) = \sum_{e \in E^+_{G'}(R)} \hat{y}'_e \leq \delta' \sqrt{|E'|}$. By Claim 27, $w(\delta_G(R)) \leq 2L\delta' \sqrt{|E'|}$. Therefore, the point $\hat{y}$ is at most $\frac{2L\delta'\sqrt{|E'|}}{\sqrt{|E|}}$ distance from K_c. Since $\delta' < \frac{\sqrt{|E|}\delta}{2L\sqrt{|E'|}}$, so $\hat{y} \in S(K_c, \delta)$.

Suppose the algorithm for Span Weak Membership problem asserts that the point $\hat{y}'$ is not in $S(K_s, -\delta')$. If $\hat{y}'$ violates a constraint $\sum_{e \in E^+_{G'}(S)} y_e \leq 0$ for some $S \in 2^V \setminus \emptyset$ then the point $\hat{y}$ also violates $\sum_{e \in \delta_G(S)} y_e \leq 0$ for S (by Claim 27). Hence, it can be asserted that $\hat{y}$ is not in $S(K_c, -\delta)$. So now assume that $\hat{y}'$ satisfies all the constraints $\sum_{e \in E^+_{G'}(S)} y_e \leq 0$ for all $S \in 2^V \setminus \emptyset$. Also, as shown earlier, $\hat{y}'$ satisfies the other constraints of K_s. Since $\hat{y}'$ is in K_s but not in $S(K_s, -\delta')$, some $y \in S(\hat{y}', \delta')$ must have distance $< \delta'$ from some hyperplane of K_s. The distance of $\hat{y}'$ from the hyperplane $\sum_{e \in E^+_{G'}(S)} y_e = 0$ for S containing s or t is at least $\frac{|-B+|E|+1/2|V||E'||}{\sqrt{|E'|L}} = \frac{|E|+1/2|V||E'|}{\sqrt{|E'|L}} > \delta'$. Also for any $y \in S(\hat{y}', \delta')$, we have $||y||_\infty - ||\hat{y}'||_\infty \leq ||y - \hat{y}||_\infty \leq ||y - \hat{y}'||$. So $||y||_\infty \leq \delta + 1/2 \leq 1$ for all $y \in S(\hat{y}', \delta')$. Therefore, it must be the case that distance of $\hat{y}'$ from the hyperplane $\sum_{e \in E^+_{G'}(S)} y_e = 0$ for some $S \in 2^V \setminus \emptyset$ is $< \delta'$. By Claim 27, the distance of the point $\hat{y}$ from the hyperplane $\sum_{e \in \delta_G(S)} y_e = 0$ is at most $\frac{2L\sqrt{|E'|}\delta'}{\sqrt{|E|}} < \delta$. Hence, $\hat{y}$ is not in $S(K_c, -\delta)$. ◀

Now we finish the proof by giving reduction from Densest-Cut to Cut Weak Membership.

▶ **Lemma 28.** *There is a reduction from Densest-Cut to Cut Weak Membership.*

Proof. In the Densest Cut problem, a graph $G = (V, E)$ and a positive rational M are given and the goal is to determine if there exists a set $S \subset V$ s.t. the density of the cut $(S, V \setminus S)$ is at least M, i.e., $\frac{|\delta_G(S)|}{|S||V \setminus S|} \geq M$. Let $M = \frac{p}{q}$ for positive integers p, q. We set L to $2\max\{M, |1 - M|\}$ (so that later, $||\hat{y}||_\infty = 1/2$) and t to $\frac{1}{qL}$.

Given the graph $G = (V, E)$ and M, the instance of Cut Weak Membership is a complete graph $G' = (V, E')$ (so $|E'| = \frac{|V|(|V|-1)}{2}$), weight $\hat{y}_e$ on each edge $e \in E'$ such that $\hat{y}_e$ is $\frac{1-M}{L}$ if it existed in E and $\frac{-M}{L}$ otherwise, and $\delta = \frac{1}{2}\min\{\frac{1}{2}, \frac{t}{\sqrt{|E'|}}\}$. Let $\hat{y} = (\hat{y}_e)_{e \in E'}$. This defines the polytope K_c as in (10) for the instance of Cut Weak Membership.

It is easy to see that $w(\delta_{G'}(S)) = \frac{1}{L}(|\delta_G(S)| - M|S||V \setminus S|)$. Therefore, $\exists S \subset V$ s.t. $w(\delta_{G'}(S)) \geq 0 \Leftrightarrow \exists S \subset V$ s.t. $\frac{|\delta_G(S)|}{|S||V \setminus S|} \geq M$.

Since M is equal to $\frac{p}{q}$ for some $p, q \in \mathbb{Z}_+$, therefore the weight of an edge is either $\frac{q-p}{qL}$ or $\frac{-p}{qL}$. So if a cut value $w(\delta_{G'}(S))$ is strictly positive for any S then $w(\delta_{G'}(S))$ must be at least $\frac{1}{qL} = t$. Similarly, if $w(\delta_{G'}(S)) < 0$ then we have $w(\delta_{G'}(S)) \leq -t$.

Now suppose an algorithm for Cut Weak Membership asserts $\hat{y}$ is in $S(K_c, \delta)$. Thus for all S, $w(\delta_{G'}(S)) = \sum_{e \in \delta_{G'}(S)} \hat{y}_e \leq \sqrt{|E'|}\delta$. Since $\delta < \frac{t}{\sqrt{|E'|}}$, so it must be the case that for all S, $w(\delta_{G'}(S)) \leq 0$. This implies that for all S, the cut density $\frac{|\delta_G(S)|}{|S||V \setminus S|} \leq M$.

Suppose the algorithm for Cut Weak Membership asserts that $\hat{y}$ is not in $S(K_c, -\delta)$. If $\hat{y} \notin K_c$ (and since $||\hat{y}||_\infty \leq 1$) then clearly there exists a set S such that $w(\delta_{G'}(S)) > 0$. This implies there is a cut $(S, V \setminus S)$ with density $\frac{|\delta_G(S)|}{|S||V \setminus S|} > M$. Now assume $\hat{y}$ is in K_c. Now for any $y \in S(\hat{y}, \delta)$, we have $||y||_\infty - ||\hat{y}||_\infty \leq ||y - \hat{y}||_\infty \leq ||y - \hat{y}|| \leq \delta$. So $||y||_\infty \leq \delta + 1/2 < 1$. So there must exist a hyperplane $\sum_{e \in \delta_{G'}(S)} y_e = 0$ for some S with at most δ distance from $\hat{y}$. Therefore, there exist a set S with $0 \geq w(\delta_{G'}(S)) \geq -\sqrt{|E'|}\delta$. Since $\delta < \frac{t}{\sqrt{|E'|}}$, this

means $w(\delta_{G'}(S)) = 0$ and hence density of cut $(S, V \setminus S)$ is M. Thus, if an algorithm for Cut Weak Membership asserts that $\hat{y}$ is not in $S(K_c, -\delta)$ then there exists a cut with density at least M.

Therefore, assuming an efficient algorithm for Cut Weak Membership, it can be determined if there exists a cut with density at least M or all cuts have density at most M. However, the goal in Densest Cut is to determine if there is a cut with density $\geq M$ or all cuts have density strictly less than M. But since the density can take only polynomial number of values $\left\{ \frac{r}{s(|V|-s)} | 1 \leq r \leq |E|, 1 \leq s \leq |V| - 1, r, s \in \mathbb{Z}_+ \right\}$ (as noted before), by using at most two oracle calls to the Cut-Weak-Membership problem we can solve the original problem. ◀

Almost Optimal Classical Approximation Algorithms for a Quantum Generalization of Max-Cut

Sevag Gharibian
University of Paderborn, Germany
Virginia Commonwealth University, Richmond, VA, USA
sevag.gharibian@upb.de

Ojas Parekh
Sandia National Laboratories, Albuquerque, New Mexico, USA
odparek@sandia.gov

Abstract

Approximation algorithms for constraint satisfaction problems (CSPs) are a central direction of study in theoretical computer science. In this work, we study classical product state approximation algorithms for a physically motivated quantum generalization of Max-Cut, known as the quantum Heisenberg model. This model is notoriously difficult to solve exactly, even on bipartite graphs, in stark contrast to the classical setting of Max-Cut. Here we show, for any interaction graph, how to classically and efficiently obtain approximation ratios 0.649 (anti-feromagnetic XY model) and 0.498 (anti-ferromagnetic Heisenberg XYZ model). These are almost optimal; we show that the best possible ratios achievable by a product state for these models is 2/3 and 1/2, respectively.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Semidefinite programming; Theory of computation → Quantum complexity theory

Keywords and phrases Approximation algorithm, Max-Cut, local Hamiltonian, QMA-hard, Heisenberg model, product state

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.31

Category APPROX

Funding *Sevag Gharibian*: NSF grants CCF-1526189 and CCF-1617710
Ojas Parekh: Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. Also supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Quantum Algorithms Teams program.

Acknowledgements We thank David Gosset and Mark Wilde for helpful discussions, and an anonymous referee for catching a technical error in an earlier version of this draft.

1 Introduction

The study of approximation algorithms for NP-complete problems is a central area of research in theoretical computer science (see, e.g., [20, 30]). Indeed, the field has seen breakthroughs such as the celebrated Goemans-Williamson [19] 0.878-approximation algorithm for Max-Cut, and the PCP theorem [4, 3], which yielded a general framework for showing hardness of approximation results. Here, an approximation algorithm A with ratio $0 < r < 1$ is defined as follows: Given an instance Π of a maximization problem with optimal value OPT, A runs in polynomial time and outputs a value $\widetilde{\text{OPT}}$ satisfying $r\text{OPT} \leq \widetilde{\text{OPT}} \leq \text{OPT}$. Focal points


© Sevag Gharibian and Ojas Parekh;
licensed under Creative Commons License CC-BY

of study in approximation algorithms are Boolean constraint satisfaction problems (CSPs) such as Max-SAT and Max-Cut, in which one is roughly given a set of local constraints acting on $k \in O(1)$ bits each (out of a total of n bits), and asked to compute the largest number of constraints which are simultaneously satisfiable.

In the quantum setting, CSPs are naturally generalized by the *k-local Hamiltonian problem (k-LH)* [24]. In the latter, one is given as input an exponentially large (in the number of qubits, n) Hermitian matrix H known as a *local Hamiltonian*, which has a succinct description in terms of local "quantum clauses." The goal is to estimate the smallest eigenvalue of H, $\lambda_{\min}(H)$, i.e. the *ground state energy* of H. Slightly more formally, a k-local Hamiltonian $H = \sum_{S \subseteq [n]} H_S$ acts on n qubits in total, with each local quantum "clause" H_S acting on a constant number k of qubits denoted by subset $S \subseteq [n]$ with $|S| = k$. (Thus, each H_S is a $2^k \times 2^k$ Hermitian matrix. Note that formally, H_S implicitly denotes operator $I_{[n]\setminus S} \otimes H_S$; this ensures dimensions match in the sum over clauses.) Quantum CSPs in which the matrices H_S are diagonal correspond to classical CSPs.

The problem k-LH is not only physically motivated (it is the problem of estimating the energy of a quantum many-body system when cooled to near absolute zero), but also complexity theoretically – it was the first known QMA-complete problem [24], where Quantum Merlin Arthur (QMA) is the quantum analogue of NP. As such, k-LH has been a central problem of study in the field of Quantum Hamiltonian Complexity (see, e.g. [28, 18] for surveys), which (among other aims) uses tools from complexity theory to uncover the limits and structure of physical systems in nature. In recent years, this interdisciplinary research has led to a growing body of work on classical *approximation algorithms* for k-LH. It is this direction which we pursue in this paper.

1.1 Product state algorithms and previous work

We begin by reviewing previous work on approximation algorithms for k-LH.

Mean-field or product-state algorithms

All known classical approximation algorithms for k-LH fall under the category of *mean-field* or *product-state* algorithms. Here, the issue is that the optimal solution to a k-LH instance may be an exponentially large quantum state $|\psi\rangle \in \mathbb{C}^{2^n}$ (which would be the *ground state* or eigenvector of H corresponding to its ground state energy, $\lambda_{\min}(H)$). Any classical algorithm for approximating k-LH must hence presumably pick a reasonable succinctly representable class of quantum states to optimize over; the simplest such class is the set of n-qubit *product states*. A product state is the quantum analogue of a product distribution – the entire 2^n-dimensional vector $|\psi\rangle$ is fully specified by locally giving an assignment $|\psi_i\rangle \in \mathbb{C}^2$ to each qubit i, i.e. $|\psi\rangle = |\psi_1\rangle \otimes \cdots \otimes |\psi_n\rangle$.

▶ Remark. It is crucial to note that even though product states are not entangled, they nevertheless generalize classical bit string assignments, and are thus *NP-hard to optimize over* in the worst case. Thus, even with this simplest ansatz of product states, approximating k-LH is highly non-trivial.

Previous work for QMA-complete models

We now outline the known approximation algorithms for k-LH, which are all mean-field algorithms. The first such work was due to Bansal, Bravyi, and Terhal [5], who gave a classical polynomial-time approximation scheme (PTAS) for k-LH on bounded degree planar

graphs. Next, Gharibian and Kempe [17] gave a PTAS for computing product-state solutions to dense CSPs, and showed their algorithm yielded a d^{1-k} approximation for dense k-LH on local d-dimensional systems. Brandão and Harrow [7] then gave PTAS-es for k-LH in three settings: Planar, dense, and low threshold rank graphs. Most recently, Bravyi, Gosset, König, and Temme [9] gave a $O(\log n)$-approximation algorithm for traceless 2-local Hamiltonians. As we shall see, this last work may be viewed as complementary to ours (and indeed, the techniques used are similar, although independently developed) – the algorithm of [9] is more general than ours (applies to all traceless Hamiltonians) but has a non-constant approximation ratio ($O(\log n)$ ratio). We take the complementary route: We study a more specific model, the central quantum Heisenberg model, but in return are able to achieve substantially stronger *constant* approximation ratios. Finally, Lee and Hallgren [25] obtain a non-trivial constant-factor approximation algorithm for 2-LH when each clause is positive semi-definite. We remark that with the exception of [5], all of these works are based on semidefinite programs (SDP).

Previous work for Hamiltonians of "intermediate" complexity

For completeness, we also note that Bravyi [8] and Bravyi and Gosset [11] showed fully polynomial randomized approximation schemes (FPRAS) for approximating the partition function[1] of certain ferromagnetic models, such as the ferromagnetic transverse field Ising model (ferromagnetic TIM). In general, the TIM problem is StoqMA-complete, as shown by Bravyi and Hastings [10]. Here, MA $\subseteq$ StoqMA $\subseteq$ QMA, and it is generally believed StoqMA is strictly smaller than QMA (the former is in the Polynomial-time Hierarchy, whereas the latter is believed not to be). Thus, such models may be thought of as being of "intermediate" complexity.

Brief note on the quantum PCP theorem

An advantage of any mean-field classical approximation algorithm for k-LH is that it yields negative progress on the central open question: Does a quantum PCP theorem[2] hold [1, 2]? This is because such algorithms show that a *classical* (i.e. NP) witness suffices to attain certain approximation ratios for k-LH. Thus, unless NP = QMA (which is believed highly unlikely), a quantum PCP theorem for k-LH with the same approximation ratios cannot hold.

1.2 Our results

We give classical approximation algorithms for a maximization version of the fundamental *quantum Heisenberg model*, which can be thought of as a family of Hamiltonians generalizing the NP-complete Max-Cut problem.

Maximization versus minimization

For clarity, we study the natural *maximization* variant of k-LH, in which one is given H and asked to estimate its *largest* eigenvalue $\lambda_{\max}(H)$. We study this variant for two reasons (see also [16]): First, in the minimization setting, if $\lambda_{\min}(H) = 0$, the notion of an approximation

[1] The ability to compute the partition function allows one in turn to solve k-LH.

[2] Recently, the "entangled non-local games" version of the PCP theorem has been established under randomized reductions [26]. The "hardness of approximation" version involving approximating ground state energies of local Hamiltonians, however, which is relevant to this work, remains open.

ratio is not well-defined, and second, the maximization setting allows us to naturally align with classical approximation algorithms for CSPs such as Max-Cut. We remark that in the exact setting, computing $\lambda_{\min}(H)$ is equivalent in complexity to computing $\lambda_{\max}(H)$ since $\lambda_{\min}(H) = \lambda_{\max}(-H)$ – thus, both maximization and minimization variants of k-LH are QMA-complete. More precisely, if H is a Hamiltonian corresponding to an instance of the (anti-ferromagnetic) quantum Heisenberg model, then we approximate the instance $\lambda_{\max}(mI - H)$, where m is the number of clauses. In terms of approximability, the complexity of both models need not coincide. An appropriate classical analogy is the relationship of the Ising problem on graphs, $\min_{z_i \in \{-1,1\}} \sum_{ij \in E} z_i z_j$, for which an $O(\log n)$-approximation is the best known (see, e.g., [13]) and the Max-Cut problem, $\max_{z_i \in \{-1,1\}} \sum_{ij \in E} (1 - z_i z_j)/2$, for which the Goemans-Williamson 0.878-approximation is known. These problems are equivalent from an exact optimization perspective. From an approximation perspective, the standard quantum Heisenberg model is a generalization of the Ising problem, while the problem we study is a generalization of Max-Cut (see Appendix A for details). We note that Bravyi et al.'s $O(\log n)$-approximation for traceless 2-local Hamiltonians [9] includes the standard quantum Heisenberg model as a special case.

The quantum Heisenberg model

The Heisenberg model is fundamental to the study of magnetism, and has received attention for at least almost a century now (e.g. the well-known Bethe ansatz of 1931 [6]). It is a family of 2-local Hamiltonians, defined in this paper as having constraints H_{ij} acting on qubits i and j of the form (see Section 2 for formal definitions):

$$H_{ij} = I - \alpha X_i \otimes X_j - \beta Y_i \otimes Y_j - \gamma Z_i \otimes Z_j,$$

for Pauli matrices X, Y, Z, and where X_i indicates X acts on qubits i. (Recall we study *maximization*, i.e. estimating $\lambda_{\max}(H)$.) Three important well-known special cases of this model are: (1) the Max-Cut problem ($\alpha = \beta = 0$, $\gamma = 1$) (in Appendix A, we sketch why this case indeed captures Max-Cut), (2) the (anti-ferromagnetic) XY model ($\alpha = \beta = 1$, $\gamma = 0$), and (3) the (anti-ferromagnetic) Heisenberg model ($\alpha = \beta = \gamma = 1$), which we also refer to as the anti-ferromagnet. The latter, for example, is notoriously difficult to solve *even on bipartite graphs*, in contrast to Max-Cut. The only solutions for the anti-ferromagnet we are aware of is on the 1D chain [6] and on the complete graph (see, e.g., [14]). This notoriety is well-deserved – when non-negative polynomial-size weights are allowed on each constraint, both the XY model and anti-ferromagnet are QMA-hard [14, 29].

In this paper, we first show (Section 4) how to approximate the XY model and anti-ferromagnet almost optimally. The following is an informal statement (see Theorem 6 for a formal statement).

▶ **Theorem 1.** *Let $\alpha, \beta, \gamma \in \{0, 1\}$. Then, there exists a randomized, polynomial time classical algorithm for the quantum Heisenberg model which outputs a product state solution with ratio at least:*

- 0.878 *if $\alpha + \beta + \gamma = 1$ (equivalent to Max-Cut),*
- 0.649 *if $\alpha + \beta + \gamma = 2$ (equivalent to the XY model),*
- 0.498 *if $\alpha + \beta + \gamma = 3$ (anti-ferromagnet).*

We then show in Corollary 5 that these ratios are almost optimal, in the sense that the *best* approximation ratios possible for a product state solution (whether efficiently attainable or not) to the XY model and anti-ferromagnet are at most 2/3 and 1/2, respectively. It

should be noted that, in contrast, the naive "random assignment" strategy (i.e. choose the maximally mixed state $I/2^n$ as the assignment) yields ratios of only 1/3 and 1/4 for the XY model and anti-ferromagnet, respectively.

Next, in Section 4.1 we give two ways in which our algorithm (or a variant of it) can be applied to a broader class of Hamiltonians:

- Section 4.1.1 shows how to relax the constraint that $\alpha, \beta, \gamma \in \{0, 1\}$. Specifically, we allow a different set of parameters $\alpha_{ij}, \beta_{ij}, \gamma_{ij} \in [-1, 1]$ for each edge $(i, j) \in E$. In return for this generality, the approximation ratios we obtain are slightly weaker.
- Section 4.1.2 uses a trick from entanglement theory [22, 23] to characterize the class of models which can be reduced to the Heisenberg model via application of local unitaries, and hence to which our algorithms apply.

1.3 Techniques

Our algorithms are based on semidefinite programming (SDP), and in particular use the first level of a non-commutative generalization of the Lasserre SDP hierarchy. Similar generalizations have been used previously in [7, 9]. Note that a key difference between our approach and the previous SDP-based works of [16, 7] is that the SDPs we derive are relaxations not just of the best attainable *product state* objective function value, but rather of the true optimal value $\lambda_{\max}(H)$ itself. This is why the ratios we obtain in Theorem 6 can be close to optimal for a product state ansatz. We note that a simple modification of our SDP relaxation does give an upper bound on the NP-hard problem of finding the best product-state solution; our techniques can be used to yield classical approximation algorithms for this problem as well.

1.4 Open questions

Many questions in the study of approximability in the quantum setting remain open. For example, what are the best achievable approximation ratios classically for the Heisenberg model, and do hardness of approximation results based on the unique games conjecture yield tight bounds as they do for Max-Cut and related classical CSPs? Can tight ratios of 2/3 and 1/2 be obtained for the XY model and anti-ferromagnet, respectively? Are there constant-factor approximation algorithms for general k-LH (recall [9] give $O(\log n)$ approximations for traceless 2-local Hamiltonians)? How well can one approximate "intermediate" Hamiltonian models such as the *anti-ferromagnetic* TIM (recall [8, 11] approximate the ferromagnetic TIM)? Can one optimize approximately over more general ansatzes than mean-field/product states, such as tensor network states? Can quantum approximation algorithms *provably* outperform the best classical approximation algorithms? Finally, does a quantum PCP theorem (in the sense of "hardness of approximation for quantum CSPs") hold? It is hoped that the current paper will act as a step towards resolutions for some of these problems.

1.5 Organization

In Section 2, we give definitions and preliminaries. Section 3 gives upper bounds on the power of the mean-field ansatz. Section 4 gives our approximation algorithms. Certain technical proofs are deferred to Appendix B. Some background in basic quantum information is assumed; see, for example, Nielsen and Chuang [27] for a standard reference.

2 Preliminaries

2.1 Notation

Let $[n] := \{1, \ldots, n\}$. The sets $\mathcal{H}(\mathcal{X})$ and $\mathcal{D}(\mathcal{X})$ denote the sets of Hermitian and density operators acting on complex Euclidean space $\mathcal{X}$. For $A, B \in \mathcal{H}(\mathcal{X})$, we say $A \succeq B$ if $A - B$ is positive semidefinite, i.e. $A - B \succeq 0$. The spectral/operator norm of A is denoted $\|A\|_\infty = \mathrm{tr}(\sqrt{A^\dagger A})$.

2.2 Physically motivated 2-local Hamiltonians

Let $G = (V, E)$ be a simple, undirected graph with $|V| = n$ and $|E| = m$. In this section, we study physically motivated 2-local Hamiltonians H based on the quantum Heisenberg model, $H = \sum_{(i,j)\in E} w_{ij} H_{ij}$ for $H_{ij} = \alpha X_i X_j + \beta Y_i Y_j + \gamma Z_i Z_j$ (more accurately, since we are in the setting of maximization, we use local terms as given in Equation (1)), where we consider $\alpha, \beta, \gamma \in \{0, 1\}$ and $w_{ij} \geq 0$. This includes QMA-hard special cases such as the quantum Heisenberg anti-ferromagnet [14, 29]. Here, X, Y, Z are the Pauli matrices

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

and X_i, Y_i, Z_i refer to the Pauli matrices acting on the ith qubit (i.e., tensored with identity on all other qubits).

Specifically, we consider the equivalent (in the setting of exact computation) maximization variant where each local term is defined

$$H_{ij} = I - \alpha X_i X_j - \beta Y_i Y_j - \gamma Z_i Z_j, \tag{1}$$

and our goal is to estimate the *largest* eigenvalue of $H = \sum_{(i,j)\in E} w_{ij} H_{ij}$ with $w_{ij} \geq 0$. This variant is clearly still QMA-hard, and includes as a special case, for example, the canonical NP-complete problem Max-Cut, obtained up to scaling by a constant factor of 2) by setting $\alpha = 0, \beta = 0, \gamma = 1$.

We now set definitions for the rest of this paper. Let $F_{\alpha,\beta,\gamma}$ denote the set of all H with (non-negative weighted) constraints of the form of Equation (1), with parameters α, β, γ and on all interaction graphs G (for all $n \geq 0$). For example, $F_{0,0,1}$ denotes the set of all possible Max-Cut instances with non-negative edge weights. In this paper, we refer to the family $F = \bigcup_{\alpha,\beta,\gamma\in\{0,1\}} F_{\alpha,\beta,\gamma}$ as "the Heisenberg model". Let $\mathrm{SEP} = \mathrm{conv}(\bigotimes_{i=1}^n \rho_i \mid \rho_i \in \mathcal{D}(\mathbb{C}^2))$ for $\mathrm{conv}(S)$ the convex hull of set S, i.e. SEP is the set of fully separable quantum states on n qubits.

3 Upper bounds on product state ratios

As quantum states on n qubits generally require exponential space to represent, a classical approximation algorithm for estimating ground state energies must generally optimize over a restricted class of quantum states, or an *ansatz*. Our ansatz in this section will be to optimize over SEP. To formalize this, we first define the notion of a product state ratio.

Product state ratio

Let $H \in \mathcal{H}((\mathbb{C}^2)^{\otimes n})$ be a Hermitian operator with largest eigenvalue $\mathrm{OPT}(H) = \lambda_{\max}(H)$, and let

$$\mathrm{OPT}_{\mathrm{prod}}(H) := \max_{\rho\in\mathrm{SEP}} \mathrm{tr}(H\rho).$$

By convexity, the optimal ρ here is a (pure) product state. The *product state ratio* is defined as $\mathrm{OPT}_{\mathrm{prod}}(H)/\mathrm{OPT}(H)$. For the Heisenberg model in particular, for any fixed $\alpha, \beta, \gamma \in \{0, 1\}$, define

$$\Gamma_{\alpha,\beta,\gamma} = \min_{H \in F_{\alpha,\beta,\gamma}} \frac{\mathrm{OPT}_{\mathrm{prod}}(H)}{\mathrm{OPT}(H)},$$

the worst-case product state ratio over all Hamiltonians in $F_{\alpha,\beta,\gamma}$.

By definition, $\Gamma_{\alpha,\beta,\gamma}$ yields an upper bound on the best approximation ratio achievable by any approximation algorithm using a product state ansatz. It is thus crucial to understand $\Gamma_{\alpha,\beta,\gamma}$, which we now do for the Heisenberg model. For this, we first give two lemmas which fully characterize the optimal product state ratio on a *single* (unit weight) edge. Note the characterization we give is more general than how we defined the Heisenberg model here, in that it applies for any $\alpha, \beta, \gamma \in \mathbb{R}$. (For clarity, the term proportional to the identity is omitted in Lemmas 2 and 3 below, but is accounted for in the subsequent statement of Corollary 4.) The proofs of both lemmas are deferred to Appendix B.

▶ **Lemma 2.** *Let $H = \alpha X \otimes X + \beta Y \otimes Y + \gamma Z \otimes Z$ for $\alpha, \beta, \gamma \in \mathbb{R}$. Then $\mathrm{OPT}_{\mathrm{prod}}(H) = \|(\alpha, \beta, \gamma)\|_\infty$.*

▶ **Lemma 3.** *Let $H = \alpha X \otimes X + \beta Y \otimes Y + \gamma Z \otimes Z$ for $\alpha, \beta, \gamma \in \mathbb{R}$. Then*

$$\mathrm{OPT}(H) = \max(|\alpha - \beta| + \gamma, |\alpha + \beta| - \gamma).$$

The following corollary now follows essentially immediately by applying Lemmas 2 and 3 to a single unit weight edge of the form in Equation 1 (i.e. with an identity term).

▶ **Corollary 4.** *For any $\alpha, \beta, \gamma \in \mathbb{R}$,*

$$\Gamma_{\alpha,\beta,\gamma} \leq \frac{1 + \max(|\alpha|, |\beta|, |\gamma|)}{1 + \max(|\alpha - \beta| - \gamma, |\alpha + \beta| + \gamma)}.$$

Proof. Combine Lemmas 2 and 3 with the following additional observation: The values α, β, γ, as defined for $F_{\alpha,\beta,\gamma}$, should be interpreted as $-\alpha, -\beta, -\gamma$ for Lemmas 2 and 3 due to how Equation 1 is stated. As a result, the positions of the γ and $-\gamma$ terms are swapped in the result of Lemma 3. ◀

We thus have the following for the special case of the Heisenberg model we consider here (i.e. $\alpha, \beta, \gamma \in \{0, 1\}$).

▶ **Corollary 5.** *For any $\alpha, \beta, \gamma \in \{0, 1\}$, if:*

- *$\alpha + \beta + \gamma = 1$, then $\Gamma_{\alpha,\beta,\gamma} = 1$.*
- *$\alpha + \beta + \gamma = 2$, then $\Gamma_{\alpha,\beta,\gamma} \leq 2/3$.*
- *$\alpha + \beta + \gamma = 3$, then $\Gamma_{\alpha,\beta,\gamma} \leq 1/2$.*

Proof. When $\alpha, \beta, \gamma \in \{0, 1\}$, the bound of Corollary 4 simplifies to

$$\Gamma_{\alpha,\beta,\gamma} \leq \frac{2}{1 + \alpha + \beta + \gamma},$$

from which the upper bounds claimed follow. The matching lower bound for $\alpha + \beta + \gamma = 1$ is obtained since H can be mapped via local Pauli gates to $H' \in F_{0,0,1}$, i.e. H' is diagonal in the standard basis. Thus, product states are optimal in this case. For example, applying local Hadamard gates to each qubit maps any $H \in F_{1,0,0}$ to $H' \in F_{0,0,1}$. (A matching lower bound can also be obtained for $\alpha + \beta + \gamma = 3$ by observing that $H_{ij} \succeq 0$, and using the general result that any local Hamiltonian H' (not necessarily from the Heisenberg model) with positive semidefinite constraints satisfies $\mathrm{OPT}_{\mathrm{prod}}(H')/\mathrm{OPT}(H') \geq 1/2$ [16]. However, unlike Theorem 6, the lower bound of [16] is not known to be efficiently achievable.) ◀

4 Almost optimal product-state approximation algorithms

In Section 3, we gave upper bounds on $\Gamma_{\alpha,\beta,\gamma}$ for the Heisenberg model. In this section, we give almost matching algorithmic lower bounds on $\Gamma_{\alpha,\beta,\gamma}$ when $\alpha+\beta+\gamma \in \{2,3\}$ (recall $\alpha+\beta+\gamma = 1$ is equivalent to Max-Cut, and so $\Gamma_{\alpha,\beta,\gamma} = 1$). Specifically, we give an approximation algorithm which is almost optimal in the following sense: Given $H \in F_{\alpha,\beta,\gamma}$, it outputs a product state ρ_{prod} with approximation ratio at least 0.649 and 0.498 when $\alpha+\beta+\gamma$ equals 2 and 3, respectively, which by Corollary 5 almost matches the best possible mean-field ratios of 2/3 and 1/2, respectively.

▶ **Theorem 6.** *Let $H \in F_{\alpha,\beta,\gamma}$ for $\alpha,\beta,\gamma \in \{0,1\}$. There exists a randomized, polynomial-time algorithm which obtains approximation ratios at least* 0.878, 0.649 *or* 0.498, *when $\alpha+\beta+\gamma$ equals* 1, 2 *or* 3, *respectively.*

Proof. Suppose H has interaction graph $G = (V, E)$ for $|V| = n$ and edge weights $w_{ij} \geq 0$ for $(i,j) \in E$. We first define a semidefinite programming (SDP) relaxation of $\text{OPT}(H)$ via the first level of the Lasserre hierarchy (see, e.g., [7] for a similar exposition for the setting of low threshold rank graphs). We then show that applying a generalization of the Goemans-Williamson (GW) [19, 12] rounding scheme yields the desired result.

The SDP. Each solution of the SDP relaxation will be a "moment matrix" $M \in \mathbb{R}^{3n\times 3n}$, whose rows (resp., columns) are indexed by 2-tuples $(i,k) \in [n]\times[3]$ (resp., $(j,l) \in [n]\times[3]$) such that ideally, i,j denote qubits, and k,j a choice of Pauli matrix from sequence $(\sigma_1,\sigma_2,\sigma_3) = (X,Y,Z)$. Under this interpretation, an ideal solution M corresponds to a density matrix $\rho \in \mathcal{D}((\mathbb{C}^2)^{\otimes n})$, such that

$$M(ik,jl) = \text{tr}(\rho\sigma_k^i\sigma_l^j), \tag{2}$$

where σ_k^i corresponds to Pauli operator σ_k applied to qubit i, i.e. implicitly we have $\sigma_k^i \otimes I_{[n]\setminus\{i\}}$.

Let us remark about the assumption that M is real. Note that for an ideal solution (i.e. as in Equation (2)), M is Hermitian. Indeed, for $i \neq j$, $M(ik,jl) = M(jl,ik) \in \mathbb{R}$, since the Pauli terms act on different qubits and hence commute. (A similar argument holds for $i = j$ and $k = l$.) If, however, $i = j$ and $k \neq l$, then since the Pauli matrices anti-commute, we have $M(ik,jl) = -M(jl,ik)$, and indeed $M(ik,jl), M(jl,ik) \in \mathbb{C}\setminus\mathbb{R}$ (since, e.g., $XY = iZ$), implying $M(ik,jl)^* = M(jl,ik)$ (thus M is Hermitian; here, * denotes complex conjugate). Note, however, that the case of $i = j$ and $k \neq l$ corresponds to *linear* local terms, i.e. those of the form σ_k^i, and these are the only non-real entries of M. Since our objective function involves only quadratic local terms (i.e. $\sigma_k^i\sigma_l^j$ for $i \neq j$), we can hence eliminate entries of M with $i = j$ and $k \neq l$ by replacing M with moment matrix $M' = (M + M^*)/2$, which is real and matches M on all entries with $i \neq j$ (as well as on $i = j$ and $k = l$). The real symmetric matrix M' is positive semidefinite if the Hermitian M is, and M' results in an equal objective value to that of M, hence the restriction to real moment matrices is without loss of generality.

We have thus far described the ideal solutions, M. Next, we add constraints to the SDP to help enforce this ideal interpretation of M:

1. For all $i \in [n], k \in [3]$, set $M(ik,ik) = 1$, since ideally $M(ik,ik) = \text{tr}(\rho\sigma_k^i\sigma_k^i) = \text{tr}(\rho) = 1$.
2. For all $i \in [n], k \neq l \in [3]$, set $M(ik,il) = -M(il,ik)$, since distinct Pauli matrices anti-commute.

3. Set $M \succeq 0$. This is since, ideally, for all $s \in \mathbb{R}^{3n}$, we have

$$s^T M s = \sum_{ijkl} s_{ik} s_{jl} M(ik, jl) = \mathrm{tr}\left(\rho \left(\sum_{ik} s_{ik}\sigma_k^i\right)\left(\sum_{jl} s_{jl}\sigma_l^j\right)\right) = \mathrm{tr}(\rho S^2) \geq 0, \quad (3)$$

where $S := \sum_{ik} s_{ik}\sigma_k^i$, and since $\rho, S^2 \succeq 0$.

Finally, the relaxed objective function is obtained by replacing each term $\mathrm{tr}(\rho\sigma_k^i\sigma_l^j)$ with $M(ik, jl)$. For example, the relaxed objective function for $F_{1,1,1}$ becomes $\sum_{(i,j)\in E} w_{ij}(1 - M(i1, j1) - M(i2, j2) - M(i3, j3))$.

Let us remark that our formulation is essentially the first level $s = 1$ of the Lasserre SDP hierarchy. Higher levels $s > 1$ are obtained by considering s-local terms for the moment matrices, i.e. $M(i_1k_1, \ldots, i_sk_s) = \mathrm{tr}(\rho\sigma_{k_1}^{i_1} \cdots \sigma_{k_s}^{i_s})$.

Rounding solutions to the SDP. Given any solution M to the SDP, we take the Cholesky decomposition of M to obtain a set of vectors $v_{ik} \in \mathbb{R}^{3n}$ for $i \in [n]$ and $k \in [3]$, such that $M(ik, jl) = v_{ik}^T v_{jl}$. Since $M(ik, ik) = 1$, each v_{ik} is a unit vector. Now, our aim is to round M to a product state solution $\rho_{\mathrm{prod}} = \rho_1 \otimes \cdots \otimes \rho_n$ on n qubits. Thus, writing ρ_i in terms of its Bloch vector $\rho_i = (I + r_{i1}X + r_{i2}Y + r_{i3}Z)/2$ each v_{ik} should be thought of as a $3n$-dimensional relaxation of $r_{ik} \in \mathbb{R}$. For any $v \in \mathbb{R}^p$, $w \in \mathbb{R}^q$, define operation

$$v \circ w = \begin{cases} 0 & \text{if } v = 0 \text{ and } w = 0 \\ v & \text{if } v \neq 0 \text{ and } w = 0 \\ w & \text{if } w \neq 0 \text{ and } v = 0 \\ (v^T, w^T)^T & \text{otherwise,} \end{cases}$$

where $(v^T, w^T)^T \in \mathbb{R}^{p+q}$ denotes the concatenation of v and w. Recalling that $H \in F_{\alpha,\beta,\gamma}$ for $\alpha, \beta, \gamma \in \{0, 1\}$, we now set

$$u_i := (\alpha v_{i1}) \circ (\beta v_{i2}) \circ (\gamma v_{i3}) \in \mathbb{R}^{(\alpha+\beta+\gamma)3n}.$$

This yields first that $w_{ij}(1 - u_i^T u_j)$ equals the term in the relaxed SDP objective function for H corresponding to edge $(i, j) \in E$. For example, if $H \in F_{1,1,0}$ (i.e. the local terms are $w_{ij}(I - X_iX_j - Y_iY_j)$), then $u_i \in \mathbb{R}^{6n}$ and for edge $(i, j) \in E$ we have $M(i1, j1) + M(i2, j2) = u_i^T u_j$. Second, we have $\|u_i\|_2 = \sqrt{\alpha + \beta + \gamma}$.

To obtain the desired claim, define now $x_i = u_i/\|u_i\|_2$. We use a generalization of the Goemans-Williamson (GW) [19] rounding procedure due to Briët, de Oliveira Filho and Vallentin [12]. Specifically, we randomly round each $x_i \in \mathbb{R}^{(\alpha+\beta+\gamma)3n}$ to a Bloch vector $y_i \in \mathbb{R}^{\alpha+\beta+\gamma}$ as follows. Let R be a random $(\alpha + \beta + \gamma) \times (\alpha + \beta + \gamma)3n$ matrix, each of whose entries is chosen independently from a standard normal distribution with mean 0 and variance 1. Then, for each i, set

$$y_i = Rx_i/\|Rx_i\|_2 \in \mathbb{R}^{\alpha+\beta+\gamma}.$$

We map this to a (pure) single-qubit state ρ_i as follows. Let $I(k)$ be the index in sequence (α, β, γ) of the kth non-zero entry (if it exists), for $k \in \{1, 2, 3\}$. Then, set the $I(k)$-th Bloch vector entry of ρ_i to $y_{i,k}$. For example, if $\alpha = \beta = \gamma = 1$, this yields $\rho_i = (I + y_{i,1}X + y_{i,2}Y + y_{i,3}Z)/2$, if $\alpha = \beta = 1$ and $\gamma = 0$, this yields $\rho_i = (I + y_{i,1}X + y_{i,2}Y)/2$, and if $\alpha = \beta = 0$ and $\gamma = 1$, this yields $\rho_i = (I + y_{i,1}Z)/2$ (note the subscript 1 in $y_{i,1}$). For ease of exposition, henceforth we refer to the Bloch vector for ρ_i as $\mathbf{r_i} = (r_1, r_2, r_3)$, where the entries of $\mathbf{r_i}$ which are not set in the rounding scheme above being implicitly set to 0. For example, $\mathbf{r_i} = (y_{i,1}, y_{i,2}, y_{i,3})$, $\mathbf{r_i} = (y_{i,1}, y_{i,2}, 0)$, and $\mathbf{r_i} = (0, 0, y_{i,1})$, respectively, in the examples above.

Approximation ratio. To analyze the approximation ratio obtained, note that for edge $(i,j) \in E$, we have

$$\begin{aligned} w_{ij}\operatorname{tr}(H_{ij}\rho_{\text{prod}}) &= \frac{w_{ij}}{4}\operatorname{tr}(H_{ij}(I + r_{i,1}X_i + r_{i,2}Y_i + r_{i,3}Z_i)(I + r_{j,1}X_j + r_{j,2}Y_j + r_{j,3}Z_j)) \\ &= \frac{w_{ij}}{4}\operatorname{tr}((I - \alpha X_iX_j - \beta Y_iY_j - \gamma Z_iZ_j)\cdot \\ &\qquad (I + r_{i,1}X_i + r_{i,2}Y_i + r_{i,3}Z_i)(I + r_{j,1}X_j + r_{j,2}Y_j + r_{j,3}Z_j)) \\ &= w_{ij}(1 - \alpha r_{i,1}r_{j,1} - \beta r_{i,2}r_{j,2} - \gamma r_{i,3}r_{j,3}) \\ &= w_{ij}(1 - y_i^T y_j). \end{aligned}$$

On the other hand, recall the SDP obtains value $w_{ij}(1 - u_i^T u_j)$ on edge $(i,j) \in E$. For brevity, let $F[r,\, u^T v]$ denote the right hand side of Equation 12 (Lemma 10 in Appendix C). A direct application of Lemma 10 yields $\mathbb{E}[y_i^T y_j] = F[\alpha+\beta+\gamma,\, x_i^T x_j]$. Then, by linearity of expectation, the expected approximation ratio is given by the expected ratio attained on each edge, which is

$$\frac{1 - \mathbb{E}[y_i^T y_j]}{1 - u_i^T u_j} = \frac{1 - F[\alpha+\beta+\gamma,\, x_i^T x_j]}{1 - u_i^T u_j} = \frac{1 - F[\alpha+\beta+\gamma,\, t]}{1 - (\alpha+\beta+\gamma)t},$$

where we defined $t = x_i^T x_j$ (note the value of F only depends on t; see Appendix C). Numerically evaluating via Mathematica (see Appendix C for Mathematica code)

$$\min_{t \in [-1, 1/(\alpha+\beta+\gamma))} \frac{1 - F[t]}{1 - (\alpha+\beta+\gamma)t},$$

we obtain ratios of 0.878 (for $\alpha+\beta+\gamma = 1$), 0.649 (for $\alpha+\beta+\gamma = 2$), and 0.498 (for $\alpha+\beta+\gamma = 3$), respectively. Note we minimize over $t \in [-1, 1/(\alpha+\beta+\gamma))$, since for $t \in [1/(\alpha+\beta+\gamma), 1]$ the ratio can only be negative (the denominator is negative, and the numerator is in range $[0,2]$). This completes the proof. ◀

4.1 Generalizations beyond the Heisenberg model

We defined the Heisenberg model $F_{\alpha,\beta,\gamma}$ in Section 2 as having all constraints identical with some fixed $(\alpha,\beta,\gamma) \in \{0,1\}^3$. We now show how to extend the algorithm to two more general settings: The first will allow different choices of $\alpha_{ij}, \beta_{ij}, \gamma_{ij} \in [1,-1]$ on each edge (note the use of $[1,-1]$ instead of $\{0,1\}$), and the second will require that all constraints remain identical but in exchange allows new interaction terms beyond XX, YY, ZZ.

4.1.1 Approximating Heisenberg models with varying Pauli weights

The approximation algorithm developed in the previous section made critical use of the fact that $\alpha,\beta,\gamma \in \{0,1\}$ for our Heisenberg model $F_{\alpha,\beta,\gamma}$. Here, we generalize by allowing two relaxations, captured below in the form of constraints now allowed:

$$H_{ij} = w_{ij}(I - \alpha_{ij}X_iX_j - \beta_{ij}Y_iY_j - \gamma_{ij}Z_iZ_j),$$

where $\alpha_{ij}, \beta_{ij}, \gamma_{ij} \in [-1,1]$. The two relaxations to note are (1) $\alpha_{ij}, \beta_{ij}, \gamma_{ij} \in [-1,1]$ instead of in $\{0,1\}$, and (2) each edge $(i,j) \in E$ may have a different choice of $\alpha_{ij}, \beta_{ij}, \gamma_{ij}$ In this setting, we shall use the same relaxation as Section 4, but utilize another rounding strategy. In exchange for the added generality, the approximation ratios obtained are slightly weaker than those of Section 4.

In the theorem below, for brevity we call the sets $\{\alpha_{ij}\}, \{\beta_{ij}\}, \{\gamma_{ij}\}$ *parameter families*. We say a parameter family is *non-zero* if at least one parameter in the family is non-zero, e.g. there exists $(i,j) \in E$ such that $\alpha_{ij} \neq 0$ for family $\{\alpha_{ij}\}$.

▶ **Theorem 7.** *Let $H = \sum_{(i,j)\in E} H_{ij}$ be a 2-local Hamiltonian on qubits with constraints*

$$H_{ij} = w_{ij}(I - \alpha_{ij} X_i X_j - \beta_{ij} Y_i Y_j - \gamma_{ij} Z_i Z_j),$$

where $\alpha_{ij}, \beta_{ij}, \gamma_{ij} \in [-1,1]$ and $w_{ij} \in \mathbb{R}^+$. There exists a randomized, polynomial-time algorithm which obtains approximation ratio at least 0.878 *(if precisely one parameter family is non-zero),* 0.609 *(if precisely two parameter families are non-zero), and* 0.462 *(if all three parameter families are non-zero).*

Proof. We begin by mapping H to a "canonical" form.

Setup in "canonical" form. For now, assume $\alpha_{ij}, \beta_{ij}, \gamma_{ij} \neq 0$ (later we will get improved ratios when some of these values are 0 for every $(i,j) \in E$). Our first observation is that we may assume $\alpha_{ij}, \beta_{ij}, \gamma_{ij} \in \{-1,1\}$. This is because any vector $(\alpha_{ij}, \beta_{ij}, \gamma_{ij}) \in [-1,1]^3$ is a convex combination of vectors with coordinates in $\{-1,+1\}$ (i.e. the former lies in the convex hull of discrete points $(x,y,z) \in \{-1,1\}^3$). Thus any H_{ij} of the above form may be expressed as convex combination,

$$H_{ij} = \sum_{k=1}^{4} w_{ij}\lambda_k (I - \alpha_{ij,k} X_i X_j - \beta_{ij,k} Y_i Y_j - \gamma_{ij,k} Z_i Z_j), \tag{4}$$

with $\alpha_{ij,k}, \beta_{ij,k}, \gamma_{ij,k} \in \{-1,1\}$, and $\lambda_k \geq 0$ with $\sum_{k=1}^{4} \lambda_k = 1$. Notes: (1) Since we allow multiple edges between i and j, we may include an edge for each term of the convex combination. (2) That we require at most 4 terms λ_k follows from Carathéodory's theorem, which says that a point in $\mathbb{R}^d$ in the convex hull of some set P requires at most $d+1$ points of P to express as a convex combination. (3) Our approximation ratio analysis below will again be via expectation per edge, which by linearity of expectation yields that no loss in approximation is incurred by writing our constraints as in Equation (4).

Rounding algorithm. We employ the same moment SDP relaxation as in Section 4, and continue to use the terminology therein. Consider the vectors $v_{i1}, v_{i2}, v_{i3} \in \mathbb{R}^{3n}$ corresponding to an optimal solution of the SDP relaxation. The objective value of the relaxation is $w_{\text{SDP}} := \sum_{(i,j)\in E} w_{ij}(1 - \alpha_{ij} v_{i1}^T v_{j1} - \beta_{ij} v_{i2}^T v_{j2} - \gamma_{ij} v_{i3}^T v_{j3})$. Now suppose, without loss of generality (any other ordering is handled analogously):

$$-\sum_{(i,j)\in E} w_{ij}\gamma_{ij} v_{i3}^T v_{j3} \geq -\sum_{(i,j)\in E} w_{ij}\beta_{ij} v_{i2}^T v_{j2} \geq -\sum_{(i,j)\in E} w_{ij}\alpha_{ij} v_{i1}^T v_{j1},$$

so that

$$\sum_{(i,j)\in E} w_{ij}(1 - 3\gamma_{ij} v_{i3}^T v_{j3}) \geq w_{\text{SDP}}. \tag{5}$$

Recall that the v_{i3} are unit vectors (since our SDP had constraint $M(ik,ik) = 1$ for all i,k). Hence, we may view the v_{i3} as a feasible solution for the Max-Cut SDP relaxation of Goemans and Williamson and consequently, use the same rounding algorithm [19]:

1. Select a random vector $r \in \mathbb{R}^{3n}$ with each entry independently and normally distributed with mean 0 and variance 1.
2. Let $r_i = r^T v_{i3}/|r^T v_{i3}| \in \{-1, 1\}$.
3. Output the product state, $\prod_i \frac{1}{2}(I + r_i Z_i)$.

Note that since the assignment above is diagonal in the Z basis (i.e. is a standard basis state), it lies in the null space of each XX and YY term of our Hamiltonian. Consequently, our expected objective value for this assignment on our Hamiltonian is

$$w_{\text{EXP}} := \sum_{(i,j)\in E} \mathbb{E}[w_{ij}(1 - \gamma_{ij} r_i r_j)] = \sum_{(i,j)\in E} w_{ij}(1 - \gamma_{ij} 2 \arcsin\big(v_{i3}^T v_{j3}\big)/\pi),$$

where the second equality follows by (1) linearity of expectation and (2) the standard analysis of the Goemans-Williamson algorithm [19], which states that $E[r_i r_j] = 2 \arcsin\big(v_{i3}^T v_{j3}\big)/\pi$.

Approximation ratio. We conclude by bounding the expected approximation ratio, $w_{\text{EXP}}/w_{\text{SDP}}$. As for the analysis of the algorithm from the previous section, we need only consider the worst-case behavior on any edge. Using (5), this is:

$$\min_{\gamma\in\{-1,1\},\, t\in[-1,1]:3\gamma t<1} \frac{1 - \gamma 2 \arcsin(t)/\pi}{1 - 3\gamma t},$$

where γ represents γ_{ij}, and t represents $v_{i3}^T v_{j3}$. Numerically, this yields a ratio of 0.462. A similar analysis produces an approximation ratio of 0.609 for the case when either $\alpha_{ij} = 0$ for all $(i,j) \in E$, $\beta_{ij} = 0$ for all $(i,j) \in E$, or $\gamma_{ij} = 0$ for all $(i,j) \in E$. We recover the Goemans-Williamson 0.878-approximation in the case when two of these parameters are 0 for all $(i,j) \in E$. ◀

4.1.2 Reductions via local unitaries

We now generalize the algorithm of Section 4 in a different manner. Specifically, using a standard trick from entanglement theory (used also in [14] in a somewhat different manner), we may give an approximation-preserving reduction to the Heisenberg model in certain cases. Namely, recall that any two-qubit Hermitian operator H_{ij} can be expanded in the Pauli basis as follows (sometimes known as the *Fano form* [15]), given by:

$$H_{ij} = \kappa I + \sum_{a=1}^{3}\sum_{b=1}^{3} M_{ab}\sigma_a \otimes \sigma_b + \sum_{a=1}^{3} r_a \sigma_a \otimes I + \sum_{b=1}^{3} s_b I \otimes \sigma_b, \tag{6}$$

where $\kappa, M_{ab}, r_a, s_b \in \mathbb{R}$. The 3×3 real matrix M, which has no particular structure in general (for example, it need not be diagonalizable), is called the *correlation matrix* in entanglement theory.

▶ **Theorem 8.** *Let H be a 2-local Hamiltonian on n qubits, and with directed interaction graph $G = (V, E)$, where $H = \sum_{(i,j)\in E} w_{ij} H_{ij}$ for non-negative real weights w_{ij}. Assume*

1. *all H_{ij} are identical with $\kappa = r_1 = r_2 = r_3 = s_1 = s_2 = s_3 = 0$, and*
2. *the correlation matrix M of H_{ij} is an orthogonal projection (i.e. M is symmetric with $M^2 = M$).*

Then, there exists a randomized, polynomial-time algorithm which obtains approximation ratios at least 0.878, 0.649 or 0.498, when the rank of M equals 1, 2 or 3, respectively. Conversely, the best possible product-state ratio (not necessarily efficiently attainable) in each case is 1, 2/3, and 1/2, respectively.

Proof. We use the approach of [22, 23] of simulating orthogonal rotations on M via local unitary operations on H_{ij}. Namely, due to the surjective homomorphism from SU(2) to SO(3), if one wishes to map M to $O_1 M O_2^T$ for orthogonal matrices O_1 and O_2, there exist single-qubit unitaries U and V such that $U_i \otimes V_j H_{ij} U_i^\dagger \otimes V_j^\dagger$ has correlation matrix $O_1 M O_2^T$. Since M is symmetric, it is diagonalizable by an orthogonal matrix $O \in \mathbb{R}^{3\times 3}$ (Corollary 2.5.14 of [21]). Thus, there exists a single-qubit unitary U such that $U_i \otimes U_j H_{ij} U_i^\dagger \otimes U_j^\dagger$ has a diagonal correlation matrix with eigenvalues from set $\{0, 1\}$. Since all constraints H_{ij} are identical, it follows that $U^{\otimes n} H (U^\dagger)^{\otimes n}$ is a Hamiltonian in family $F_{\alpha,\beta,\gamma}$ for some $\alpha, \beta, \gamma \in \{0, 1\}$. The algorithm of Theorem 6 now yields the claimed lower bound on approximation. The claimed upper bound on approximation follows from Corollary 5. In both cases, we are leveraging the fact that our reduction applies only single-qubit unitary operations, and hence perfectly preserves approximation ratios attained by product states. ◀

Note that Theorem 8 uses the algorithm of Section 4. If we are willing to obtain slightly worse approximation ratios, we can relax the second requirement of Theorem 8 by instead applying the algorithm of Section 4.1.1.

▶ **Theorem 9.** *Let H be a 2-local Hamiltonian on n qubits, and with directed interaction graph $G = (V, E)$, where $H = \sum_{(i,j)\in E} w_{ij} H_{ij}$ for non-negative real weights w_{ij}. Assume*

1. *all H_{ij} are identical with $\kappa = r_1 = r_2 = r_3 = s_1 = s_2 = s_3 = 0$, and*
2. *the correlation matrix M of H_{ij} is symmetric.*

Then, there exists a randomized, polynomial-time algorithm which obtains approximation ratios at least 0.878, 0.609 or 0.462, when the rank of M equals 1, 2 or 3, respectively. Conversely, the best possible product-state ratio (not necessarily efficiently attainable) in each case is 1, 2/3, and 1/2, respectively.

The proof is identical to that of Theorem 8, except we using the rounding algorithm of Section 4.1.1 instead; we hence omit the proof.

References

1. D. Aharonov, I. Arad, Z. Landau, and U. Vazirani. The detectibility lemma and quantum gap amplification. In *Proceedings of 41st ACM Symposium on Theory of Computing (STOC 2009)*, volume 287, pages 417–426, 2009.
2. Dorit Aharonov, Itai Arad, and Thomas Vidick. Guest Column: The Quantum PCP Conjecture. *SIGACT News*, 44(2):47–79, June 2013. doi:10.1145/2491533.2491549.
3. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. Prelim. version FOCS '92.
4. S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Prelim. version FOCS '92.
5. N. Bansal, S. Bravyi, and B. M. Terhal. Classical approximation schemes for the ground-state energy of quantum and classical Ising spin Hamiltonians on planar graphs. *Quantum Information & Computation*, 9(7&8):0701–0720, 2009.
6. H. Bethe. Zur Theorie der Metalle. *Zeitschrift für Physik*, 71(3–4):205–226, 1931.
7. F. Brandão and A. Harrow. Product-state Approximations to Quantum Ground States. In *Proceedings of the 45th ACM Symposium on the Theory of Computing (STOC 2013)*, pages 871–880, 2013.
8. S. Bravyi. Monte Carlo simulation of stoquastic Hamiltonians. *Quantum Information & Computation*, 15(13&14):1122–1140, 2015.

9 S. Bravyi, D. Gosset, R. Koenig, and K. Temme. Approximation algorithms for quantum many-body problems. Available at arXiv.org e-Print quant-ph/arXiv:1808.01734, 2018. arXiv:1808.01734.

10 S. Bravyi and M. Hastings. On complexity of the quantum Ising model. *Communications in Mathematical Physics*, 349(1):1–45, 2014.

11 Sergey Bravyi and David Gosset. Polynomial-Time Classical Simulation of Quantum Ferromagnets. *Physical Review Letters*, 119:100503, September 2017. doi:10.1103/PhysRevLett.119.100503.

12 J. Briët, F. M. de Oliveira Filho, and F. Vallentin. Grothendieck inequalities for semidefinite programs with rank constraint. *Theory of Computing*, 10:77–105, 2014.

13 Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending Grothendieck's inequality. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 54–60. IEEE, 2004.

14 T. Cubitt and A. Montanaro. Complexity classification of local Hamiltonian problems. *SIAM Journal on Computing*, 45(2):268–316, 2016.

15 U. Fano. Pairs of two-level systems. *Reviews of Modern Physics*, 55:855–874, 1983.

16 S. Gharibian and J. Kempe. Approximation algorithms for QMA-complete problems. *Siam Journal on Computing*, 41(4):1028–1050, 2012.

17 S. Gharibian and J. Kempe. Hardness of approximation for quantum problems. In *Proceedings of 39th International Colloquium on Automata, Languages and Programming (ICALP 2012)*, pages 387–398, 2012. © 2012 Springer, www.springerlink.com. doi:10.1007/978-3-642-31594-7.

18 Sevag Gharibian, Yichen Huang, Zeph Landau, and Seung Woo Shin. Quantum Hamiltonian Complexity. *Foundations and Trends® in Theoretical Computer Science*, 10(3):159–282, 2014. doi:10.1561/0400000066.

19 M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.

20 D. Hochbaum. *Approximation Algorithms for NP-Hard Problems*. Wadsworth Publishing Company, 1997.

21 R. A. Horn and C. H. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

22 R. Horodecki and M. Horodecki. Information-theoretic aspects of quantum inseparability of mixed states. *Physical Review A*, 54(3):1838–1843, 1996.

23 R. Horodecki and P. Horodecki. Perfect correlations in the Einstein-Podolsky-Rosen experiment and Bell's inequalities. *Physics Letters A*, 210:227, 1996.

24 A. Kitaev, A. Shen, and M. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, 2002.

25 E. Lee and S. Hallgren. Approximation of MAX-2-local Hamiltonians. To be presented at the 19th Asian Quantum Information Science Conference (AQIS), 2019.

26 A. Natarajan and T. Vidick. Low-degree testing for quantum states, and a quantum entangled games PCP for QMA. In *Proceedings of the 59th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 731–742, 2018.

27 M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

28 T. J. Osborne. Hamiltonian complexity. *Reports on Progress in Physics*, 75(2):022001, 2012. URL: http://stacks.iop.org/0034-4885/75/i=2/a=022001.

29 Stephen Piddock and Ashley Montanaro. The Complexity of Antiferromagnetic Interactions and 2D Lattices. *Quantum Information & Computation*, 17(7-8):636–672, June 2017. URL: http://dl.acm.org/citation.cfm?id=3179553.3179559.

30 V. Vazirani. *Approximation Algorithms*. Springer, 2001.

A Max Cut as a special case of the Heisenberg model

We briefly sketch why local constraints $H_{ij} = I - Z_i \otimes Z_j$ in the Heisenberg model yield the NP-complete problem Max Cut. Namely, the Pauli Z operator

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

is diagonal in the standard basis with eigenvalues 1 for eigenvector $|0\rangle$ and -1 for eigenvector $|1\rangle$. It follows that $Z \otimes Z$ also diagonalizes in the standard basis, with eigenvectors $|00\rangle$ and $|11\rangle$ attaining eigenvalue 1 and $|01\rangle$ and $|10\rangle$ attaining eigenvalue -1. As a result, operator $I - Z_i \otimes Z_j$ has eigenvalues 0 (with eigenspace spanned by $|00\rangle$ and $|11\rangle$) and 2 (with eigenspace spanned by $|01\rangle$ and $|10\rangle$). But this means that on each edge $(i,j) \in E$,

$$\begin{aligned} \langle 00|I - Z_i \otimes Z_j|00\rangle &= \langle 11|I - Z_i \otimes Z_j|11\rangle = 0, \text{ and} \\ \langle 01|I - Z_i \otimes Z_j|01\rangle &= \langle 10|I - Z_i \otimes Z_j|10\rangle = 2. \end{aligned}$$

In other words, if neighboring qubits are set to opposing standard basis states (e.g. $|01\rangle$), then we obtain value 2 from an edge, and if the qubits are set to identical standard basis states (e.g. $|00\rangle$), we obtain value 0 from this edge. Finally, since all local terms are diagonal in the standard basis, the entire Hamiltonian $H = \sum_{(i,j)\in E} H_{ij}$ will also be diagonal in the standard basis. The largest eigenvalue of H will hence be the sum of the values obtained on each edge by the best standard basis state, which will correspond to a maximum cut in the graph. The actual largest eigenvalue will equal twice the maximum cut on the underlying graph (since we obtain value 2 on each cut edge, rather than 1 as for the standard Max Cut problem).

B Proofs for Section 2

Proof of Lemma 2. Observe that for standard basis vectors $|i\rangle, |j\rangle, |k\rangle, |l\rangle \in \mathbb{C}^2$, we have

$$\langle ij|X \otimes X|kl\rangle = (i \oplus k)(j \oplus l), \tag{7}$$

$$\langle ij|Y \otimes Y|kl\rangle = (-1)^{\delta_{kl}}(i \oplus k)(j \oplus l), \tag{8}$$

$$\langle ij|Z \otimes Z|kl\rangle = (-1)^{k \oplus l}\delta_{ik}\delta_{jl}, \tag{9}$$

were δ_{ij} is the usual Kronecker delta. Denoting an arbitrary product state as $|\psi\rangle = ac|00\rangle + ad|01\rangle + bc|10\rangle + bd|11\rangle$ for $|a|^2 + |b|^2 = |c|^2 + |d|^2 = 1$, we have

$$\begin{aligned} \langle\psi|H|\psi\rangle &= \alpha(a^*c^*bd + acb^*d^* + a^*d^*bc + adb^*c^*) + \\ &\quad \beta(-a^*c^*bd - acb^*d^* + a^*d^*bc + adb^*c^*) + \\ &\quad \gamma(|a|^2|c|^2 + |b|^2|d|^2 - |a|^2|d|^2 - |b|^2|c|^2) \qquad (10) \\ &= 2\,\mathrm{Re}[acb^*d^*](\alpha - \beta) + 2\,\mathrm{Re}[adb^*c^*](\alpha + \beta) + \\ &\quad \gamma(|a|^2|c|^2 + |b|^2|d|^2 - |a|^2|d|^2 - |b|^2|c|^2) \\ &\leq 2|a||b||c||d|(|\alpha + \beta| + |\alpha - \beta|) + \\ &\quad |\gamma|\Big|(|a|^2 - |b|^2)(|c|^2 - |d|^2)\Big|. \end{aligned}$$

where the last inequality follows from the triangle inequality. Let us simplify the notation above by assuming without loss of generality $a, b, c, d \in \mathbb{R}^+$. We may also assume without loss of generality that $a \geq b$ and $c \geq d$ (since this maximizes the upper bound). Thus:

$$\langle\psi|H|\psi\rangle \leq 2abcd(|\alpha + \beta| + |\alpha - \beta|) + |\gamma|(a^2 - b^2)(c^2 - d^2).$$

Note now for any $\alpha, \beta \in \mathbb{R}$, $|\alpha+\beta| + |\alpha-\beta| = ||\alpha|+|\beta|| + ||\alpha|-|\beta||$. Assume first $|\alpha| \geq |\beta|$. Then

$$\langle\psi|H|\psi\rangle \leq 4abcd|\alpha| + |\gamma|(a^2-b^2)(c^2-d^2). \tag{11}$$

Let $p = 4abcd$ and $q = (a^2-b^2)(c^2-d^2)$. Note $p, q \geq 0$. Also, we claim $p+q \leq 1$; this will imply $\langle\psi|H|\psi\rangle \leq \max(|\alpha|, |\gamma|)$. To see this claim, note

$$p+q = (ac+bd)^2 - (ad-bc)^2 \leq (ac+bd)^2 \leq 1,$$

where the last inequality follows from the Cauchy-Schwarz inequality. The case of $|\beta| \geq |\alpha|$ follows analogously with $|\alpha|$ in Equation (11) replaced with $|\beta|$. We hence have $\langle\psi|H|\psi\rangle \leq \max(|\alpha|, |\beta|, |\gamma|) = \|(|\alpha|, |\beta|, |\gamma|)\|_\infty$.

We now show matching lower bounds, i.e. that $|\alpha|$, $|\beta|$, and $|\gamma|$ are attainable. Returning to Equation (10):

- For $|\alpha|$: If $\alpha \geq 0$, set $a = b = c = d = 1/\sqrt{2}$, and if $\alpha < 0$, set $a = b = c = 1/\sqrt{2}$ and $d = -1/\sqrt{2}$.
- For $|\beta|$: If $\beta \geq 0$, set $a = i/\sqrt{2}$, $c = i/\sqrt{2}$, $b = d = 1\sqrt{2}$, and if $\beta < 0$, set $a = -i/\sqrt{2}$, $c = i/\sqrt{2}$, $b = d = 1\sqrt{2}$.
- For $|\gamma|$: If $\gamma \geq 0$, set $a = c = 1$ and $b = d = 0$. and of $\gamma < 0$, set $a = d = 1$, $b = c = 0$.

◀

Proof of Lemma 3. Denoting an arbitrary two-qubit state as $|\psi\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$ for $|a|^2 + |b|^2 = |c|^2 + |d|^2 = 1$, we have via Equations (7)-(9) that

$$\begin{aligned}
\langle\psi|X\otimes X|\psi\rangle &= a^*d + ad^* + b^*c + bc^*, \\
\langle\psi|Y\otimes Y|\psi\rangle &= -a^*d - ad^* + b^*c + bc^*, \\
\langle\psi|Z\otimes Z|\psi\rangle &= |a|^2 - |b|^2 - |c|^2 + |d|^2.
\end{aligned}$$

Thus, $\langle\psi|H|\psi\rangle$ equals

$$\begin{aligned}
& \alpha(2\operatorname{Re}(ad^*) + 2\operatorname{Re}(bc^*)) + \beta(-2\operatorname{Re}(ad^*) + 2\operatorname{Re}(bc^*)) + \gamma(|a|^2 + |d|^2 - |b|^2 - |c|^2) \\
= \;& 2\operatorname{Re}[ad^*](\alpha-\beta) + 2\operatorname{Re}[bc^*](\alpha+\beta) + (|a|^2 + |d|^2 - |b|^2 - |c|^2)\gamma.
\end{aligned}$$

Observe that since the coefficient of γ depends on only *absolute values* of a, b, c, d, we can assume without loss of generality that the optimal assignment has $a, b, c, d \geq 0$ and satisfies

$$\langle\psi|H|\psi\rangle = 2ad|\alpha-\beta| + 2bc|\alpha+\beta| + (a^2+d^2-b^2-c^2)\gamma.$$

By applying the Arithmetic-Geometric mean inequality, we hence have

$$\begin{aligned}
\langle\psi|H|\psi\rangle &\leq (a^2+d^2)|\alpha-\beta| + (b^2+c^2)|\alpha+\beta| + (a^2-b^2-c^2+d^2)\gamma \\
&= (a^2+d^2)(|\alpha-\beta|+\gamma) + (b^2+c^2)(|\alpha+\beta|-\gamma) \\
&\leq \max(|\alpha-\beta|+\gamma, |\alpha+\beta|-\gamma),
\end{aligned}$$

where the last statement follows since $a^2+b^2+c^2+d^2 = 1$. The matching lower bound is obtained as follows. To achieve $|\alpha-\beta|+\gamma$ when $\alpha \geq \beta$, set $a = d = 1/\sqrt{2}$, and when $\alpha \leq \beta$, set $a = 1/\sqrt{2}, d = -1/\sqrt{2}$. Similarly, to achieve $|\alpha+\beta|-\gamma$ when $\alpha \geq -\beta$, set $b = c = 1/\sqrt{2}$, and when $\alpha \leq -\beta$, set $b = 1/\sqrt{2}, c = -1/\sqrt{2}$. ◀

C Lemmas and Mathematica code

In Section 4 we use the following lemma, which is stated as given in [12]. Below, ${}_2F_1(a,b;c;z)$ is the hypergeometric function, defined for $|z|<1$ as

$$ {}_2F_1(a,b;c;z) = \sum_{n=0}^{\infty} \frac{(a)_n (b)_n}{(c)_n} \frac{z^n}{n!},$$

where for $n \geq 0$, we have Pochhammer symbol $(x)_n = \Gamma(x+n)/\Gamma(x) = x(x+1)\cdots(x+n-1)$ for Γ the Gamma function.

▶ **Lemma 10** (Briët, de Oliveira Filho and Vallentin [12]). *Let u, v be unit vectors in $\mathbb{R}^n$ and let $Z \in \mathbb{R}^{r\times n}$ be a random matrix whose entries are distributed independently according to the standard normal distribution with mean 0 and variance 1. Then,*

$$\mathbb{E}\left[\frac{Zu}{\|Zu\|_2}\cdot\frac{Zv}{\|Zv\|_2}\right] = \frac{2}{r}\left(\frac{\Gamma((r+1)/2)}{\Gamma(r/2)}\right)^2 (u\cdot v)\ {}_2F_1\left(1/2,1/2;r/2+1;(u\cdot v)^2\right). \quad (12)$$

Mathematica code

Below, we give the Mathematica code used to numerically calculate the approximation ratios of Theorem 6:

```
g[r_] := 2/r (Gamma[(r + 1)/2]/Gamma[r/2])^2
F[r_, t_] := g[r] t Hypergeometric2F1[1/2, 1/2, r/2 + 1, t^2]
ApproxRatio[r_] := Min[Select[ Table[(1 - F[r, t])/(1 - r t),
                                          {t, -1, 1/r, 0.01}], # > 0 &]]
ApproxRatio[1]
ApproxRatio[2]
ApproxRatio[3]
```

The code for the approximation ratios in Section 4.1.1 is:

```
ApproxRatio[r_] :=
Min[Select[ Flatten[Table[(1 - g 2 ArcSin[t]/Pi)/(1 - r g t),
                                  {g, -1, 1}, {t, -1, 1, 0.01}]], # > 0 &]]
ApproxRatio[1]
ApproxRatio[2]
ApproxRatio[3]
```

Maximizing Covered Area in the Euclidean Plane with Connectivity Constraint

Chien-Chung Huang
DI ENS, École Normale supérieure, Université PSL, Paris, France
CNRS, Paris, France
chien-chung.huang@ens.fr

Mathieu Mari
DI ENS, École Normale supérieure, Université PSL, Paris, France
mathieu.mari@ens.fr

Claire Mathieu
CNRS, Paris, France
clairemmathieu@gmail.com

Joseph S. B. Mitchell
Stony Brook University, Stony Brook, NY 11794, USA
joseph.mitchell@stonybrook.edu

Nabil H. Mustafa
Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge, ESIEE Paris, France
mustafan@esiee.fr

Abstract

Given a set $\mathcal{D}$ of n unit disks in the plane and an integer $k \leq n$, the maximum area connected subset problem asks for a set $\mathcal{D}' \subseteq \mathcal{D}$ of size k that maximizes the area of the union of disks, under the constraint that this union is connected. This problem is motivated by wireless router deployment and is a special case of maximizing a submodular function under a connectivity constraint.

We prove that the problem is NP-hard and analyze a greedy algorithm, proving that it is a $\frac{1}{2}$-approximation. We then give a polynomial-time approximation scheme (PTAS) for this problem with resource augmentation, i.e., allowing an additional set of εk disks that are not drawn from the input. Additionally, for two special cases of the problem we design a PTAS without resource augmentation.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases approximation algorithm, submodular function optimisation, unit disk graph, connectivity constraint

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.32

Category APPROX

Related Version `https://www.di.ens.fr/~mmari/content/papers/MACS.pdf`

Funding *Joseph S. B. Mitchell*: Partially supported by the National Science Foundation (CCF-1540890), the US-Israel Binational Science Foundation (CCF-1540890), and DARPA (Lagrange).
Nabil H. Mustafa: The work of Nabil H. Mustafa in this paper has been supported by the grant ANR SAGA (JCJC-14-CE25-0016-01).

1 Introduction

Maximizing a submodular function[1] under constraints is a classical problem in computer science and operations research [8, 23]; the most commonly studied constraints are cardinality, knapsack and matroids constraints. A natural constraint that has received little attention is

[1] Given a set X, a function $f : 2^X \to \mathbb{R}$ is *submodular* if given any two subsets $A, B \subseteq X$, $f(A) + f(B) \geq f(A \cap B) + f(A \cup B)$.


© Chien-Chung Huang, Mathieu Mari, Claire Mathieu, Joseph S. B. Mitchell, and Nabil H. Mustafa; licensed under Creative Commons License CC-BY

the *connectivity* constraint. In this paper, we study the following problem: given a set of n unit disks in the plane, select a subset of k disks that maximize the area of their union, under the constraint that this union is connected. We call this problem the *Maximum Area Connected Subset* problem (MACS). Notice that the area covered by a set of disks is a monotone submodular function.

The problem is motivated by wireless router deployment, first introduced in [14], where the goal is to install a given number of routers to maximize the number of clients covered. When the clients are uniformly spread in the plane, the number of clients in a region can be approximated by the area of that region, leading to our problem.

Our Contributions

We first analyze a variant of the greedy algorithm and show that it computes a $\frac{1}{2}$-approximation to MACS (Theorem 3); further, the analysis is tight. In contrast, the natural algorithm that greedily adds one disk at a time can end up with a solution with area a factor of $\Omega(k)$ worse than the optimal solution.

To improve upon the $\frac{1}{2}$-approximation ratio, we turn to the resource augmentation setting in which the algorithm is allowed to add a few additional disks that need not be drawn from the input. We design a PTAS for the resource augmentation version of the problem using Arora's shifted quadtree technique (Theorem 4)[2]. The proof hinges on the existence of a near-optimal solution with $O(\varepsilon k)$ additional disks and with additional structure that allows its computation by dynamic programming.

As a corollary, we show that for two special cases of MACS we can in fact design a PTAS without resource augmentation: *i*) when the Euclidean distances between centers of the input disks are well-approximated by shortest paths in their intersection graph (Corollary 6), and *ii*) when every point of the relevant region of the Euclidean plane is covered by at least one input disk (Corollary 9).

On the other hand, via a reduction from the Rectilinear Steiner Tree problem, we show that MACS is NP-hard (Theorem 3). We also show that MACS for the input of a set of quadrilaterals instead of disks, the problem is APX-hard (Theorem 12). We leave open the question whether MACS is APX-hard or admits a PTAS without resource augmentation.

Related work

Maximising a monotone submodular function under constraint(s) is a subject that has received a large amount of attention over the years. We refer the readers to [2, 5, 6, 8, 13, 15, 23] and the references therein. Our problem can be regarded as maximising a submodular function under a cardinality (knapsack) constraint and a connectivity constraint. Notice that the connectivity constraint is central to the difficulty of our problem: without connectivity constraints, MACS admits a PTAS even in the more general case of convex pseudodisks [4]. However even without the connectivity constraint the problem remains NP-hard[3].

Another motivation for studying the connectivity constraint is related to cancer genome studies. Suppose that a vertex represents an individual protein (and associated gene), an edge represents pairwise interactions, and each vertex has an associated set. Finding the connected subgraph of k genes that is mutated in the largest number of samples is equivalent to the problem of finding the connected subgraph with k nodes that maximizes the cardinality of the union of the associated sets (see [21]).

---

[2] We also develop an alternative algorithm using Mitchell's m-guillotine dissection technique. See the full version for details.

[3] The reduction is from Maximum independent set problem that is NP-hard in unit-disk graphs.

In the general (non-geometric) setting, there exists a $O\left(\frac{1}{\sqrt{k}}\right)$-approximation algorithm for maximizing a monotone submodular function [14]. Our results show that when the submodular function and the connectivity are induced by a geometric configuration, the approximation ratio can be significantly improved.

We next consider several related problems where the connectivity constraint plays an important role. The goal of the node-cost budget problem [20] is to find a connected set of vertices in a general graph to collect the maximum profit on the vertices while guaranteeing the total cost does not exceed a certain budget. Notice that in this setting the submodular function is a simple additive function of the profits. Another related problem [3] is to assign radii to a given set of points in the plane so that the resulting set of disks is connected and the objective is to minimize the sum of radii.

Khuller et al. [12] study the budgeted connected dominating set problem where given a general undirected graph, the goal is to select k vertices whose induced subgraph is connected and that maximizes the number of dominated vertices. It was pointed out to us that their algorithm can be used to give a $\frac{1}{13}\left(1-\frac{1}{e}\right)$-approximate solution for MACS. The authors of [10] consider the problem of selecting k nodes of an input node-weighted graph to form a connected subgraph, with the aim of maximizing or minimizing the selected weight.

We now turn to the geometric setting. A logarithmic-factor approximation algorithm is known [9] for the connected sensor coverage problem in which one must select at most k sensors in the plane forming a connected communication network and covering the desired region, where the region covered by each sensor is convex [7, 11]. A $(1-\varepsilon)$-approximation algorithm in time $n^{O(1/\varepsilon)}$ for the maximum independent set problem on unit disk graphs is known [17]. The authors of [16] present a constant-factor approximation algorithm for several problems on unit disk graphs, including maximum independent set. For the geometric set cover problem where the goal is to cover a given set of input points with a minimum number of given disks, a PTAS is possible [18].

2 Our results

The Euclidean distance between two points x and y is denoted by $\|x-y\|$. When there is no confusion, we will refer to a point x in the plane and the unit disk centered at x interchangeably.

▶ **Definition 1.** *Given a finite set S in the plane, the* unit disk intersection graph $\mathrm{UDG}(S)$ *is a graph on S where $\{x,y\} \subseteq S$ is an edge of* $\mathrm{UDG}(S)$ *if and only if $\|x-y\| \leq 2$.*

A set S of points in the plane are *connected* if $\mathrm{UDG}(S)$ is a connected graph.

▶ **Definition 2.** *The Maximum Area Connected Subset (*MACS*) problem is as follows.*
***Input**: a finite set of points $X \subseteq \mathbb{R}^2$ and a non-negative integer k, where $k \leq |X|$.*
***Output**: a subset $S \subseteq X$ of size at most k such that the unit-disk graph* $\mathrm{UDG}(S)$ *of S is connected.*
***Goal**: maximize the area of the union of the unit disks centered at the points of S.*
The optimal solution of MACS *on input (X,k) is denoted by $\boldsymbol{OPT}(X,k)$.*

When the context is clear, we refer to $\mathbf{OPT}(X,k)$ as $\mathbf{OPT}$, which is also used to denote the area covered by the optimal solution; observe that $\mathbf{OPT}$ is trivially upper-bounded by πk. Any $S \subseteq X$ with $|S| \leqslant k$ for which $\mathrm{UDG}(S)$ is connected is called a *feasible solution*.

We state our main results below. All omitted proofs and figures can be found in the appendix or in the full version.

▶ **Theorem 3** (Approximation). *There exists a polynomial-time algorithm that computes a* $\frac{1}{2}$*-approximation for* MACS *(Algorithm 1).*

With resource augmentation, we obtain a $(1-\varepsilon)$-approximation.

▶ **Theorem 4** (Resource augmentation). *Let* $\varepsilon > 0$ *be a given parameter. Given a set of points* $X \subseteq \mathbb{R}^2$ *and a non-negative integer* k*, there is an algorithm (Algorithm 2) that computes, in time* $n^{\mathcal{O}(\varepsilon^{-3})}$*, a subset* $S \subseteq X$ *of size at most* k *and a set* $S_{add} \subseteq \mathbb{R}^2$ *of at most* εk *points, such that* $\mathrm{UDG}(S \cup S_{add})$ *is connected and the area of the union of the unit disks centered at* S *is at least* $(1-\varepsilon)\boldsymbol{OPT}(X,k)$ *.*

Theorem 5 can be obtained alternatively by a (deterministic) guillotine cut approach with a faster running time. We leave that for the full version of the paper.

Let $d_G(x,y)$ denote the distance between two vertices x and y of G. A set X of points in the plane is called *α-well-distributed* if $\mathrm{UDG}(X)$ is an α-spanner for X [19]:

▶ **Definition 5.** *Given* $\alpha > 0$*, a finite set* X *of points in the plane is called* α-well-distributed *if for all* $x, y \in X$*,* $d_{UDG(X)}(x,y) \leqslant \lceil \alpha \cdot \|x-y\| \rceil$*.*

▶ **Corollary 6.** *There exists a* PTAS *for* MACS *on* α*-well-distributed inputs, where* α *is a fixed constant (Algorithm 3).*

▶ **Definition 7.** *A set* X *is called* pseudo-convex *if the convex-hull of* X *is covered by the union of the unit disks centered at points of* X*.*

▶ **Lemma 8.** *A pseudo-convex set* X *is* 3.82*-well-distributed.*

▶ **Corollary 9.** MACS *on pseudo-convex inputs admits a polynomial-time approximation scheme.*

We next turn to the hardness of MACS.

▶ **Theorem 10** (Hardness). MACS *is NP-hard.*

▶ **Definition 11.** *The* QUAD-CONNECTED-COVER *is defined as follows.*
Input*: a set* $\mathcal{T}$ *of* n *convex quadrilaterals in the plane, and an integer* k*.*
Output*: a subset* T *of* $\mathcal{T}$ *of size* k *such that the intersection graph of* T *is connected.*
Goal*: Maximise the area covered by the union of quadrilaterals in* T*.*

▶ **Theorem 12.** QUAD-CONNECTED-COVER *is APX-hard.*

3 Proof of Theorem 3: the Two-by-two algorithm

In the section we present a simple $\frac{1}{2}$-approximation for MACS based on a greedy approach: we iteratively add two unit disks that maximize the area covered while maintaining feasibility. Interestingly, the algorithm that adds disks one at a time is not a constant approximation algorithm. See Figure 1 for an example. Moreover, trying all possible sets of s disks, for any $s \geq 3$, in the neighborhood of the current solution does not improve the approximation ratio. This can be seen on Figure 2 where the first disk chosen by the algorithm is not x, but x_s.

Let B_x denote the unit disk centered at $x \in \mathbb{R}^2$ and $B(S) = \bigcup_{x \in S} B_x$ denote their union. The area covered by a set $C \subset \mathbb{R}^2$ is denoted by $\mathcal{A}(C)$. When $C = B(S)$, its area is simply written as $\mathcal{A}(S)$. Given a graph G, $G[S]$ denotes the subgraph induced by a subset S of vertices. A subset of the vertices of a graph is a *dominating set* if every vertex belongs to the set or is adjacent to some vertex of it.

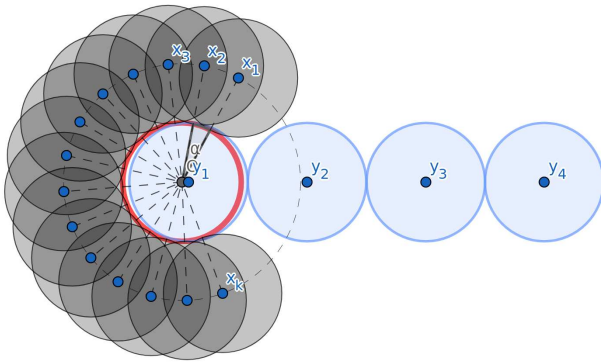


Figure 1 The greedy algorithm that adds only one connected disk maximising the marginal area covered is not a constant factor algorithm. For any $k \geqslant 0$ and $\varepsilon > 0$, consider the above input where $O = (0, 0)$, and $y_i = (2(i-1) + \varepsilon, 0)$ for all i. Then, put all $x_1, \ldots, x_k$ evenly spaced (by an angle α) on a circle of radius 2 around O so that none of them intersect y_2. Each light grey regions are covered by only one disk x_i so the marginal gain of adding x_i to any solution is at least the area of one of these regions, say $a > 0$. If ε is chosen such that $\mathcal{A}(B_{y_1} \setminus B_O) < a$, then if the algorithm starts by picking disk O, it will then choose all x_j, so that the area covered by the solution is upper-bounded by the area of a radius 3 disk, 9π, while the optimal solution (disks y_i) has area πk.

One can find an example similar to Figure 2 to show that optimising the initial choice of the first disk(s) does not improve the approximation ratio.

▶ **Theorem 3** (Approximation). *There exists a polynomial-time algorithm that computes a* $\frac{1}{2}$*-approximation for* MACS *(Algorithm 1).*

We can assume w.l.o.g. that UDG(X) is connected; otherwise we return the maximum value over all connected components. The execution of Algorithm 1 is divided in two phases. An iteration belongs to the first phase as long as the current solution S is not a dominating set in the graph UDG(X).

During the first phase, in each iteration the area covered increases by at least π. During the second phase, since the current solution is a dominating set, any disk can be added while keeping the solution feasible. Therefore the algorithm reduces to a standard greedy algorithm to maximize a submodular function, and the analysis is similar to the proof that Nemhauser's algorithm is a $\left(1 - \frac{1}{e}\right)$-approximation for classic submodular functions.

Algorithm 1 The Two-by-two algorithm for MACS.

Input: $X \subseteq \mathbb{R}^2, k \geq 0$, where X is finite and $k \leq |X|$.
Output: a feasible set of size k.

if *k is even* **then**
  $S \leftarrow$ any two intersecting disks of X;
else
  $S \leftarrow$ any one disk of X;
while $|S| \leqslant k - 2$ **do**
  $\{x, x'\} \leftarrow \arg\max \{\mathcal{A}(S \cup \{x, x'\}) \colon x, x' \in X,\ S \cup \{x, x'\} \text{ is feasible }\}$;
  $S \leftarrow S \cup \{x, x'\}$;
return S;

Proof. We first analyze the even case where $k = 2\kappa$, and then we reduce the odd case to the even one. Let $S_\kappa = \{x_1, x_2, \ldots, x_{2\kappa}\}$ be the solution returned by the algorithm. Let $S_i = \{x_1, \ldots, x_{2i}\}$ be the set right before the i-th iteration and let d be the smallest integer such that S_d is a dominating set in UDG(X). If such an integer does not exist, i.e., S_κ is not a dominating set, then set $d = \kappa$.

▷ Claim 13. The area $\mathcal{A}(S_d)$ is at least πd.

Proof. For $i < d$, S_i is not a dominating set. Then there exist two disks y, y' such that $B(S_i) \cap B_y = \emptyset$ and $S \cup \{y, y'\}$ is connected. Adding such a pair increases the area covered by at least $\mathcal{A}(B_y) = \pi$. Since (x_{2i+1}, x_{2i+2}) is chosen to maximize $\mathcal{A}(S_i \cup \{x, x'\})$ among all feasible pairs, $\mathcal{A}(S_{i+1}) \geqslant \mathcal{A}(S_i \cup \{y, y'\}) \geqslant \mathcal{A}(S_i) + \pi$. By induction, $\mathcal{A}(S_d) \geqslant \pi d$. ◁

Note that when $d = \kappa$, Claim 13 immediately implies that $\mathcal{A}(S_\kappa) \geqslant \frac{\mathbf{OPT}}{2}$. Also regardless of the initial choice, the area covered by the first two disks is at least π. This observation will be useful when we prove the case where k is odd.

▷ Claim 14. For all $d \leqslant i \leqslant \kappa$, $\mathcal{A}(\mathbf{OPT}) \leqslant \mathcal{A}(S_i) + \kappa \cdot (\mathcal{A}(S_{i+1}) - \mathcal{A}(S_i))$.

Proof. It is easy to check that the function $\mathcal{A}(\cdot)$ satisfies the following properties for all $H \subseteq H' \subseteq X$:

positivity: $\mathcal{A}(H) \geqslant 0$.
monotonicity: $\mathcal{A}(H) \leqslant \mathcal{A}(H')$.
submodularity: $\forall H'' \subseteq X,\ \mathcal{A}(H' \cup H'') \leqslant \mathcal{A}(H \cup H'') - \mathcal{A}(H) + \mathcal{A}(H')$.

Let $\mathbf{OPT} = \{y_1, \ldots, y_{2\kappa}\}$. We have for all $d \leqslant i \leqslant \kappa$:

$$\begin{aligned}
\mathcal{A}(\mathbf{OPT}) &\leqslant \mathcal{A}(S_i \cup \mathbf{OPT}) \\
&= \mathcal{A}(S_i) + (\mathcal{A}(S_i \cup \{y_1, y_2\}) - \mathcal{A}(S_i)) + \ldots \\
&\qquad + (\mathcal{A}(S_i \cup \{y_1, \ldots, y_{2\kappa}\}) - \mathcal{A}(S_i \cup \{y_1, \ldots, y_{2\kappa-2}\})) \\
&\leqslant \mathcal{A}(S_i) + (\mathcal{A}(S_i \cup \{y_1, y_2\}) - \mathcal{A}(S_i)) + \cdots + (\mathcal{A}(S_i \cup \{y_{2\kappa-1}, y_{2\kappa}\}) - \mathcal{A}(S_i)) \\
&\leqslant \mathcal{A}(S_i) + \kappa \cdot (\mathcal{A}(S_i \cup \{x_{2i+1}, x_{2i+2}\}) - \mathcal{A}(S_i)) \\
&= \mathcal{A}(S_i) + \kappa \cdot (\mathcal{A}(S_{i+1}) - \mathcal{A}(S_i)) .
\end{aligned}$$

The first and the second inequality respectively come from *monotonicity* and *submodularity*, while the third one follows from the fact that for $i \geq d$, (x_{2i+1}, x_{2i+2}) is the pair of disks maximizing $\mathcal{A}(S_i \cup \{x, x'\})$ among **all pairs** (x, x') in X. As S_d is a connected dominating set in X, all pairs (y_{2j-1}, y_{2j}) for $1 \leqslant i \leqslant \kappa$ are considered. ◁

We can now re-write Claim 14 as

$$\text{For all } d \leqslant i \leqslant \kappa : \mathcal{A}(S_{i+1}) \geqslant \left(1 - \frac{1}{\kappa}\right) \mathcal{A}(S_i) + \frac{\mathbf{OPT}}{\kappa}.$$

Combined with Claim 13, simple algebra yields that for $d \leqslant i \leqslant \kappa$, we have

$$\mathcal{A}(S_i) \geqslant \left[1 - \left(1 - \frac{d}{2\kappa}\right)\left(1 - \frac{1}{\kappa}\right)^{i-d}\right] \mathbf{OPT}.$$

Therefore, for $i = \kappa$ we have

$$\mathcal{A}(S) = \mathcal{A}(S_\kappa) \geqslant \left[1 - \left(1 - \frac{d}{2\kappa}\right)\left(1 - \frac{1}{\kappa}\right)^{\kappa-d}\right] \mathbf{OPT} = \left[1 - \frac{1}{2}(1+t)\left(1 - \frac{1}{\kappa}\right)^{\kappa t}\right] \mathbf{OPT}$$

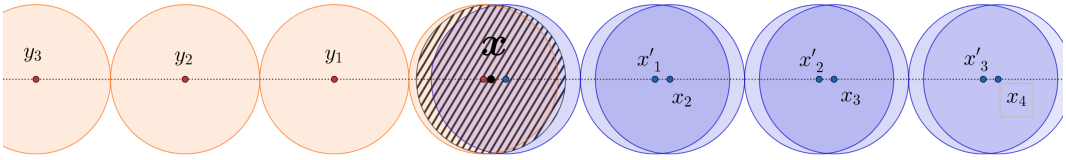


Figure 2 A tight example for Algorithm 1. For any $\varepsilon > 0$, X contains $x = (0, 0)$ (stripe-shaded), $x_i = (2(i-1) + i\varepsilon, 0)$ and $x'_i = ((2+\varepsilon)i, 0)$ for $1 \leqslant i \leqslant k$ (blue) and $y_i = (-2i - \varepsilon/2, 0)$ for $0 \leqslant i \leqslant k$ (orange). Suppose that $k = 1 + 2\kappa$ is odd and the algorithm starts with $S_0 := \{x, x\}$. Then the algorithm will add $\{x_i, x'_i\}$ in iteration i since it covers more additional area than $\{y_0, y_1\}$. The solution returned (blue disks) covers an area of $\pi + \kappa(\pi + f(\varepsilon)) \approx \frac{k}{2}\pi$, for some function $f(\cdot)$ with $\lim_{\varepsilon \to 0} f(\varepsilon) = 0$, while **OPT** (orange disks) covers an area $k\pi$.

where $t = \dfrac{\kappa - d}{\kappa} \in [0, 1]$. As $1 + x \leqslant e^x$ for all $x \in \mathbb{R}$, we get

$$\mathcal{A}(S) \geqslant \left(1 - \frac{1}{2}(1+t)e^{-t}\right)\mathbf{OPT} \geqslant \left(1 - \frac{1}{2}e^t e^{-t}\right)\mathbf{OPT} = \frac{1}{2}\mathbf{OPT},$$

concluding the proof of the case when k is an even number.

For the odd case $k = 2\kappa - 1$: in the first iteration, instead of adding two disks to S_1, we add a single disk of X to S_1. This is equivalent to adding two copies of the same disk. This iteration belongs to the first phase, and the only properties we used in the first phase is that each iteration adds an area of π, and keeps the solution feasible; these are clearly true for the first iteration even with one disk. ◀

Figure 2 shows a tight example.

4 Proof of Theorem 4: PTAS with resource augmentation

▶ **Theorem 4** (Resource augmentation). *Let $\varepsilon > 0$ be a given parameter. Given a set of points $X \subseteq \mathbb{R}^2$ and a non-negative integer k, there is an algorithm (Algorithm 2) that computes, in time $n^{\mathcal{O}(\varepsilon^{-3})}$, a subset $S \subseteq X$ of size at most k and a set $S_{add} \subseteq \mathbb{R}^2$ of at most εk points, such that* $\mathrm{UDG}(S \cup S_{add})$ *is connected and the area of the union of the unit disks centered at S is at least $(1-\varepsilon)\boldsymbol{OPT}(X, k)$.*

We first summarise the high level ideas; the details are then presented in subsequent sections. Let (X, k) denote an input of MACS and **OPT** be the optimal solution of MACS on input (X, k). When the context is clear **OPT** can also denote the total area covered by the union of the unit disks centered in points of **OPT**.

We start by guessing a bounding box of size $\Theta(k) \times \Theta(k)$ that contains **OPT**. Next, another square of size $L \times L$, where $L = \Theta(k)$, is randomly shifted so that it always contains the bounding box. We remove all disks that are outside the square. That square is then recursively partitioned into smaller squares until they have (large) constant size. This hierarchical dissection induces a grid.

We remove all disks that intersect the lines of the grid. In contrast, we deploy some new disks (X_{add}) in some strategic *portal* positions along the lines and near the boundary of all the smallest squares.

Next, we use dynamic programming to build a solution from the smallest squares upwards. The difficulty lies in having to guarantee the connectivity when combining solutions from smaller squares into larger squares using additional disks, while controlling the time complexity and the number of disks added.

The key of our approach lies in Lemma 20, in which we argue that with constant probability, there exists a well-structured near-optimal solution that uses at most εk additional disks.

4.1 The grid

The first step is to reduce significantly the size of the input by guessing the position of the optimal solution.

▶ **Lemma 15.** *There exists a point $c \in X$ such that **OPT** is contained in an axis-parallel square of side length $4k$ and centered in c.*

Proof. For c, take any disk in **OPT** and recall that **OPT** is connected and has at most k disks, so all the disks in **OPT** are contained in the square centered at c and with side length $4k$. ◀

Given the randomly shifted hierarchical dissection, we use the same terminology as Vazirani [22, Chapter 11] to define the *root square*, the *shift* of the dissection, the *horizontal* and *vertical lines*, the *levels* of squares and of lines of the dissection, and the *portals*. The recursive dissection stops when a square has side length $L_0 = \Theta(\varepsilon^{-1})$ (*leaf square*). Portals are either at the intersection of grid lines or distributed along the grid lines (with varying density). We make some observations here (all details and proofs are in the following section and the appendix). First, the distance between two consecutive portals on a line at level ℓ is $\mathcal{O}(L/(m2^\ell))$, where m represents the density of portals on the grid. The greater this parameter, the greater the accuracy of the solution and higher the running time. Choosing $m = \Theta(\varepsilon^{-1}\log(L/L_0)) = \mathcal{O}(\varepsilon^{-1}\log(\varepsilon k))$ allows us to compute a near-optimal solution in polynomial time.

▶ **Observation 16.** *If an horizontal line of level ℓ crosses a vertical line of level greater than or equal to ℓ then the intersection point is a horizontal portal.*

We define a set $\mathcal{P}$ of *portal disks* which we position at or near the portals. If a portal (i, j) is on exactly one line of the grid then we add the portal disk (i, j) to $\mathcal{P}$. If a portal (i, j) is at the intersection of two lines of the grid, then i) if it is a horizontal portal then we add to $\mathcal{P}$ two portal disks $(i, j + 2)$ and $(i, j - 2)$, and ii) if it is a vertical portal then we add to $\mathcal{P}$ two portal disks $(i - 2, j)$ and $(i + 2, j)$.

Given a square C of the dissection, the *potential portal disks* of C, denoted by $\mathcal{P}_C$, are the portal disks on the boundary of C.

▶ **Observation 17.** *For any square, the number of potential portal disks is $\mathcal{O}(m) = \mathcal{O}(\varepsilon^{-1}\log(\varepsilon k))$.*

The *border* of a leaf square C, denoted as ∂C, is the set of points in C within distance 1 from C's boundary. The remaining points of C are called the *core* of C, written as $core(C)$. A unit disk with its center in C intersects the boundary if and only if its center lies in the border. If two disks are in the core of two different leaf squares, then they do not intersect. We refer to the union of the core of all *leaf squares* as the *core*. In a leaf square $C = [a, b] \times [c, d]$, the set of points formed by the boundary of the square $[a + 2, b - 2] \times [c + 2, d - 2]$ is called the *fence*. We cover the fence of C by *fence disks*, aligned such that each corner of this square is the center of a fence disk. See Figure 4. We denote by $\mathcal{F}$ the set of all fence disks for all leaf squares. The set of portal disks and fence disks form the set of *additional disks* $X_{add} = \mathcal{P} \cup \mathcal{F}$.

4.2 Detailed construction of the grid

Let L' be the sidelength of the box given by the Lemma 15, and set X' be the set of points of X lying inside this box. Let L be the smallest power of 2 greater than $2L'$. The *root square* is defined to be the axis-parallel $L \times L$ square with the same left-bottom corner as the bounding-box.

A *shift* is an non-negative integer a smaller than or equal to $L/2$. We say that the root square is *shifted* by a if it is translated by the vector $(-a, -a)$. Notice that any shifted root square contains the bounding-box.

Given a shifted root square, we can define its *dissection* as a recursive partitioning into smaller squares. The $L \times L$ root square is divided into four squares of size $L/2 \times L/2$. Each of these squares is again divided into four $L/4 \times L/4$ squares, so forth. The process stops when the side length of a square is equal to $L_0 = \Theta(\varepsilon^{-1})$. Let $d = \log(L/L_0) = \mathcal{O}(\log(\varepsilon k))$. We can think of this partitioning as 4-ary tree, where each node at level ℓ corresponds to a $L_0 2^\ell \times L_0 2^\ell$ square and has four children corresponding to four $L_0 2^{\ell-1} \times L_0 2^{\ell-1}$ squares. The root square is at level 0 and the *leaf squares* are at level d. Given two squares of level ℓ and level ℓ', $\ell > \ell'$, we say the former is of higher level than the latter. So the leaf square is the one with the highest level.

This dissection defines a *grid* composed of $2 \cdot (2^d - 1)$ horizontal and vertical lines of length L. We say that a line is at *level* $\ell \in \{1, \ldots, d\}$ if it was added on the grid to divide a square at level $l - 1$ into four squares at level ℓ. There are 2^ℓ horizontal (*resp.* vertical) lines at level ℓ. See figure 3.

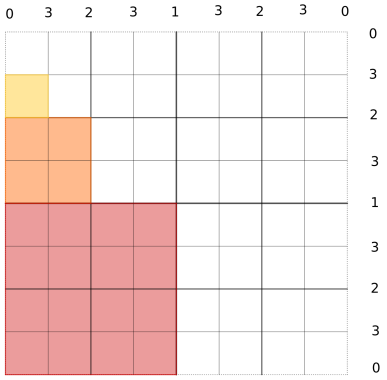


Figure 3 An illustration of the grid with $d = 3$. Numbers on the top and the right are the level of the corresponding lines and the red, orange and yellow are respectively the example of square of the dissection at level 1, 2 and 3.

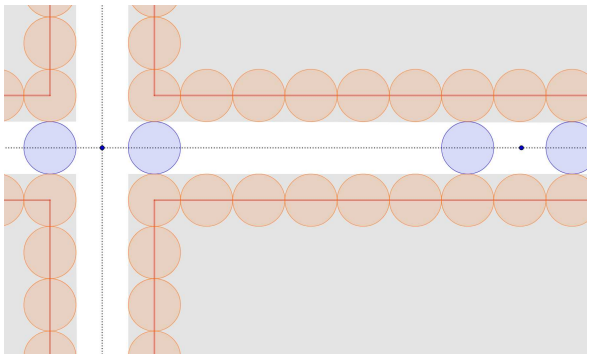


Figure 4 The grey and white area are respectively the *core* and the *border*. Dotted lines are from the grid while the orange lines represent the *fence* and orange disks are the *fence disks*. Blue points are (vertical) *portals* and blue disks are *portal disks*.

On each horizontal line of level $\ell \geqslant 1$, we will place a set of *vertical* – notice the naming asymmetry – *portals* of level ℓ, near which (not exactly on which) we will deploy the portal disks to facilitate the connection of disks on both sides of this line. We define a set of horizontal portals for each vertical line in an analogous manner. Notice that it is possible that a point is both a vertical portal and a horizontal portal. Let $m = \mathcal{O}(\varepsilon^{-1} d)$ be a power of two. Along a line of level ℓ, there are $m2^\ell + 1$ portals evenly spaced so that the distance between two neighboring portals have distance exactly $\frac{L}{m2^\ell}$.

4.3 Dynamic program

The algorithm uses dynamic programming. The dynamic programming table is indexed by *configurations*.

▶ **Definition 18.** *A* configuration *is a 5-tuple* $\mathcal{C} = [C, t, t_{add}, P, \sim]$*, where:*

- C *is a square of the dissection.*
- $0 \leq t \leq k$ *is an integer, denoting the number of disks of* S *used by the solution inside* C*.*
- $0 \leq t_{add} \leq \varepsilon k$ *is an integer, denoting the number of additional disks used by the solution inside* C*.*
- $P \subseteq \mathcal{P}_C$ *is a subset of potential portal disks of* C*, those that are used by the solution.*
- $\sim$ *is a planar connectivity relation on* P *(described below), representing the connectivity achieved so far by the part of the solution inside* C*.*

In the following, to facilitate discussion, we will refer to portals disks as simply portals. An equivalence relation $\sim$ on P is a *planar connectivity relation* if each equivalence class has an associated tree with the portals at the leaves, and there exists a planar embedding of those trees inside the square, such that the trees do not intersect.

The content of the dynamic programming table, the *value* of a configuration $\mathcal{C} = [C, t, t_{add}, P, \sim]$, denoted by $\mathcal{A}(\mathcal{C})$, is the maximum area that can be covered by a set $S \subseteq X$ of t disks in $C \cap core$[4], such that there is a set $S_{add} \subseteq X_{add}$ of t_{add} additional disks such that any $p, p' \in P$ with $p \sim p'$ are in the same connected component induced by $S \cup S_{add} \cup P$. We say that p and p' are *connected in* $\mathcal{C}$. If such sets $\{S, S_{add}, P\}$ do not exist for configuration $\mathcal{C}$, the value $\mathcal{A}(\mathcal{C})$ is set to $-\infty$.

4.4 Computing leaf entries of the dynamic programming

We first explain how to fill the entries of the table corresponding to the leaf squares. For each leaf square C, we enumerate

1. all possible subsets $S \subseteq X' \cap core(C)$ of at most k_0 disks, for a parameter $k_0 = \mathcal{O}(\varepsilon^{-3})$ (see Lemma 20).
2. all possible subsets $S_f \subseteq \mathcal{F} \cap C$,
3. all possible subsets $P \subseteq \mathcal{P}_C$, and
4. all possible planar connectivity relations $\sim$ on P.

We say that $(S, S_f, P, \sim)$ is a *guess* in C and that it is *usable* if one of the following two conditions holds:

Case 1. if $P = \emptyset$, then $S \cup S_f$ is connected, otherwise

Case 2. every connected component of $S \cup S_f \cup P$ contains at least one portal disk in P.

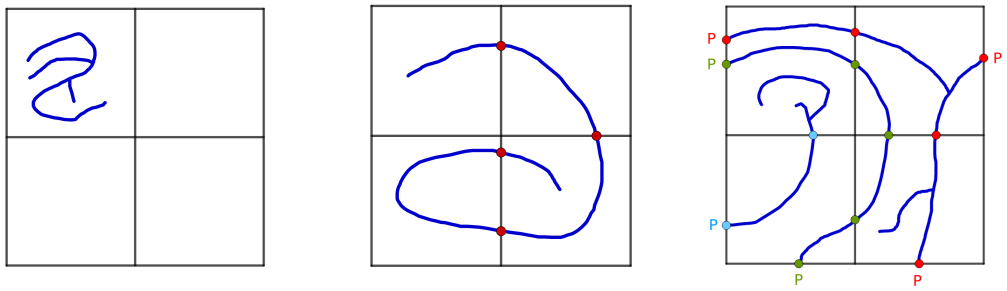
Each usable guess (S, S_f, P) in C corresponds to a configuration $\mathcal{C} := [C, |S|, |S_f|, P, \sim]$, where $\sim$ is the planar connectivity relation on P induced by the connected components of $S \cup S_f \cup P$.

Several usable guesses (S, S_f, P) can potentially correspond to the same configuration $\mathcal{C}$. The value of $\mathcal{C}$ is computed[5] as the maximum value $\mathcal{A}(S)$ over all such guesses S.

---

[4] Recall that *core* is the union of the $core(C)$ of all leaf squares C.

[5] The area covered by the union of a set of disks is a real number that can be computed exactly. When the desired accuracy is a fixed constant (for instance ε), one can give an approximation of this area with the desired precision in polynomial time.



(a) the top-left configuration is closed while other configurations are empty.

(b) there is unique connected component independent from the outside world.

(c) Each connected component contains a portal in P.

Figure 5 Illustration of cases (a)-(b)-(c) of point 6. in Definition 19.

4.5 Computing all entries

It remains to show how to compute the solution of a configuration, say $\mathcal{C} = [C, t, t_{add}, P, \sim]$, for a square C at level ℓ, by combining the solutions $\left[C^i, t^i, t^i_{add}, P^i, \sim^i\right]$ of the four child squares C^i, $i = 1, 2, 3, 4$, at level $\ell + 1$. Recall that connectivity relations $\sim^i$ capture the information about connectivity in the squares C^i. Let $P = \{p_0, \ldots, p_s\}$ be the subset of potential portal disks. We define $\sim'$ as the *transitive closure* of all $\sim^i$: $p \sim' p'$ if and only if there exists a sequence of squares $i_1, \ldots, i_s \in \{1, 2, 3, 4\}$ and a sequence of portals $p = p_0, \ldots, p_s = p'$ such that for all $1 \leqslant j \leqslant s$, p_j is a common portal of $P^{i_{j-1}}$ and P^{i_j}. Further, p_{j-1} and p_j must be connected in $\mathcal{C}^{i_j}$. We call $\mathcal{C}$ *empty* if $P = \emptyset$ and $t = 0$, and *closed* if $P = \emptyset$ and $t > 0$.

We now define the notion of compatibility of configurations.

▶ **Definition 19.** *Five configurations* $\left(\mathcal{C}, \mathcal{C}^1, \mathcal{C}^2, \mathcal{C}^3, \mathcal{C}^4\right)$ *with* $\mathcal{C} = [C, t, t_{add}, P, \sim]$ *and* $\mathcal{C}^i = [C^i, t^i, t^i_{add}, P^i, \sim^i]$ *are* compatible *if all the following properties are satisfied.*

1. *all* C^i *have the same level and their union is the square* C*.*
2. $P = \bigcup_{i=1}^4 P^i \cap \partial C$*.*
3. $\sim$ *is the restriction of the transitive closure* $\sim'$ *of* $\left(\sim^i\right)_{1 \leqslant i \leqslant 4}$ *to* P*.*
4. $t = t^1 + t^2 + t^3 + t^4$ *and* $t \leqslant k$*.*
5. $t_{add} = t^1_{add} + t^2_{add} + t^3_{add} + t^4_{add} + |\bigcup_{i=1}^4 P^i \setminus P|$ *and* $t_{add} \leqslant \varepsilon k$*.*
6. *exactly one of following three conditions holds.*
   - **(a)** $\mathcal{C}^i$*,* $i \in \{1, 2, 3, 4\}$*, is closed and all* $\mathcal{C}^j$*,* $j \neq i$ *are empty.*
   - **(b)** $\mathcal{C}$ *is closed and there is exactly one equivalence class for* $\sim'$*.*
   - **(c)** *all equivalence classes of* $\sim'$ *contain a portal in* P*.*

▶ Remark. By condition 2, the set P of portals used by C is obtained by removing from $\bigcup_{i=1}^4 P^i$ the portals not on the border of C. Notice that these removed portals in $\bigcup_{i=1}^4 P^i \setminus P$ are now counted as additional disks (in condition 5). Condition 6 attempts to capture all possible situations – either we have a single connected component not connected to the "outside world", which is a feasible solution by itself, (see Condition (6a) and Condition (6b)), or we have several connected components, each of which must be further connected to the outside world in a later stage (see Condition (6c)). See Figure 5. Finally, it is easy to see that if all $\sim^i$ satisfy the connectivity relation, then so does $\sim$.

Let a be a shift chosen uniformly at random in $\left\{0, \frac{L}{2}\right\}$. We consider the grid associated to this shift and the set of additional disks on this grid as defined in the previous section. The following lemma is essential to our main theorem. Recall that $\mathcal{P}$ denotes the set of portal disks and $\mathcal{F}$ the set of fence disks.

▶ **Lemma 20** (Structural Lemma). *Given a fixed parameter $\varepsilon > 0$, there exists a subset $S \subseteq$ core of input disks and a set $S_{add} \subseteq \mathcal{P} \cup \mathcal{F}$ of* additional disks*, such that with probability at least $1/3$,*

- **(i)** *(feasibility) $|S| \leqslant k$ and $S \cup S_{add}$ is connected,*
- **(ii)** *(bounded resource augmentation) $|S_{add}| \leqslant \varepsilon k$,*
- **(iii)** *(near-optimality) $\mathcal{A}(S) \geqslant (1-\varepsilon)\boldsymbol{OPT}$,*
- **(iv)** *(bounded local size) For each leaf square C, $|C \cap S| = \mathcal{O}(\varepsilon^{-3})$.*

Our dynamic programming aims at finding a solution satisfying all conditions of this Structural Lemma. We show that such a solution can be computed in time $n^{\mathcal{O}(\varepsilon^{-3})}$. The *bounded local size* property ensures that we can try all possible configurations in the leaf squares in polynomial time. We also prove that for any square, the number of different planar connectivity relations is upper-bounded by the *Catalan number* of the number of potential portal disks of the square. It follows from Observation 17 that this number is polynomially bounded.

4.6 Proof of the structural Lemma

We construct S and S_{add} from **OPT** in two steps. In the first step, we build sets S' and S_{add} that satisfy properties *(i)-(iii)*; and in the second step, we construct $S \subseteq S'$ by removing some disks from S' so as to satisfy property *(iv)* while maintaining the validity of the three first properties.

4.6.1 Part 1: Construction of the set of additional disks

Fix any shift, consider its associated grid and dissection and the corresponding set of additional disks $X_{add} = \mathcal{P} \cup \mathcal{F}$. Let S' be the union of disks in **OPT** that are located in the core of a leaf square of the dissection, namely

$$S' = \mathbf{OPT} \cap \mathit{core}.$$

Observe that S' might be disconnected since we have removed from **OPT** all the disks that were intersecting the grid. Letting *border* denote $\bigcup_{C \text{ is leaf}} \partial(C)$, we show how to replace the set of input disks $\mathbf{OPT} \cap \mathit{border}$ by a subset $S_{add} \subseteq \mathcal{F} \cup \mathcal{P}$ of additional disks.

Each leaf square $[a,b] \times [c,d]$ has an associated fence that is the boundary of the square $[a+2, b-2] \times [c+2, d-2]$. For each vertical (*resp.* horizontal) portal disk (x,y), we define a *connection line*, which is $\{x\} \times [y-2, y+2]$ (*resp.* $[x-2, x+2] \times \{y\}$). The set of fences and connection lines naturally partition the set of points which are at distance at most 2 from the lines of the grid into a set of *rectangles* $\mathcal{R}$. See Figure 6. Notice that all connections and fences are covered by the union of additional disks. Given a rectangle $R \in \mathcal{R}$, we define $\text{disk}(R) \subseteq X_{add}$ as the minimal set of additional disks that contain R.

We construct S_{add} as the union of $\text{disk}(R)$, over all rectangles R that intersect a disk $x \in \mathbf{OPT} \cap \mathit{border}$.

$$S_{add} = \bigcup \{\text{disk}(R) \colon R \in \mathcal{R}, \exists x \in \mathbf{OPT} \cap \mathit{border} \text{ such that } B_x \cap R \neq \emptyset\}$$

Notice that each disk $x \in \mathbf{OPT} \cap \mathit{border}$ intersects at most two rectangles. Furthermore, such a disk does not intersect with any fence and can intersect at most one connection line.

▷ Claim 21. Sets S' and S_{add} are such that $S' \cup S_{add}$ is connected, S' has size at most k and with probability at least $1/3$: $|S_{add}| \leqslant \mathcal{O}(\varepsilon k)$ and $\mathcal{A}(S') \geqslant (1 - \mathcal{O}(\varepsilon))\mathbf{OPT}$.

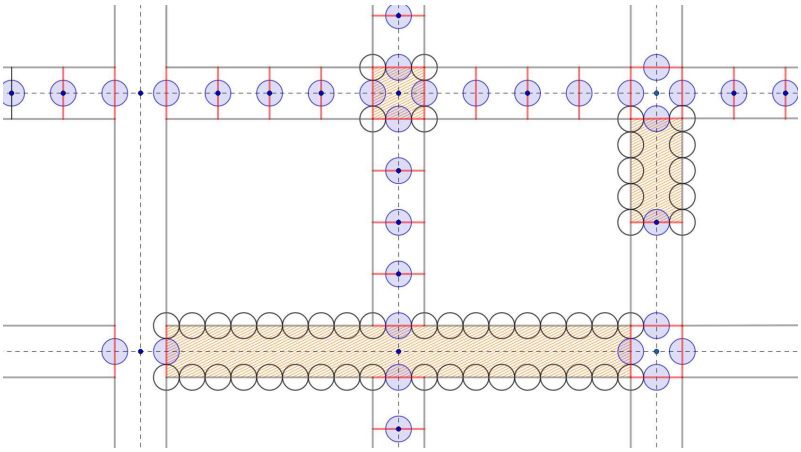


Figure 6 Dotted lines are the grid lines. The bottom and top horizontal lines have respectively level 8 and 10, and the vertical lines from left to right have level 5, 10 and 9. Grey continuous line are the *fence*, and the red ones, the *connection lines*. Points and blue disks are *portals* and *portal disks*. Striped orange areas illustrate some rectangles $R \in \mathcal{R}$, and other disks are *fence disks* of the corresponding sets disk(R).

The proof is in the appendix (the argument is similar to the one of Arora [1]). We first upper-bound the expectation of $|S_{add}|$ and $\mathcal{A}(\mathbf{OPT}) - \mathcal{A}(S')$, and then use Markov's inequality. To bound the expectation of $|S_{add}|$, we observe that the number of additional disks added in S_{add} for each disk in **OPT** intersecting a line at level ℓ is $\mathcal{O}(L/(m2^\ell))$ while the probability that a disk intersects a line at level ℓ is $\mathcal{O}(2^\ell/L)$.

4.6.2 Part 2: Sparsification of S'

The sets $S' \cup S_{add}$ obtained so far may not satisfy the last property (*bounded local size*). In this section, we show how to remove some disks from S' to guarantee this property while still maintaining the other required properties in Lemma 20.

Suppose that there exists a leaf square C such that $S'_C := S' \cap C$ has size greater than $k_0 := (1 + \beta^{-1})L_0^2 = \mathcal{O}(\varepsilon^{-3})$, where $\beta = \min\{\varepsilon/12, 1\}$. Then the core of C is "overcrowded" and we show how to construct a non-overcrowded subset maintaining connectivity while losing only an $\varepsilon/2$-th fraction of the covered area.

Define a set S to be initially equal to S'. Consider each overcrowded leaf square C one by one, and define $S_C = S \cap C$. Start with an empty set H and for each disk $x \in S_C$, add x in H if $\mathcal{A}(H \cup \{x\}) - \mathcal{A}(H) \geqslant \beta$. Define $\overline{H} = S_C \setminus H$ as the complement of H and then apply Claim 22 to $G = UDG(S \cup S_{add})$ and $D = S \cup S_{add} \setminus \overline{H}$ to define $D' \subseteq \overline{H}$. Finally update S to $(S \setminus \overline{H}) \cup D'$.

▷ Claim 22. Let $G = (V, E)$ be a connected graph and D a dominating set with μ connected components. There exists a subset $D' \subseteq V \setminus D$ of size at most $2(\mu - 1)$ such that $G[D \cup D']$ is connected.

Proof. Let H and H' be two connected components in D that minimize $d_G(H, H')$. Then, $d_G(H, H') \leqslant 3$. Indeed, if $d_G(H, H') \geqslant 4$, then there exists a vertex x on a shortest path from H to H' that is not dominated by D. This implies that we can find two vertices that connect H and H'. We repeat this operation until there is only one connected component. This requires at most $2(\mu - 1)$ vertices. ◁

The following claim, together with Claim 21 ensures that sets S and S_{add} built in Part 1 and Part 2 satisfy the expected properties of our structural Lemma.

▷ Claim 23. The constructed sets S and S_{add} satisfy

(i) $S \cup S_{add}$ is connected,

(ii) for each leaf square C, $|S \cap C| \leqslant k_0$, and

(iii) $\mathcal{A}(S) \geqslant (1 - \varepsilon/2)\mathcal{A}(S')$.

This Claim might not be true if the radius of disks considered are arbitrary. The proof of this fact follows from geometrical observations about unit disks.

Algorithm 2 PTAS for MACS with resource augmentation.

Input: X, k, ε.
Output: a real number $maxi \geqslant (1-\varepsilon)\mathbf{OPT}$.

forall $c \in X$ **do**
  let $\mathcal{B}'$ be the $4k \times 4k$ square centered at c;
  $X' \leftarrow X \cap \mathcal{B}'$;
  $L \leftarrow$ the smallest power of 2 such that $L \geqslant 8k$;
  forall *shift* $a \in \{0, \ldots, L/2\}$ **do**
    Create a table *tab*;
    foreach *configuration* $\mathcal{C}$ **do**
      $tab[\mathcal{C}] \leftarrow -\infty$;
    `/* Initialization */`
    foreach $\mathcal{C}$ *at level* d *(leaf square)* **do**
      $tab[\mathcal{C}] \leftarrow \max\{\mathcal{A}(S) : (S, S_f, P)$ is usable and corresponds to $\mathcal{C}\}$;
    `/* Fusion */`
    foreach *level* $0 \leqslant i \leqslant d-1$ *in decreasing order* **do**
      foreach *configuration* $\mathcal{C}$ *at level* i **do**
        $tab[\mathcal{C}] \leftarrow \max\left\{\sum_{i=1}^{4} tab[\mathcal{C}^i] : (\mathcal{C}, \mathcal{C}^1, \mathcal{C}^2, \mathcal{C}^3, \mathcal{C}^4)$ are compatible $\right\}$;
return $maxi = \max_{\substack{\text{configuration } \mathcal{C} \\ \text{for root square}}} tab[\mathcal{C}]$;

Notice that since the root square has no potential portals (portals are only placed on lines at level at least 1), any configuration that corresponds to the root square has only one connected component. We can easily add information in the table so that the algorithm also outputs the corresponding sets S and S_{add}.

Notice that Algorithm 2 tries all possible shift a. Our structural Lemma 20 ensures that there exists at least one shift such that the output satisfies all expected properties of Theorem 4.

▶ **Theorem 24.** *Algorithm 2 has a running time* $n^{\mathcal{O}(\varepsilon^{-3})}$.

The key ingredient in order to prove that our algorithm is polynomial follows from Observation 17. We show that the number of connectivity relations of a set of $\mathcal{O}(m)$ portals corresponds to its *Catalan number* which is polynomial when $m = \mathcal{O}(\varepsilon^{-1}\log(\varepsilon k))$.

Algorithm 3 PTAS for MACS for well-distributed inputs.

Input: X an α-well-distributed input, $k \geqslant 0$, $\varepsilon > 0$.
Output: A feasible solution to MACS(X, k).
1 Choose $\varepsilon' > 0$ and $k' \leqslant k$ such that $(1-\varepsilon')(1-10(22\alpha+4)\varepsilon') \geqslant (1-\varepsilon)$ and $k'(1+(22\alpha+4)\varepsilon') = k$;
2 Let S, S_{add} be the solution of Algorithm 2 on input (X, k', ε') ;
3 Let S' be the set obtained from S_{add} by Lemma 25;
4 **return** $S \cup S'$;

5 A PTAS for well-distributed inputs

Let us recall the definition of well-distributed input.

▶ **Definition 5.** *Given $\alpha > 0$, a finite set X of points in the plane is called* α-well-distributed *if for all $x, y \in X$, $d_{UDG(X)}(x, y) \leqslant \lceil \alpha \cdot \|x - y\| \rceil$.*

Here $\lceil \cdot \rceil$ is the ceiling function. This ensures that the right-hand side is always at least one. Notice that a well-distributed set is necessarily connected.

One intuitive view of a well-distributed input is to look at the shape of the "holes" of the input, that are the different connected components of the complement of the union of the input disks in the plane. The assumption of well-distribution means that these holes are roughly *fat*.

One particular interesting case arises when there is no hole at all. We call these sets *pseudo-convex*, and we prove that this is a particular case of well-distributed inputs.

▶ **Definition 7.** *A set X is called* pseudo-convex *if the convex-hull of X is covered by the union of the unit disks centered at points of X.*

▶ **Lemma 8.** *A pseudo-convex set X is 3.82-well-distributed.*

Our Corollary 6 states that the restriction of MACS to well-distributed inputs admits a PTAS. The algorithm works as follows. Given a parameter $0 < \varepsilon \leqslant 1/2$, and an input (X, k) of MACS, we run Algorithm 2 on input (X, k', ε') for suitable values k' and ε' specified below. Next, we transform the set of additional disks obtained into a set of input disks that has roughly the same size while maintaining the connectivity of the solution. See Lemma 25 and Algorithm 3 for details. This algorithm naturally applies to pseudo-convex inputs (Corollary 9).

▶ **Lemma 25.** *Given an α-well-distributed input X and two finite sets $S \subseteq X$ and $S_{add} \subseteq \mathbb{R}^2$ such that $UDG(S \cup S_{add})$ is connected, there exists a set $S' \subseteq X$ of size at most $(22\alpha+4)|S_{add}|$ such that $UDG(S \cup S')$ is connected. Moreover, such a set can be computed in polynomial time.*

In the previous lemma, the set S_{add} is not supposed to be a set of additional disks as defined in Section 4.

Since $\varepsilon' = \Theta(\varepsilon/\alpha)$, the previous algorithm runs in polynomial time when ε and α are fixed constants.

▷ Claim 26. The solution returned by Algorithm 3 on input (X, k, ε) is a feasible solution to MACS(X, k) and covers an area at least $(1-\varepsilon)\mathbf{OPT}(X, k)$.

In order to prove this result we need to state the following "stability" property over optimal solutions.

▶ **Lemma 27.** *Let $\eta < \frac{1}{2}$. Then $\boldsymbol{OPT}(X, k) \geqslant (1 - 10\eta) \cdot \boldsymbol{OPT}(X, k(1+\eta))$.*

References

1. Sanjeev Arora. Polynomial Time Approximation Schemes for Euclidean Traveling Salesman and Other Geometric Problems. *J. ACM*, 45(5):753–782, September 1998. doi:10.1145/290179.290180.
2. Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a Monotone Submodular Function Subject to a Matroid Constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.
3. Erin W. Chambers, Sándor P. Fekete, Hella-Franziska Hoffmann, Dimitri Marinakis, Joseph S.B. Mitchell, Venkatesh Srinivasan, Ulrike Stege, and Sue Whitesides. Connecting a set of circles with minimum sum of radii. *Computational Geometry*, 68(1-3):62–76, January 1991. special issue in memory of Ferran Hurtado.
4. Steven Chaplick, Minati De, Alexander Ravsky, and Joachim Spoerhase. Approximation Schemes for Geometric Coverage Problems. In *ESA*, volume 112 of *LIPIcs*, pages 17:1–17:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.
5. Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular Function Maximization via the Multilinear Relaxation and Contention Resolution Schemes. *SIAM J. Comput.*, 43(6):1831–1879, 2014.
6. Yuval Filmus and Justin Ward. Monotone Submodular Maximization over a Matroid via Non-Oblivious Local Search. *SIAM J. Comput.*, 43(2):514–542, 2014. doi:10.1137/130920277.
7. Stefan Funke, Alex Kesselman, Fabian Kuhn, Zvi Lotker, and Michael Segal. Improved Approximation Algorithms for Connected Sensor Cover. *Wirel. Netw.*, 13(2):153–164, April 2007. doi:10.1007/s11276-006-3724-9.
8. L. A. Wolsey G.L. Nemhauser and M.L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14(1):265–294, 1978.
9. Himanshu Gupta, Zongheng Zhou, Samir R. Das, and Quinyi Gu. Connected Sensor Cover: Self-organization of Sensor Networks for Efficient Query Execution. *IEEE/ACM Trans. Netw.*, 14(1):55–67, February 2006. doi:10.1109/TNET.2005.863478.
10. Dorit S. Hochbaum and Anu Pathria. Node-Optimal Connected k-Subgraphs, 1994.
11. Koushik Kar and Suman Banerjee. Node Placement for Connected Coverage in Sensor Networks, 2003.
12. Samir Khuller, Manish Purohit, and Kanthi K. Sarpatwar. Analyzing the Optimal Neighborhood: Algorithms for Budgeted and Partial Connected Dominating Set Problems. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 1702–1713, Philadelphia, PA, USA, 2014. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2634074.2634197.
13. Ariel Kulik, Hadas Shachnai, and Tami Tamir. Maximizing submodular set functions subject to multiple linear constraints. In *SODA*, pages 545–554. SIAM, 2009.
14. Tung-Wei Kuo, Kate Ching-Ju Lin, and Ming-Jer Tsai. Maximizing Submodular Set Function with Connectivity Constraint: Theory and Application to Networks. *IEEE/ACM Trans. Netw.*, 23(2):533–546, April 2015. doi:10.1109/TNET.2014.2301816.
15. Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular Maximization over Multiple Matroids via Generalized Exchange Properties. *Math. Oper. Res.*, 35(4):795–806, 2010.
16. M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz. Simple heuristics for unit disk graphs. *NETWORKS*, 1995.
17. Tomomi Matsui. Approximation Algorithms for Maximum Independent Set Problems and Fractional Coloring Problems on Unit Disk Graphs. In Jin Akiyama, Mikio Kano, and Masatsugu Urabe, editors, *Discrete and Computational Geometry*, pages 194–200, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.
18. Nabil H. Mustafa and Saurabh Ray. Improved Results on Geometric Hitting Set Problems. *Discrete & Computational Geometry*, 44(4):883–895, 2010.
19. Giri Narasimhan and Michiel Smid. *Geometric Spanner Networks*. Cambridge University Press, New York, NY, USA, 2007.

20 Yuval Rabani and Gabriel Scalosub. Bicriteria approximation tradeoff for the node-cost budget problem. *ACM Trans. Algorithms*, 5(2):19:1–19:14, 2009.

21 Fabio Vandin, Eli Upfal, and Benjamin J. Raphael. Algorithms for Detecting Significantly Mutated Pathways in Cancer. *Journal of Computational Biology*, 18(3):507–522, 2011.

22 Vijay V. Vazirani. *Euclidean TSP*, pages 84–89. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. doi:10.1007/978-3-662-04565-7_11.

23 L. Wolsey. Maximising real-valued submodular functions: primal and dual heuristics for location problems. *Mathematics of Operations Research*, 7(3):410–425, 1982.

A Omitted proofs

Proof of Claim 21. Clearly $|S'| \leqslant |\mathbf{OPT}| \leqslant k$. We now prove that $S' \cup S_{add}$ is connected. Suppose that there exists a disk $x \in \mathbf{OPT} \cap \mathit{border}$ such that $\mathbf{OPT} \setminus \{x\}$ is split into several connected components. We know that x intersects only one rectangle $R_1 \in \mathcal{R}$ or two rectangles $R_1, R_2 \in \mathcal{R}$. Since $\mathbf{OPT}$ is connected, and B_x is contained in the set $U = R_1$ or $U = R_1 \cup R_2$, each connected component intersects the boundary of U. Then, B_x intersects a disk in $\mathrm{disk}(R_1)$ or $\mathrm{disk}(R_2)$. Therefore, $\mathbf{OPT} \setminus \{x\} \cup \mathrm{disk}(R_1) \cup \mathrm{disk}(R_2)$ is connected. By doing so for each $x \in \mathbf{OPT} \cap \mathit{border}$, it follows that $S' \cup S_{add}$ is connected.

It remains to show that, under a uniform random shift a, with probability at least one third we have $|S_{add}| \leqslant \mathcal{O}(\varepsilon k)$ **and** $\mathcal{A}(S') = \mathcal{A}(\mathbf{OPT} \cap \mathit{core}) \geqslant (1 - \mathcal{O}(\varepsilon))\mathbf{OPT}$. The proof is very similar to Arora's approach, we first upper-bound the expectation of $|S_{add}|$ and $\mathcal{A}(\mathbf{OPT}) - \mathcal{A}(S')$, and then use *Markov inequality* to conclude.

We first upper-bound the expected number of additional disks. For each $x \in \mathbf{OPT}$ intersecting a line at level ℓ, we have added at most two sets of additional disks associated to rectangles with side length smaller than the distance between two consecutive portals of this line. It follows that $\mathcal{O}(L/(m2^\ell))$ additional disks have been added to S_{add} for each disk in $\mathbf{OPT}$ intersecting a line of level ℓ. This can be observed in Figure 7. Moreover, the probability that a disk intersects a line at level ℓ is $\mathcal{O}(2^\ell/L)$. Then,

$$\begin{aligned}\mathbb{E}(|S_{add}|) &\leqslant \sum_{x \in \mathbf{OPT}} \sum_{\ell=0}^{d-1} P(x \text{ intersects exactly one line at level } \ell)\mathcal{O}\left(\frac{L}{m2^\ell}\right) \\ &= \sum_{x \in \mathbf{OPT}} \sum_{\ell=0}^{d-1} \mathcal{O}\left(\frac{2^\ell}{L} \cdot \frac{L}{m2^\ell}\right) = \mathcal{O}\left(\frac{dk}{m}\right) = \mathcal{O}(\varepsilon k)\end{aligned}$$

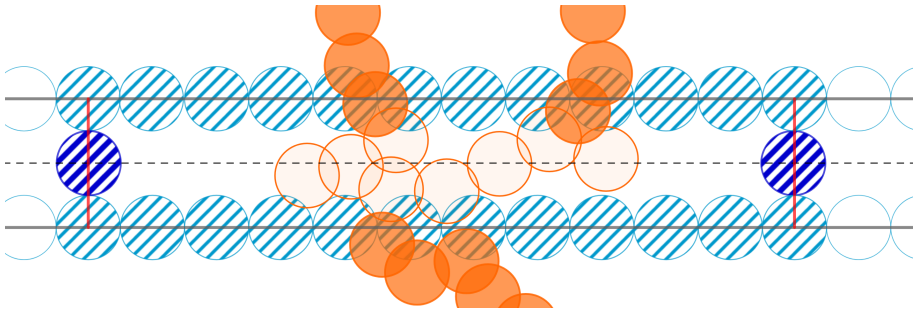


Figure 7 **OPT** is represented by orange disks. Disks of **OPT** that intersect the grid (dotted line) are replaced by additional disks (striped blue disks). This operation maintains the connectivity of the set.

We now upper-bound the expectation of $\mathcal{A}(\mathbf{OPT}) - \mathcal{A}(S')$. First we have $\mathcal{A}(\mathbf{OPT}) - \mathcal{A}(S') \leqslant \mathcal{A}(\mathbf{OPT} \cap border)$, and the probability that a point $p \in B(\mathbf{OPT})$ is in $B(\mathbf{OPT} \cap border)$ is smaller that p is at distance 2 from the lines of the grid. Therefore

$$\begin{aligned}\mathbb{E}\left(\mathcal{A}(\mathbf{OPT}) - \mathcal{A}(S')\right) &\leqslant \mathbb{E}(\mathcal{A}(\mathbf{OPT} \cap border)) \\ &\leqslant \int_{p \in B(\mathbf{OPT})} P(p \text{ is at distance at most 4 from the grid}) \mathrm{d}p \\ &\leqslant \int_{p \in B(\mathbf{OPT})} 2 \cdot \frac{4}{L_0} \mathrm{d}p \\ &\leqslant \frac{8 \cdot \mathbf{OPT}}{L_0} = \mathcal{O}(\varepsilon \mathbf{OPT})\end{aligned}$$

By choosing the constant properly in the big O notation and using the Markov inequality, we can show that the probability of $|S_{add}| > O(\varepsilon k)$ and the probability of $A(OPT) - A(S) > O(\varepsilon OPT)$ are both upper bounded by $\frac{1}{3}$. Thus, by a union bound, we conclude the proof. ◁

Proof of Claim 23. For (i), we just need to argue that for each leaf square C, after $\overline{H}$ is defined, $S \cup S_{add} \setminus \overline{H}$ is a dominating set in $UDG(S \cup S_{add})$ (then the proof follows from Claim 22). Indeed if a disk x is in $\overline{H}$ then it means that $\mathcal{A}(H \cup \{x\}) - \mathcal{A}(H) < \beta \leqslant 1$. In particular, it implies that there exists a disk in $H \subseteq S \cup S_{add} \setminus \overline{H}$ that intersects x.

For (ii), observe that the size of $S \cap C$ is the sum of the size of the corresponding sets H and D' built during the "sparsification" of C. Since all disks in H increases the area covered by at least β and are contained in a square of area L_0^2, the number of disks in H is upper-bounded by $\beta^{-1}L_0^2$. Moreover, each connected component of $S \cup S_{add} \setminus \overline{H}$ had a disk contained in C so that the number μ of connected component is upper-bounded by $L_0^2/\pi < L_0^2/2$. Therefore $|D'| < L_0^2$. Finally $|H \cup D'| < (1 + \beta^{-1})L_0^2 = k_0$.

For (iii), we start by observing that the union $B(S')$ of disks in S' is contained in the set $B^+(S)$, which is defined as

$$B^+(S) := \{z \in \mathbb{R}^2 \mid \exists x \in S \text{ such that } ||z - x|| \leqslant 1 + \beta\}$$

Indeed, if there exists a point p covered by a disk x in S' but at distance at least $1 + \beta$ from any disk of S then adding x to S would increase the area covered by S by more that β.

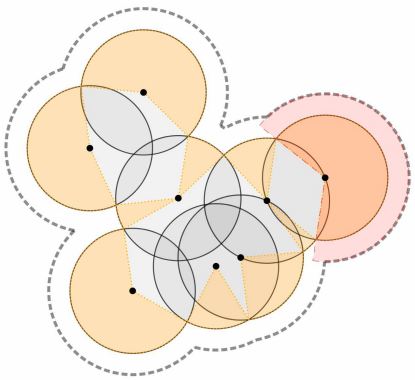


Figure 8 S consists of grey disks. The boundary of $B^+(S)$ is the dotted curve. Circular sectors are in orange while the red one represents a circular sector in $B^+(S)$.

Therefore, we have the following inclusion

$$B(S) \subseteq B(S') \subseteq B^+(S), \tag{1}$$

and if the following geometrical claim holds, our proof of (iii) will be complete.

▷ Claim 28. $\mathcal{A}(B(S)) \geqslant (1-\varepsilon/2)\mathcal{A}(B^+(S))$

The result follows from the fact that $B(S)$ is a union of unit-disks. See Figure 8. The boundary of $B(S)$ is made of *circular arcs* and each of these arcs is associated with a *circular sector* θ_i. Circular sectors intersect with other circular sectors only on the extreme points of their corresponding arcs, thus $\mathcal{A}(\cup_i \theta_i) = \sum_i \mathcal{A}(\theta_i)$.

We can associate with each circular sector θ_i (of a disk of radius 1) its "dilation" θ_i^+ which corresponds to the same circular sector in a disk of radius $1+\beta$. We have $\mathcal{A}(\theta_i^+) = (1+\beta)^2\mathcal{A}(\theta_i)$ and can see that $B^+(S) \setminus B(S) \subseteq \bigcup_i (\theta_i^+ \setminus \theta_i)$. Then

$$\begin{aligned}
\mathcal{A}(B^+(S)) - \mathcal{A}(B(S)) = \mathcal{A}(B^+(S) \setminus B(S)) &= \mathcal{A}\left(\bigcup_i (\theta_i^+ \setminus \theta_i)\right) \\
&\leqslant \sum_i \mathcal{A}(\theta_i^+ \setminus \theta_i) = \sum_i \mathcal{A}(\theta_i^+) - \mathcal{A}(\theta_i) \\
&\leqslant \sum_i (1+\beta)^2\mathcal{A}(\theta_i) - \mathcal{A}(\theta_i) \\
&\leqslant \sum_i 3\beta\mathcal{A}(\theta_i) = 3\beta\mathcal{A}\left(\bigcup_i \theta_i\right) \leqslant 3\beta\mathcal{A}(B(S))
\end{aligned}$$

Therefore, $\mathcal{A}(B(S)) \geqslant \dfrac{\mathcal{A}(B^+(S))}{1+3\beta} \geqslant (1-\varepsilon/2)\mathcal{A}(B^+(S))$. This concludes the proofs of Claims 28 and 23. ◁

Proof of Theorem 24.

Size of *tab*. There exists 4^i squares at level i so the total of squares is $\sum_{i=0}^{d} 4^i = \mathcal{O}(4^{d+1})$. For any square C, the number of potential portal disks is at most $4m$. To see this, observe that if C is of level i, it is of size $L/2^i \times L/2^i$. Furthermore, it is surrounded by lines of level at most i and two adjacent portals on such a line has distance $\Omega(L/(m2^i))$.
Therefore, the number of possible sets $P \subseteq \mathcal{P}_C$ is 2^{4m}, and for each set P of size r the total number of planar connectivity relations is equal to the r-th *Catalan number* : $P(r) = \dfrac{1}{r-1}\binom{2r}{r} = \mathcal{O}\left(\dfrac{1}{m-1}\binom{8m}{m}\right)$ and then by Stirling formula we get $P(r) = \mathcal{O}(4^{4m})$. To see that $P(r)$ is the r-th Catalan number, we check that it satisfies the same recurrence relation :

$$P(r) = \sum_{k=1}^{r} P(k-1) \cdot P(r-k) \tag{2}$$

with $P(0) = 1$. Indeed, if k denotes the index of the first portal p_k that is on the connected component of the r-th portal disk p_r, then the portal disk p_i with $1 \leqslant i \leqslant k-1$ cannot be equivalent to a portal p_j disk with $k \leqslant j \leqslant n$, and then the equivalence relation can be restricted to the set $\{p_i, 1 \leqslant i \leqslant k-1\}$ and there are $P(k-1)$ possible distinct choices. Next observe that since p_n and p_k are connected (i.e. $p_n \sim p_k$), it is enough to count the number of different equivalence relations in $\{p_j, k+1 \leqslant j \leqslant r\}$, which is $P(r-k)$.

Finally, observe that k can be from 1 to r ($k = r$ means that p_r is alone in its connected component.) We thus concludes (2). Therefore, creating *tab* in line 6 can be done in time $\mathcal{O}(4^{d+1}\varepsilon k^2 8^{4m}) = k^{\mathcal{O}(1/\varepsilon)}$.

Initialization. There exists 4^d leaf squares and for each of them, we try all possible guesses. This can be done in time $n^{\mathcal{O}(\varepsilon^{-3})}$.

Fusion. Trying all possible combinations can be done in time $k^{\mathcal{O}(1/\varepsilon)}$ ◀

Proof of Lemma 27. Let X be a set of points of the plane, k a positive integer and $\eta \leqslant 1/2$ a parameter. We prove a stronger result. Given any solution feasible solution S to $\mathrm{MACS}(X, k(1+\eta))$, there exists a subset S' of S that is a feasible solution to $\mathrm{MACS}(X, k)$ with value at least $(1-10\eta)\mathcal{A}(S)$. Obviously Lemma 27 follows when S is optimal. If $\mathcal{A}(S) \geqslant k/3$, then remove ηk disks from S without disconnecting S. For instance, consider a spanning tree on $UDG(S)$ and remove the nodes from the leaves to the root until you reach the desired size. Let S' denote the subset obtained.

$$\mathcal{A}(S') \geqslant \mathcal{A}(S) - \eta\pi k \geqslant (1-3\pi\eta)\mathcal{A}(S) \geqslant (1-10\eta)\mathcal{A}(S)$$

If $\mathcal{A}(S) < k/3$, let I be a maximal independent set in S. We have $|I|\pi = \mathcal{A}(I) \leqslant \mathcal{A}(S) < \frac{k}{3}$. According to claim 22, there exists a connected dominating set $I \subseteq D \subseteq S$ in S of size at most $3|I| - 2 < k/\pi < \frac{k}{3}$. Consider a set $H \subseteq S \setminus D$ of size $k - |D| > \frac{2}{3}k$ built by greedily adding a disk $h \in S \setminus (D \cup H)$ maximising the marginal area $\mathcal{A}(D \cup H \cup \{h\}) - \mathcal{A}(D \cup H)$. Since D is a connected dominating set, the set $S' := D \cup H$ is connected. Since all disk where added greedily in H, for all $H \in S \setminus S'$, we have

$$\mathcal{A}(S' \cup \{h\}) - \mathcal{A}(S')) \leqslant \frac{\mathcal{A}(S) - \mathcal{A}(D)}{|H|} \leqslant \frac{2\mathcal{A}(S)}{k}$$

.

By submodularity, we deduce that $\mathcal{A}(S) - \mathcal{A}(S') \leqslant \eta k \cdot \frac{2\mathcal{A}(S)}{3k}$. That implies $\mathcal{A}(S') \geqslant (1 - \frac{3}{2}\eta)\mathcal{A}(S)$. This concludes the proof of lemma 27. ◀

Remark that this proof is constructive and it is easy to check that finding S' from any given set S can be done in polynomial time.

Proof of Lemma 25. Let us use the same notation as in the statement of Lemma 25. We prove how to build S' from S_{add} such that $|S'| \leqslant (22\alpha+4)|S_{add}|$ while preserving connectivity.

Let Y be a connected component of S_{add}. We prove that we can find a set $Y' \subseteq X$ of input disks such that $|Y'| \leqslant (4+22\alpha)|Y|$ and $(S_{add} \setminus Y) \cup (S \cup Y')$ is connected. Removing Y might split the solution into several connected components $F_1, \ldots, F_s$. For each connected component F_i, pick one disk x_i in $F_i \cap X$ that intersects Y.

Step 1. Each additional disk y in Y is adjacent to at most 6 disks x_i. We can connect the corresponding connected component by using 20α disks of the input. Indeed, any two x_i and x_j adjacent to y has a Euclidean distance at most 4. Since X is well-distributed their distance in $UDG(X)$ is at most 4α. Then, we can find $\lceil 4\alpha - 1 \rceil$ disks in X which connect x_i and x_j. In order to connect all the x_i that are adjacent to y, it is sufficient to repeat this operation 5 times, which asks at most 20α disks. We can perform this operation for each additional disk that was not already considered. Then, in total for this first step we need to use at most $20\alpha|Y|$ disks.

Step 2. During step 1, we may have connected some disks x_i, so that the number of connected components has decreased. The number of connected components is $s' \leqslant s$, each of them corresponds to a disk x_i, and without loss of generality we can assume that the corresponding indexes are such that $1 \leqslant i \leqslant s'$. Let T be a spanning tree on $UDG(Y)$. Without loss of generality, we can suppose that indexes i are such that the sequence $(x_1, \ldots, x_{s'})$ correspond to a T transversal. Note that after step 1, each x_i can be associated to a different y in Y. Then, we reconnect each x_i to x_{i+1} for $1 \leqslant i \leqslant s-1$. If x_i and x_{i+1} are respectively associated to y_i and y_{i+1}, then $||x_i - x_{i+1}|| \leqslant 2 + 2d_T(y_i, y_{i+1})$ and thus $d_{UDG(X)}(x_i, x_{i+1}) \leqslant \lceil \alpha(2 + 2d_T(y_i, y_{i+1}) \rceil$. Then, we can find $\lceil \alpha(2 + 2d_T(y_i, y_{i+1}) \rceil - 1$ disks in X to connect x_i and x_{i+1}. In order to connect all x_i we need to use at most

$$\sum_{i=1}^{s'-1} \lceil \alpha(2 + 2d_T(y_i, y_{i+1}) \rceil - 1 \leqslant 2(s'-1)\alpha + 2\sum_{i=1}^{s'-1} d_T(y_i, y_{i+1})$$

input disks. Since the order corresponds to a T transversal, each edge is visited at most twice and then $\sum_{i=1}^{s'-1} d_T(y_i, y_{i+1}) \leqslant 2(|Y| - 1)$. Therefore the total number of disks that were added during this second step is bounded by $|Y|(4 + 2\alpha)$.

We proved that there exists a subset $Y' \subseteq X$ of size at most $(4 + 22\alpha)|Y|$ such that $(S \cup S_{add} \setminus Y) \cup Y'$ is connected. By doing so for each connected component of S_{add}, we get the result claimed. ◀

Proof of Lemma 8. Let X be a pseudo-convex set, G its unit-disk-graph, and x and y be any two disks in X at distance $L = \|x - y\|$. We show that $d_G(x, y) \leqslant \lceil \alpha L \rceil$ where $\alpha = 12/\pi < 3.82$.

If $L < 2$ then the two unit disks associated to x and y overlap so that $d_G(x, y) = 1 \leqslant \lceil \alpha L \rceil$. Otherwise suppose that $L \geqslant 2$. Since X is pseudo-convex, it is connected and any point in the line segment $[x, y]$ is covered by a disk in X. Let $S = \{z \in X \mid B_z \cap [x, y] \neq \emptyset, \|x - z\| > 2 \text{ and } \|y - z\| > 2\}$ and let I be any maximal independent set in $S \cup \{x, y\}$. Since S is at distance at least 2 from x and y, we deduce that $x, y \in I$ and all disks in $I \setminus \{x, y\}$ are inside a $L \times 4$ rectangle and then $|I| \leqslant 4L/\pi$. Since I is maximal, it is a dominating set in S. Therefore, claim 22 implies that there exists a connected subset $D \subseteq X$ such that $I \subseteq D$ and $|D| \leqslant 3|I| - 2 \leqslant 12L/\pi - 2$. We deduce that $d_G(x, y) \leqslant (12L/\pi - 2) + 1 \leqslant \lceil \alpha L \rceil$. ◀

Proof of Claim 26. The solution output by Algorithm 2 on input (X, k', ε') verifies the following properties: $S \cup S_{add}$ is connected, the size of S and S_{add} are respectively upper-bounded by k' and $\varepsilon' k'$ and $\mathcal{A}(S) \geqslant (1 - \varepsilon')\mathbf{OPT}(X, k')$. Therefore, the set S' given by Lemma 25 has size at most $(22\alpha + 4)|S_{add}| \leqslant (22\alpha + 4)\varepsilon' k'$, and then $|S \cup S'| \leqslant k' + (22\alpha + 4)\varepsilon' k' \leqslant (1 + (22\alpha + 4)\varepsilon')k' = k$. Since $S \cup S'$ is connected, this set is a feasible solution to $\text{MACS}(X, k)$.

Finally, from Lemma 27 with parameter $\eta = (22\alpha + 4)\varepsilon'$, we get that the area covered by this solution is

$$\begin{aligned}\mathcal{A}(S \cup S') &\geqslant \mathcal{A}(S) \geqslant (1 - \varepsilon')\mathbf{OPT}(X, k') \geqslant (1 - \varepsilon')(1 - 10\eta)\mathbf{OPT}(X, k'(1 + \eta)) \\ &\geqslant (1 - \varepsilon')(1 - 10(22\alpha + 4)\varepsilon')\mathbf{OPT}(X, k'(1 + (22\alpha + 4)\varepsilon')) \\ &\geqslant (1 - \varepsilon)\mathbf{OPT}(X, k)\end{aligned}$$

which concludes the proof. ◁

Robust Correlation Clustering

Devvrit
BITS Pilani, Goa Campus, Goa, India
devvrit.03@gmail.com

Ravishankar Krishnaswamy
Microsoft Research, Bengaluru, India
rakri@microsoft.com

Nived Rajaraman
IIT Madras, Chennai, India
nived.rajaraman@gmail.com

Abstract

In this paper, we introduce and study the Robust-Correlation-Clustering problem: given a graph $G = (V, E)$ where every edge is either labeled $+$ or $-$ (denoting similar or dissimilar pairs of vertices), and a parameter m, the goal is to delete a set D of m vertices, and partition the remaining vertices $V \setminus D$ into clusters to minimize the cost of the clustering, which is the sum of the number of $+$ edges with end-points in different clusters and the number of $-$ edges with end-points in the same cluster. This generalizes the classical Correlation-Clustering problem which is the special case when $m = 0$. Correlation clustering is useful when we have (only) qualitative information about the similarity or dissimilarity of pairs of points, and Robust-Correlation-Clustering equips this model with the capability to handle noise in datasets.

In this work, we present a *constant-factor* bi-criteria algorithm for Robust-Correlation-Clustering on complete graphs (where our solution is $O(1)$-approximate w.r.t the cost while however discarding $O(1)m$ points as outliers), and also complement this by showing that no finite approximation is possible if we do not violate the outlier budget. Our algorithm is very simple in that it first does a simple LP-based *pre-processing* to delete $O(m)$ vertices, and subsequently runs a particular Correlation-Clustering algorithm ACNAlg [2] on the residual instance. We then consider general graphs, and show $(O(\log n), O(\log^2 n))$ bi-criteria algorithms while also showing a hardness of α_{MC} on both the cost and the outlier violation, where α_{MC} is the lower bound for the Minimum-Multicut problem.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Facility location and clustering

Keywords and phrases Correlation Clustering, Outlier Detection, Clustering, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.33

Category APPROX

1 Introduction

Clustering is one of the most widely used tools in various scientific disciplines (such as biology, computer science, machine learning and operations research to name a few) due to its wide applicability in these domains. Broadly speaking, the goal of clustering is to partition a given dataset into a number of clusters such that data items in the same cluster are more alike each other than data items in different clusters. In many application domains, the data items are naturally represented as points in a metric space, and the distance between the corresponding vectors is used as a measure of (dis)similarity. In such cases, clustering formulations such as k-median or k-means are the de-facto standards to utilize. However, there are also quite a few application domains where the information available to us is simply


© Devvrit, Ravishankar Krishnaswamy, and Nived Rajaraman;
licensed under Creative Commons License CC-BY

whether different pairs of data items are similar or dissimilar to each other. Examples of such settings where there is only qualitative information include data items being web-pages on the internet, a collection of people on a social network or even a group of proteins. Motivated by such settings, Bansal et al. [3] formulated a problem known as *correlation clustering* (in fact, a similar problem was implicitly studied by Ben-Dor et al.[4] as 'Cluster Editing').

▶ **Problem 1** (Correlation-Clustering). *We are given a complete graph $G = (V, \binom{V}{2})$, and a labelling of each edge as either positive or negative, denoting whether the end vertices of the edge are similar to each other or dissimilar. In other words, the edge set $\binom{V}{2}$ is partitioned into $E_+ \dot\cup E_-$ where E_+ denotes the similar pairs and E_- denotes dissimiliar pairs. The goal is to compute a partition $\mathcal{C} = \{C_1, C_2, \ldots, C_r\}$ of V (so $V = \dot\cup_{1 \le i \le r} C_i$ is a disjoint union of the C_i's) to minimize the* cost *of the clustering, which is the total number of E_+ edges with end-points in different clusters and E_- edges with end-points in the same cluster.*

A nice modeling aspect of this problem is that the number of clusters is not specified as part of the input, and rather, left to the algorithm. This makes it a compelling problem when we do not have a priori knowledge of the number of clusters we seek in the final partitioning.

Since being introduced formally as an optimization problem, there have been numerous works trying to understand the computational complexity of the problem. Bansal et al. [3] show that the problem is APX-hard (ruling out the design of PTASes unless P=NP) and obtain a constant-factor *approximation algorithm* for this problem. Subsequently, there have been a series of works (see, e.g., the survey by Wirth [23]) getting better factors, with the current best bound being a factor of 2.06 due to Chawla et al. [8].

Despite the simplicity and elegance of the various clustering formulations described thus far, a significant shortcoming of most of them is that they are not robust to noisy points. For example, the presence of a few outliers in the data set can completely change the *cost* and *structure* of solutions obtained by running clustering algorithms for k-median, k-means, etc. Indeed, this has prompted much recent study in the CS, ML and statistics communities of *robust* versions of these problems [6, 10, 17]. Motivated by this observation, and the fact that real-world data sets are often noisy, we investigate the *robustness* of correlation clustering.

▶ **Problem 2** (Robust-Correlation-Clustering). *The input to this problem is identical to the correlation clustering instance as in Problem 1. Additionally, we are also given a parameter m, which denotes the number of points we can discard while clustering. The goal is to identify a set $D \subseteq V$ of* outliers *of size m, and cluster the remaining points $V \setminus D$ to minimize the cost of the resulting clustering, i.e., the total number of E_+ edges (resp. E_- edges) in $V \setminus D$ with end-points in different clusters (resp. same cluster).*

We note that Correlation-Clustering problem also makes sense when the edge set $E_+ \cup E_-$ is not the complete graph, since we often do not have complete information about the (dis)similarity of each pair of points (it could be expensive or even impossible to obtain such information like in the case of protein-protein interactions). Now the problem becomes much harder, and the current best known algorithms have approximation guarantees of a factor of $O(\log n)$. Moreover, there is an approximation-preserving reduction from the Minimum-Multicut problem, for which the best known approximation is an $O(\log n)$ factor [5]. In this paper, we also consider the Robust-Correlation-Clustering problem on general graphs, analogous to the study of Correlation-Clustering in general graphs [5].

▶ **Problem 3** (Robust-Correlation-Clustering on General Graphs). *The problem is identical to Problem 2, with the exception that the union of E_+ and E_- need not be $\binom{V}{2}$.*

1.1 Our Results

Having introduced the problem, the first question we address is whether the Correlation-Clustering objective is indeed susceptible to outliers in the dataset. That is, we seek to understand whether the solution cost and/or structure can change a lot by the removal of a few points in the dataset. Classical objectives such as k-median and k-means suffer from this drawback *even in the simplest of settings* when we are promised that after removing some m data-points, *the optimal clustering of the remaining points would have* 0 *cost.* In such cases, solving k-means objective on the original instance could yield very different solutions than the intended solution, which is the 0 cost (or perfect clustering).

Somewhat surprisingly, our first simple observation is that the correlation clustering objective is inherently robust to an extent, at least in the case when the cost of the clustering after removing m outliers becomes 0. We show that in this case, the optimal correlation clustering solution and the optimal robust correlation clustering solution are structurally identical upto $O(m)$ points.

▶ **Theorem 4.** *Consider an instance $\mathcal{I}$ of* Robust-Correlation-Clustering *on complete graphs such that $\mathsf{Opt}(\mathcal{I}) = 0$, i.e., there exists a set $D^* \subseteq V$ of m vertices deleting which, the subgraph induced by $V \setminus D^*$ admits a perfect clustering $\mathcal{C}^*$. Then, consider any optimal solution $\widetilde{\mathcal{C}}$ to* Correlation-Clustering *(Problem 1). There exists a set $\widetilde{D}$ of $O(m)$ vertices s.t. the cost of $\widetilde{\mathcal{C}} \setminus \widetilde{D}$[1] has objective function value 0.*

This theorem in fact sets apart the correlation clustering objective from other clustering objectives such as k-means and k-median where an analogous statement to Theorem 4 does not hold. Moreover, we believe that a similar result is true even when $\mathsf{Opt}(\mathcal{I}) \neq 0$ when comparing the optimal solutions of the robust and non-robust problems.

Now, while this exhibits the robustness of correlation clustering w.r.t. *optimal solutions*, the problem is APX-hard and hence we typically do not deal with optimal solutions. Hence, we next consider the same question, but for approximation algorithms.

▶ **Theorem 5.** *There exists an instance $\mathcal{I}$ of* Robust-Correlation-Clustering *on complete graphs which satisfies the following properites: (a) $\mathsf{Opt}(\mathcal{I}) = 0$, i.e., there exists a set $D \subseteq V$ of $m = O(\sqrt{n})$ vertices deleting which, the subgraph induced by $V \setminus D$ admits a perfect clustering, and (b) there exists a constant-factor approximately optimal solution $\mathcal{C}$ to the* Correlation-Clustering *objective function (1), such that, for any set S of $< n - 1$ vertices, the cost of the clustering $\mathcal{C} \setminus S$ is still non-zero.*

This then provides sufficient motivation for undertaking this study, with the main focus of whether we can design efficient approximation algorithms for Robust-Correlation-Clustering. Our first result in this direction is a negative result, which says that it is in fact NP-hard to obtain any finite approximation algorithm for Robust-Correlation-Clustering, even on complete graphs. This is in stark contrast to Problem 1, where we know very good constant-factor approximations.

▶ **Theorem 6.** *It is NP-hard to obtain any finite approximation factor for* Robust-Correlation-Clustering *on complete graphs, unless we violate the budget on the number of outliers.*

[1] We somewhat abuse notation to let $\mathcal{C} \setminus D$ to denote the clustering obtained by removing the points in D from the clustering $\mathcal{C}$.

We therefore seek to obtain *bi-criteria approximation algorithms*: an (a, b) bi-criteria approximation for Robust-Correlation-Clustering is one where the solution's cost is at most a times the optimal cost, and the number of outliers in our solution is at most $b \cdot m$.

▶ **Theorem 7.** *There is an efficient bi-criteria* $(6, 6)$*-approximation algorithm for* Robust-Correlation-Clustering *on complete graphs.*

Our algorithm is extremely simple: it essentially does a simple LP-based *pre-processing* step to prune out a set of $O(m)$ outliers, and then executes a classical algorithm for Correlation-Clustering [2] (henceforth called ACNAlg) on the remaining vertices. This approach works because the LP relaxation which [2] uses for solving Correlation-Clustering is a purely covering LP (as opposed to the more natural metric LP relaxation for Correlation-Clustering), and can easily be adapted to incorporating outliers. We remark that, owing to the pre-processing step, our overall algorithm requires solving an LP: it would be very interesting to develop a purely combinatorial algorithm for Robust-Correlation-Clustering on complete graphs. It might even be possible for a simple adaptation of the ACNAlg algorithm to be a constant-factor bi-criteria approximation. We leave this as an important avenue of future research.

Finally, we turn our attention to Robust-Correlation-Clustering on general graphs, where we show poly-logarithmic bi-criteria algorithms and logarithmic hardness results on both the cost as well as the outlier budget. While the Correlation-Clustering problem is equivalent to Minimum-Multicut [14] and we can use any Minimum-Multicut algorithm to solve the problem, we show that one specific technique based on *padded decompositions* of metric spaces naturally lends itself to solving the robust problem.

▶ **Theorem 8.** *There is an efficient bi-criteria* $(O(\log n), O(\log^2 n))$*-approximation algorithm for* Robust-Correlation-Clustering *on general graphs.*

▶ **Theorem 9.** *It is NP-hard to obtain any bi-criteria* (a, b)*-approximation algorithm for* Robust-Correlation-Clustering *on general graphs for* $b < \alpha_{\text{MC}}$ *or* $a < \alpha_{\text{MC}}$ *where* α_{MC} *is the inapproximability factor for the* Minimum-Multicut *problem.*

It would be interesting to resolve the gap between the $O(\log^2 n)$ upper bound and the $\Omega(\log n)$ lower bound for the outlier budget violation.

1.2 Related Work

Since its introduction, Correlation-Clustering has received much attention with focus on designing better algorithms (see the survey of [23]), faster algorithms in the parallel and distributed [11] and streaming settings [1], stochastic/average-case settings [19], and applications [12, 13, 20]. There is also work on a related objective function of *maximizing* the number of classified edges [3]. Being a maximization objective, it is easier to design simple constant-factor approximation algorithms like random partitions, etc. There are however, better SDP-based approximation algorithms [5, 22].

Recently there has also been a large body of work on the crucial problem of noise-resilient or *robust* clustering for distance-based clustering objectives such as k-means [10, 17], and designing faster algorithms [7, 21, 16], and parallel and distributed algorithms in this model [9, 18]. To the best of our knowledge, this is the first work to study the Correlation-Clustering problem from robustness point of view.

1.3 Paper Outline

We first describe the inherent robustness to outliers of *optimal solutions* for Correlation-Clustering in Section 2. We then consider Robust-Correlation-Clustering for complete graphs, and show our hardness of approximation in Section 3, followed by the bi-criteria algorithm in Section 4. Finally, in Section 5 and Appendix A, we turn our attention to the case of general graphs and present our algorithm and hardness.

2 Robustness of the Correlation-Clustering Objective

In this section, we show two simple but illuminating results. The first result explains how, in contrast to problems like k-median and k-means, the vanilla correlation clustering objective is in fact *inherently robust* to an extent, *when solved optimally*. The second result then shows this not to be true when considering solutions which are only approximately optimal. We remark that the second result and that fact that correlation clustering is APX-hard [3] serves as a strong motivation for studying the Robust-Correlation-Clustering problem.

2.1 Optimal Correlation-Clustering Solutions are Robust

In this section, we exhibit the inherent robustness of the correlation clustering objective (1) in a specialized scenario. Indeed, consider an instance $\mathcal{I}$ of Robust-Correlation-Clustering such that $\mathsf{Opt}(\mathcal{I}) = 0$, i.e., there exists a set of m points deleting which the remaining points are perfectly clusterable, i.e., have 0 cost. Now, imagine we obtain an optimal Correlation-Clustering solution (Problem 1) to instance $\mathcal{I}$. We show that there exist $O(m)$ points, deleting which, the cost indeed becomes 0 for this solution. This tells us that the optimal solutions to 2 and 1 are nearly identical to each other (upto $O(m)$ points), and hence, that the correlation clustering objective is inherently robust!

Proof of Theorem 4. We begin by recalling the theorem statement and setting up notation. Let $\mathcal{I}$ be an instance of Robust-Correlation-Clustering such that $\mathsf{Opt}(\mathcal{I}) = 0$, i.e., there exists a set $D^* \subseteq V$ of m vertices deleting which, the subgraph induced by $V \setminus D^*$ admits a perfect clustering $\mathcal{C}^*$. And consider any optimal solution $\widetilde{\mathcal{C}}$ to instance $\mathcal{I}$ w.r.t the Correlation-Clustering objective function (1). We would like to claim that there exists a set $\widetilde{D}$ of $O(m)$ vertices such that $\widetilde{\mathcal{C}} \setminus \widetilde{D}$ is identical to $\mathcal{C}^* \setminus \widetilde{D}$. We show this by showing that the cost of the clustering $\widetilde{\mathcal{C}} \setminus \widetilde{D}$ is 0, and hence it must be the same as $\mathcal{C}^* \setminus \widetilde{D}$.

To this end, let $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_r^*\}$ denote the optimal Robust-Correlation-Clustering clustering over vertices $V \setminus D^*$, and let $\widetilde{\mathcal{C}} = \{\widetilde{C}_1, \widetilde{C}_2, \ldots, \widetilde{C}_s\}$ denote the optimal Correlation-Clustering clustering over all vertices V. We divide the clusters in $\widetilde{\mathcal{C}}$ into two types:

(a) A cluster $\widetilde{C} \in \widetilde{\mathcal{C}}$ is a *mixed cluster* if it contains points from more than one cluster in $\mathcal{C}^*$, i.e., there exists i_1, i_2 s.t $|\widetilde{C} \cap C_{i_1}^*| > 0$ and $|\widetilde{C} \cap C_{i_2}^*| > 0$, and

(b) A cluster $\widetilde{C} \in \widetilde{\mathcal{C}}$ is an *isolated cluster* if it contains points from only one cluster in $\mathcal{C}^*$.

We then show that the total number of points in mixed clusters is $O(m)$, and can simply add all such points to $\widetilde{D}$. At this point, we would only be left with isolated clusters. Subsequently, we show that two isolated clusters composed of points from the same cluster in $\mathcal{C}^*$ can contain at most $O(m)$ points. Therefore, we once again add these points to $\widetilde{D}$. Finally, we add all the remaining set of at most m outliers to $\widetilde{D}$. It is easy to see that the resulting clustering $\widetilde{\mathcal{C}} \setminus \widetilde{D} = \mathcal{C}^* \setminus D^*$. These results are established in Lemmas 10 and 11. ◀

▶ **Lemma 10.** *Let $\widetilde{C}$ be a mixed cluster, and let $X = \widetilde{C} \cap D^*$ denote its overlap with the outlier set R^* in the optimal* Robust-Correlation-Clustering *clustering. Then we have $|\widetilde{C}| \leq O(1)|X|$.*

Proof. Since $\widetilde{C} \in \widetilde{\mathcal{C}}$ is a mixed cluster, there exists $i_1 \neq i_2$ s.t $|\widetilde{C} \cap C^*_{i_1}| > 0$ and $|\widetilde{C} \cap C^*_{i_2}| > 0$. Now, since $\widetilde{\mathcal{C}}$ is an optimal solution for Correlation-Clustering, we have that the cost of the clustering must increase when we consider the following clustering $\widetilde{\mathcal{C}}_1 = (\widetilde{\mathcal{C}} \setminus \widetilde{C}) \cup (\widetilde{C} \cap C^*_{i_1}) \cup (\widetilde{C} \setminus C^*_{i_1})$ formed by replacing $\widetilde{C}$ with $(\widetilde{C} \cap C^*_{i_1})$ and $(\widetilde{C} \setminus C^*_{i_1})$. since $\mathcal{C}^*$ is an optimal clustering with cost 0, we know that all the edges between $C^*_{i_1}$ and C^*_i for $i \neq i_1$ belong to E_-. This, combined with the fact that the cost of this new clustering is more than that of $\widetilde{\mathcal{C}}$ gives us the following inequality:

$$|\widetilde{C} \cap C^*_{i_1}| \left(\sum_{i \neq i_1} |\widetilde{C} \cap C^*_i| \right) \leq |X||\widetilde{C} \cap C^*_{i_1}|$$
$$\implies \sum_{i \neq i_1} |\widetilde{C} \cap C^*_i| \leq |X| \tag{1}$$

A similar argument by replacing $\widetilde{C}$ with $(\widetilde{C} \cap C^*_{i_2})$ and $(\widetilde{C} \setminus C^*_{i_2})$ would yield $\sum_{i \neq i_2} |\widetilde{C} \cap C^*_i| \leq |X|$. Summing the two inequalities, we get that $|\widetilde{C} \setminus X| \leq 2|X|$, and so $|\widetilde{C}| \leq 3|X|$, completing the proof. ◀

▶ **Lemma 11.** *Let $\widetilde{C}_1$, $\widetilde{C}_2$ be two isolated clusters containing points from the same cluster $C^* \in \mathcal{C}^*$, and let $X_1 = \widetilde{C}_1 \cap D^*$ and $X_2 = \widetilde{C}_2 \cap D^*$ denote their intersections with the outlier set R^* in the optimal* Robust-Correlation-Clustering *clustering. Then we have $|\widetilde{C}_1 \cup \widetilde{C}_2| \leq O(1)|X_1 \cup X_2|$.*

Proof. Since $\widetilde{\mathcal{C}}$ is an optimal solution w.r.t the Correlation-Clustering objective, we know that if we modify $\widetilde{\mathcal{C}}$ by moving the points $\widetilde{C}_1 \cap C^*$ to cluster $\widetilde{C}_2$, the cost does not decrease. This gives us the following inequality, which uses the fact that all edges within C^* belong to E_+ due to the fact that cost of $\mathcal{C}^*$ is 0:

$$|\widetilde{C}_1 \cap C^*||\widetilde{C}_2 \cap C^*| \leq (|X_1| + |X_2|)|\widetilde{C}_1 \cap C^*|$$
$$\implies |\widetilde{C}_2 \cap C^*| \leq |X_1| + |X_2|$$

A similar argument would also give us $|\widetilde{C}_1 \cap C^*| \leq |X_1| + |X_2|$. Adding these inequalities gives us $|\widetilde{C}_1 \cap C^*| + |\widetilde{C}_2 \cap C^*| \leq 2(|X_1| + |X_2|)$, and adding back X_1 and X_2 will incur an additional cost of $|X_1| + |X_2|$, hence completing the proof. ◀

2.2 Approximate Solutions may not be Robust

We next focus on *approximation algorithms* to Correlation-Clustering, and show that they need not be robust to outliers (Theorem 5). Indeed, consider the following instance $\mathcal{I} = (V, E)$ of Robust-Correlation-Clustering with $n + \sqrt{n}$ points. Consider a $\sqrt{n} \times \sqrt{n}$ grid, such that all points lying on the same row are pairwise similar, i.e., belong to E_+ while any two points lying on different rows are dissimilar and belong to E_-. To this arrangement, $\sqrt{n}$ *bad points* are added, which are pairwise dissimilar to one another, but share a $+$ edge with each of the n points in the original $\sqrt{n} \times \sqrt{n}$ grid.

We first note that the optimal Correlation-Clustering solution to $\mathcal{I}$ has cost $\Omega(n\sqrt{n})$. Indeed, consider any triangle u, v, w where u is a bad point, and v and w belong to different rows. Note that there must at least be one mis-classified edge in this triangle in the optimal

solution. So, if we let $\mathcal{B}$ denote the set of all such bad triangles, the following is a valid lower bound on OPT: $\min \sum_{e \in t, t \in \mathcal{B}} z_e$ s.t $\sum_{e \in t} z_e \geq 1, \forall t \in \mathcal{B}$. The dual of this is $\max \sum_{t \in \mathcal{B}} y_t$ s.t $\sum_{t: e \in t, t \in \mathcal{B}} y_t \leq 1, \forall e \in E$. It is easy to see that the optimal value of the dual LP is at least $\Omega(n\sqrt{n})$ by setting $y_t = 1/n$ for all bad triangles in $\mathcal{B}$. Now consider a clustering $\mathcal{C}$ which *clusters each column of the grid* into a cluster, and puts the bad points in another cluster. The overall cost of the clustering is $O(n\sqrt{n})$, which is a constant-factor approximation. Moreover, note that the only way to get a 0 cost clustering from $\mathcal{C}$ (without altering the structure of $\mathcal{C}$) is by deleting all the n grid points.

3 Robust-Correlation-Clustering on Complete Graphs: Hardness

In this section, we give the proof of Theorem 6. The proof follows by an approximation preserving reduction from vertex cover. Consider an instance $\mathcal{I}_{vc}$ of vertex cover, given by a graph, $G = (V, E)$ on n vertices. We construct the Robust-Correlation-Clustering instance $\mathcal{I}$ as follows: for each vertex $v \in V$, we create two points v_1 and v_2, giving us a total of $2n$ vertices in $\mathcal{I}$. For every vertex $v \in V$, we make the edge $(v_1, v_2) \in E_+$. Similarly, for any pair of vertices $u, v \in V$ the edges (u_2, v_2), (u_1, v_2) and (u_2, v_1) all belong to E_-. Finally, we place edge $(u_1, v_1) \in E_+$ if the edge $(u, v) \in E$, and in E_- otherwise. The outlier budget is some parameter m, unrelated to the number of edges in G.

▶ **Lemma 12.** *There exists a solution of cost* 0 *for* $\mathcal{I}$ *if* G *has a vertex cover of size* m*.*

Proof. Let $S \subseteq V$ denote a vertex cover of size m for G, and let $S_1 = \{v_1 : v \in S\}$. Then, consider the natural clustering $\mathcal{C} = \{\{v_1, v_2\} : v \in V\}$ comprising of the pairs of vertices. The only mis-classified edges in this clustering are of the form (u_1, v_1) corresponding to edges (u, v) of G. But now, suppose we declare the points in S_1 as outliers, then it follows that the resulting clustering $\mathcal{C} \setminus S_1$ has 0 cost, since S is a vertex cover for G. ◀

▶ **Lemma 13.** *If there is a set* S *of* m *outliers such that the remaining points has a* 0 *cost clustering* $\mathcal{C}$ *in instance* $\mathcal{I}$*, then* G *has a vertex cover of size at most* m *in instance* $\mathcal{I}_{vc}$*.*

Proof. We construct a candidate vertex cover S' for G from the outlier-set S as follows: for each $v \in V$, include $v \in S'$ if either v_1 or v_2 is in S. We claim then that S' is a valid vertex cover for G. To the contrary, suppose an edge (u, v) is not covered by S'. Then, none of the four points u_1, u_2, v_1, v_2 are included in the outlier-set S in the robust clustering solution. Now, since clustering $\mathcal{C}$ has 0 cost, it must be that the four points u_1, u_2, v_1 and v_2 must belong to the same cluster in $\mathcal{C}$, or else, one of the edges in (u_1, u_2), (u_2, v_2), and (v_2, v_1), all of which belong to E_+, would be mis-classified. But now the edges (u_1, v_2) and (v_1, u_2) belong to E_- and would be mis-classified in $\mathcal{C}$, which contradicts the fact that $\mathcal{C}$ has 0 cost. ◀

Theorem 6 then follows from Lemmas 12 and 13.

4 Robust-Correlation-Clustering on Complete Graphs: Algorithms

In this section, we design a simple LP-rounding based bi-criteria approximation algorithm for Robust-Correlation-Clustering (Problem 2) and prove Theorem 7. We begin by recalling the problem setup: we are given an instance $\mathcal{I}$ consisting of a graph (V, E_+, E_-) on n points with $E_+ \cup E_- = \binom{V}{2}$. The goal is to identify a set of vertices D such that $|D| = m$, and a clustering $\mathcal{C}$ over $V \setminus D$ such that the total cost is minimized. We start with the following definition crucial to the design and analysis of our algorithm.

▶ **Definition 14** (Bad Triangles). *A triplet (u, v, w) of points is said to be a* bad triangle *if exactly two of the three edges among $(u, v), (v, w), (u, w)$ belong to E_+ and one to E_-.*

Note a bad triangle captures the *smallest unit of inconsistency* in the similarity information among the points: either we delete one of the vertices as an outlier, or at least one of the edges must be mis-classified. In what follows, let $\mathcal{B}$ denote the set of all bad triangles in $\mathcal{I}$.

4.1 Recap of ACNAlg for Correlation-Clustering [2]

Since the crux of our algorithm is the ACNAlg for correlation clustering, we begin with a quick recap of ACNAlg. Essentially, the algorithm iteratively picks a *random* un-clustered vertex v as a new cluster center, and includes all other un-clustered vertices similar to v.

Algorithm 1 ACNAlg(V, E_+, E_-).

set $U = V$ and $C = \emptyset$ ▷ initialize set of un-clustered points and set of cluster centers
while $U \neq \Phi$ **do**
  sample $v \sim \text{Unif}(U)$
  update $C \leftarrow C \cup \{v\}$ ▷ random v is sampled as a cluster center
  let $C_v = \{u \in U : (u, v) \in E_+\} \cup \{v\}$ ▷ un-clustered vertices similar to v
  update $U \leftarrow U \setminus C_v$
end while
return: $\mathcal{C} = \{C_v \ : v \in C\}$

▶ **Theorem 15** ([2]). *ACNAlg(V, E_+, E_-) is a 3 approximation for* Correlation-Clustering.

In what follows, we outline the proof in [2] of ACNAlg, and describe a couple of definitions and lemmas which will be useful in understanding our overall analysis.

▶ **Definition 16.** *A bad triangle $(u, v, w) \in \mathcal{B}$ is said to be* touched*, denoted by* touched$(t) = 1$*, if there exists a point in the algorithm execution when all three vertices u, v, w belong to the un-clustered set U and one of u, v, w gets sampled as a cluster center.*

▶ **Lemma 17.** *At the end of Algorithm 1, every mis-classified edge (i.e., an E_- edge which is in a single cluster, or an E_+ edge which goes across clusters) is associated with a unique bad triangle which is touched. Moreover, the opposite vertex to the mis-classified edge must be sampled as the cluster center.*

Proof. Consider a stage of the algorithm when a vertex u gets chosen as a cluster center. Then any newly mis-classified edge (v, w) can be of two types: (i) $(v, w) \in E_-$ is mis-classified due to both (u, v) and (u, w) belonging to E_+; (ii) (v, w) belonging to E_+, with $(u, v) \in E_+$ and $(u, w) \in E_-$. In both cases we can associate the newly mis-classified edge (v, w) with the unique bad triangle (u, v, w) which gets touched. ◀

Proof of Theorem 15. The first step is the following LP-based lower bound on Opt$(\mathcal{I})$. Indeed, we know that each bad triangle must have at least one mis-classified edge, and so the LP is simply a linear relaxation for finding a maximal set of disjoint bad triangles.

$$\begin{aligned} \text{maximize} \quad & \sum_{t \in \mathcal{B}} w_t, && s.t., \\ & \sum_{t \in \mathcal{B}: u,v \in t} w_t \leq 1, && \forall e = (u, v) \in E, \\ & w_t \in [0, 1], && \forall t \in \mathcal{B}. \end{aligned} \tag{LP1}$$

Since it will be useful in the next section, we state the dual program, which is a relaxation for the hitting set for all bad triangles.

$$\begin{aligned} \text{minimize} \quad & \sum_{u,v} z_{u,v}, && s.t., \\ z_{u,v} + z_{v,w} + z_{u,w} &\geq 1, && \forall t \in \mathcal{B}, \\ z_{u,v} &\in [0,1], && \forall u, v \in \mathcal{B}. \end{aligned} \tag{LP2}$$

Now, let $p_t = \mathbb{E}[\mathsf{touched}(t)]$, where $\mathsf{touched}(t)$ is the indicator random variable for whether a bad triangle t is touched in the algorithm. The crux of the proof is the following lemma.

▶ **Lemma 18.** *The values* $\{\mathbb{E}[\mathsf{touched}(t)]/3 : t \in \mathcal{B}\}$ *form a feasible solution to LP1.*

Proof. To this end, consider any edge $e = (u, v)$ and the set of bad triangles $\mathcal{B}_{u,v} = \{(u, v, w) \in \mathcal{B}\}$ it is part of. Lemma 17 tells us that (u, v) will be mis-classified if and only if one of these bad triangles $t \equiv (u, v, w) \in \mathcal{B}_{u,v}$ is touched, and the third vertex w must be picked as a cluster center when the triangle is touched. Finally note that, for any triangle $t \equiv (u, v, w)$, the probability that w is picked as the cluster center conditioned on $\mathsf{touched}(t)$ is exactly $1/3$, since the algorithm selects the new cluster center uniformly at random from the un-clustered vertices. Thus we have that: $1 \geq \mathbb{P}((u, v) \text{ is mis-classified}) = \sum_{t \in \mathcal{B}_{u,v}} p_t/3$, thereby showing the LP feasibility of $\{p_t/3\}$. ◀

Also note that by Lemma 17, we have that $\mathbb{E}[\mathsf{cost}(\mathcal{C})] = \sum_{t \in \mathcal{B}} p_t$, where $\mathsf{cost}(\mathcal{C})$ is the objective value of the clustering $\mathcal{C}$. Lemma 18 coupled with this inequality bounding the cost completes the proof of Theorem 15. ◀

4.2 LP-rounding algorithm for Robust-Correlation-Clustering

We now present our constant-factor bi-criteria approximation for Robust-Correlation-Clustering which uses ACNAlg as a sub-routine. Since the ACNAlg algorithm analysis bounds the expected cost of the clustering in terms of the LP relaxation LP1, by duality, we can also infer that the expected cost of ACNAlg is bounded by the LP relaxation LP2. We use this intuition as our starting point: indeed, we can extend this covering LP to handle outliers in the following natural manner. Let $z_{u,v}$ denote whether an edge (u, v) is mis-classified, and y_u denote whether a vertex is deleted or not. Then the following LP3 is a valid LP relaxation for Robust-Correlation-Clustering on complete graphs.

$$\begin{aligned} \text{minimize} \quad & \sum_{(u,v) \in \binom{V}{2}} z_{u,v}, && s.t. \\ y_u + y_v + y_w + z_{u,v} + z_{v,w} + z_{u,w} &\geq 1, && \forall t = (u, v, w) \in \mathcal{B}, \qquad (2) \\ \sum_u y_u &\leq m, \\ z_{u,v} &\geq 0, && \forall (u, v) \in \binom{V}{2}, \\ y_u &\geq 0, && \forall u \in V. \end{aligned} \tag{LP3}$$

Equation (2) of LP3 states that at least a unit cost is incurred for any bad triangle in $\mathcal{B}$ if no vertices from this triangle are deleted. Let $\{y^*_u : u \in V\}, \left\{z^*_{u,v} : (u, v) \in \binom{V}{2}\right\}$ denote the optimal solution to LP3.

▶ **Lemma 19.** $\mathsf{Opt}(\mathcal{I}) \geq \sum_{(u,v) \in \binom{V}{2}} z^*_{u,v} = \mathsf{Opt}(LP3)$.

Proof. Indeed, consider any optimal solution to the Robust-Correlation-Clustering instance, and set $z_{u,v} = 1$ if (u, v) is mis-classified, and $y_u = 1$ if u is deleted. For any bad triangle $(u, v, w) \in \mathcal{B}$, note that either one of u, v or w must be deleted as an outlier in the optimal solution, or one of the three edges must be mis-classified. Hence the first LP constraint is satisfied. The second is true since the optimal solution deletes at most m outliers. Finally, the objective function captures the number of mis-classified edges. ◀

Algorithm 2 RCCAlg(V, E_+, E_-, m).

Initialization: $V_{\text{del}} \leftarrow \emptyset$ ▷ Set of deleted vertices
Let the optimal solution of LP3 be denoted as $\{y^*_u : u \in V\} \cup \{z^*_{uv} : (u, v) \in \binom{V}{2}\}$
$V_{\text{del}} \leftarrow \{v \in V : y^*_v \geq 1/6\}$ ▷ Delete vertices having $y^*_v \geq 1/6$
$V' \leftarrow V \setminus V_{\text{del}}$
return: ACNAlg(V', E'_+, E'_-) ▷ E'_+, E'_-: edges in $\binom{V}{2}$ not incident on V_{del}

4.3 Analysis

▶ **Theorem 20.** *RCCAlg(V, E_+, E_-, m) is a bi-criteria $(6, 6)$-approximation for* Robust-Correlation-Clustering.

Proof. The proof of this result follows from Lemmas 21 and 22. ◀

▶ **Lemma 21.** *At most $6m$ vertices are deleted by RCCAlg(V, E_+, E_-, m).*

Proof. Recall that RCCAlg(V, E_+, E_-, m) deletes those vertices having $y^*_u \geq 1/6$ in the optimal solution to LP3. Let the set of vertices deleted by RCCAlg(V, E_+, E_-, m) be denoted V_{del}. Then,

$$|V_{\text{del}}| = \sum_{u \in V} \mathbb{1}(y^*_u \geq 1/6) \leq \sum_{u \in V} 6y^*_u \leq 6m.$$

Therefore, the budget of vertices to remove is not exceeded by more than a factor of 6. ◀

We next bound the cost incurred by the clustering output by RCCAlg(V, E_+, E_-, m).

▶ **Lemma 22.** *The cost of the clustering output by RCCAlg(V, E_+, E_-, m) is at most 6 times the cost of the optimal clustering to $\mathcal{I}$.*

Proof. Since the first step deletes vertices in $V_{\text{del}} = \{v \in V : y^*_v \geq 1/6\}$, it suffices to consider the remaining vertices $V' = V \setminus V_{\text{del}}$ and show that ACNAlg has cost at most 6Opt on the residual instance. The proof is again very simple: indeed, each vertex $v' \in V'$ has $y^*_{v'} \leq 1/6$, we get that the optimal LP solution to LP3 satisfies $z^*_{u,v} + z^*_{v,w} + z^*_{u,w} \geq 1/2$ for all $(u, v, w) \in \mathcal{B}'$, where $\mathcal{B}'$ denotes the set of all bad triangles induced in the vertex set V'. Then by simply considering the scaled variables $2z^*_{u,v}$, we get that there exists a feasible solution to LP2 for the Correlation-Clustering instance induced in (V', E'_+, E'_-), of cost at most 2Opt. Hence, since the 3-approximation of ACNAlg guarantee holds against the dual LP LP1, we can use weak duality to complete the proof. ◀

5 Algorithms for Robust-Correlation-Clustering on General Graphs

In this section, we consider Robust-Correlation-Clustering on general graphs and prove Theorem 8. Given an instance $\mathcal{I}$, comprising of graph $G = (V, E_+ \cup E_-)$ and outlier budget m, we begin with the following LP relaxation:

$$\text{Minimize} \quad \sum_{(u,v)\in E_+\cup E_-} z_{u,v}, \qquad s.t., \tag{LP6}$$

$$x_{u,v} + x_{v,w} \geq x_{u,w}, \qquad \forall u \neq v \neq w \tag{3}$$

$$y_u + y_v + z_{u,v} \geq 1 - x_{u,v}, \qquad \forall (u,v) \in E_- \tag{4}$$

$$y_u + y_v + z_{u,v} \geq x_{u,v}, \qquad \forall (u,v) \in E_+ \tag{5}$$

$$\sum_u y_u \leq m, \tag{6}$$

$$x_{u,v}, z_{u,v}, y_u \in [0,1]$$

In simple terms, on imposing integer constraints, LP6 asks to find a clustering s.t. $x_{u,v} = 1$ if u and v belong to different clusters, and 0 otherwise. It is easy to check that such an assignment of $x_{u,v}$ satisfies the triangle inequality constraint Equation (3). The objective function charges a unit cost ($z_{u,v} = 1$) for dissimilar (resp. similar) pairs of points (u, v) placed in the same (resp. different) clusters, only if neither u nor v is deleted, i.e, if $y_u = y_v = 0$. In addition, Equation (6) ensures that at most m vertices are deleted in the intended solution. The following lemma is then an immediate consequence of the fact that the optimal integral solution to Robust-Correlation-Clustering instance $\mathcal{I}$ is feasible for Equation (LP6).

▶ **Lemma 23.** *The optimal solution $\{x^*, y^*, z^*\}$ to the LP above has objective value at most* $\mathsf{Opt}(\mathcal{I})$*, the cost of an optimal* Robust-Correlation-Clustering *solution. Moreover, we may slightly perturb this solution to ensure that (a) $\min_{(u,v):x^*_{u,v}\neq 0} x^*_{u,v} \geq 1/n^2$ and $\min_{u:y^*_u\neq 0} y^*_u \geq 1/n^2$, i.e., the smallest non-zero values among x^* and y^* variables is at least $1/n^2$, and (b) the perturbed solution has same objective value and satisfies all the LP inequalities except Equation* (6)*, which is satisfied up to $\sum_u y^*_u \leq (m + 1/n)$.*

We require the lower bound on the x^* and y^* variables for technical reasons which will become clear as the proof proceeds. However, for all practical purposes, the reader may assume that it is just the optimal solution to the LP. We begin by observing that the one of the techniques of solving the Correlation-Clustering problem is by reducing it to Minimum-Multicut problem (in fact, up to constant factors, the Correlation-Clustering problem on general graphs is *equivalent* to Minimum-Multicut on general graphs in [14]), and running the best known approximation to Minimum-Multicut to get $O(\log n)$ approximations to Correlation-Clustering. In our case, for Robust-Correlation-Clustering, just like how we used a specific approximation algorithm ACNAlg for Correlation-Clustering, it turns out that the right starting point for general graphs is the following beautiful partitioning scheme (Theorem 24) for metric spaces known as *padded decompositions*. At a high level, they randomly *partition* a metric space into regions of bounded diameter, such that the probability of a *ball of radius ρ around any vertex v* being separated by the partitioning is proportional to ρ. This generalizes the standard partitioning schemes which just guarantee that the probability that any pair u, v being separated is proportional to $d(u, v)$. While any scheme which satisfies the latter suffices to get good algorithms for Correlation-Clustering, we crucially use the stronger property in our algorithm for Robust-Correlation-Clustering.

▶ **Theorem 24** ([15]). *For any finite metric space (X, d) and parameter $\Delta > 0$, there exists a randomized algorithm* PaddedClustering(X, d, Δ) *which outputs a clustering $\mathcal{C}$ of points in X such that,*

- *Every cluster $C \in \mathcal{C}$ has diameter at most Δ,*
- *For every $x \in X$ and $\rho \in (0, \Delta/8)$,*

$$\text{Prob}(\text{Ball}_\rho(x) \not\subseteq \mathcal{C}(x)) \leq \alpha(x)\frac{\rho}{\Delta}, \tag{7}$$

where $\alpha(x) = \mathcal{O}(\log(\frac{|\text{Ball}_\Delta(x)|}{|\text{Ball}_{\Delta/8}(x)|})) = O(\log n)$ and $\mathcal{C}(x)$ denotes the points in the same cluster as x in $\mathcal{C}$.

5.1 Rounding Algorithm

Before we describe the algorithm in detail, we now provide an overview.

Step 1. We first compute a near-optimal solution $\{x^*, y^*, z^*\}$ for Equation (LP6) satisfying the conditions of Lemma 23.

Step 2. We run the padded decomposition scheme on x^* with $\Delta = 0.25$ to obtain a clustering $\mathcal{C}^*$ of the points. Indeed, we can interpret $\mathcal{C}^*$ as a *rounding* of the $x_{u,v}$ variables into an integral clustering: if $x^*_{u,v} \geq 0.25$, then u and v are definitely in different clusters of $\mathcal{C}^*$, and if $x^*_{u,v}$ is small, then the are in different clusters with probability $\propto O(\log n)x^*_{u,v}$.

Step 3. If a mis-classified edge in this clustering has $z^*_{u,v}$ at least some constant, say 0.25, then we can charge such edges to the LP objective.

Step 4a. It remains to consider mis-classified edges with small $z^*_{u,v}$. If $(u, v) \in E_-$, then again this is an easy case, since we know that $x^*_{u,v} \leq 0.25$ because (u, v) is mis-classified, hence it must belong to the same cluster, and all clusters have diameter at most 0.25 w.r.t the x^* metric. Hence, if $z^*_{u,v} \leq 0.25$ for such edges, we can infer that $y^*_u + y^*_v \geq 0.5$ from Equation (4), and we can handle all such edges by *deleting all vertices* with $y^*_u \geq 0.25$.

Step 4b. We are finally left with handling the case when $(u, v) \in E_+$, and $z^*_{u,v}$ is small. Here again, we are in good shape if $x^*_{u,v}$ is at least some constant, since from Equation (5) we know that at least one of y^*_u or y^*_v or $z^*_{u,v}$ must be large, so we can either delete an end-point of (u, v), or we can charge this mis-classified edge to the LP objective. On the other hand, if $x^*_{u,v}$ is small and (u, v) is mis-classified (and so u and v belong to different clusters since $(u, v) \in E_+$), we use the padded decomposition property that such an event occurred with very low probability, and we can actually afford to the scale variables by $x^*_{u,v}$ to get that $\frac{y^*_u}{x^*_{u,v}} + \frac{y^*_v}{x^*_{u,v}} + \frac{z^*_{u,v}}{x^*_{u,v}} \geq 1$. In expectation, the overall scaling factor would be bounded from Theorem 24, and moreover, for each mis-classified edge in E_+, we can either charge it to the scaled $z^*_{u,v}$ variable, or delete an end-point due to the scaled y^*_u or y^*_v being large. Of course, this is a simplified view since we cannot consider different scaling factors for different edges. In our actual algorithm, we scale each y^*_v by a quantity r_v, where r_v is the radius of the *smallest ball around v* w.r.t metric s^* which gets separated by the clustering $\mathcal{C}^*$. This is where our proof uses the stronger properties of the padded decomposition schemes.

▶ **Theorem 25.** *RCC-general(V, E_+, E_-, m) is a randomized $(\mathcal{O}(\log n), \mathcal{O}(\log^2 n))$ bi-criteria approximation for* Robust-Correlation-Clustering *on general graphs.*

Proof. We begin by introducing some notation that will be useful for the analysis of the algorithm. Consider the clustering $\mathcal{C}^*$ output by PaddedClustering$(V, x^*, 0.25)$ in RCC-general(V, E_+, E_-, m). We slightly abuse notation and let $\mathcal{C}^*(v)$ denote the set of all vertices

Algorithm 3 RCC-general(V, E_+, E_-, m).

Let $\{x^*, y^*, z^*\}$ denote the (perturbed) optimal solution to LP6 obtained in Lemma 23
Compute $\mathcal{C}^* = \mathsf{PaddedClustering}(V, x^*, 0.25)$
Define $V_b^- = \{v \in V : \exists u \in \mathcal{C}^*(v) \text{ such that } (u,v) \in E_-\}$ ▷ candidate vertices for deletion: have a $-$ edge to at least one other vertex in the same cluster
Define $V_{\text{del}}^- = \{v \in V_b^- \ : \ y_v^* \geq 1/4\}$
Set $V' \leftarrow V \setminus V_{\text{del}}^-$
Define $V_b^+ = \{v \in V' : \exists u \in V' \setminus \mathcal{C}^*(v) \text{ such that } (u,v) \in E_+\}$ ▷ candidate vertices for deletion: have a $+$ edge to at least one vertex in a different cluster
For each $u \in V_b^+$, define

$$\hat{y}_u \stackrel{\text{def}}{=} 2^r \cdot y_u^*, \quad \text{where } \frac{1}{2^r} < \min_{v \in V' \setminus \mathcal{C}^*(u)} x_{u,v}^* \leq \frac{1}{2^{r-1}}$$

Define $V_{\text{del}}^+ = \{v \in V_b^+ : \hat{y}_v \geq 1/3\}$
Return: $D_{\mathsf{alg}} = V_{\text{del}}^- \cup V_{\text{del}}^+$ as outliers and the clustering $\mathcal{C}_{\mathsf{alg}} = \mathcal{C}^* \setminus D$

which are in the same cluster as v in the clustering $\mathcal{C}^*$. Define E_b^- as the set of $-$ edges between vertices in V in the same cluster in $\mathcal{C}^*$, $E_b^- \stackrel{\text{def}}{=} \{(u,v) \in E_- \ : \ u \in \mathcal{C}^*(v)\}$. In addition, define E_b^+ to be the set of $+$ edges between vertices in V' lying in different clusters in $\mathcal{C}^*$, i.e., $E_b^+ \stackrel{\text{def}}{=} \{(u,v) \in E_+ \ : \ u \in V' \setminus \mathcal{C}^*(v)\}$. Let $\mathsf{cost}(\mathsf{alg})$ denote the cost of the clustering output by RCC-general(V, E_+, E_-, m) and let $V_{\text{del}} = V_{\text{del}}^- \cup V_{\text{del}}^+$ denote the set of vertices deleted. Observe that any edge that contributes to $\mathsf{cost}(\mathsf{alg})$ belongs to either E_b^+ or E_b^- and is not incident on any vertex in V_{del}. Therefore, $\mathsf{cost}(\mathsf{alg})$ can be decomposed as

$$\mathsf{cost}(\mathsf{alg}) \leq \mathsf{cost}(\mathsf{alg})^- + \mathsf{cost}(\mathsf{alg})^+. \tag{8}$$

where $\mathsf{cost}(\mathsf{alg})^-$ denotes the cost associated with edges in E_b^- that are not incident on vertices in V_{del}^-, and $\mathsf{cost}(\mathsf{alg})^+$ denotes the cost associated with edges in E_b^+ that are not incident on vertices in $V_{\text{del}}^- \cup V_{\text{del}}^+$.

Let Opt^* denote the cost of the optimal solution to LP6. To bound the cost of our solution, we show in Lemmas 28 and 33 respectively that $\mathsf{cost}(\mathsf{alg})^-$ is upper-bounded by $4\mathsf{Opt}^*$, while $\mathbb{E}\left[\mathsf{cost}(\mathsf{alg})^+\right]$ is upper-bounded by $O(\log n)\mathsf{Opt}^*$.

On the other hand, to bound the number of vertices deleted by RCC-general(V, E_+, E_-, m), we follow a similar strategy. Since, $|V_{\text{del}}| = |V_{\text{del}}^+| + |V_{\text{del}}^-|$, we separately upper bound V_{del}^- and $\mathbb{E}[V_{\text{del}}^+]$ in Lemmas 27 and 32 by $4m$ and $\mathcal{O}(\log^2 n)m$ respectively. ◀

Recall that the optimal solution of LP6 is denoted as $\left(\{y_u^*\}, \{x_{u,v}^*\}, \{z_{u,v}^*\}\right)$. We begin by establishing some basic properties of the clustering $\mathcal{C}^*$.

▷ Claim 26. For any edge $(u,v) \in E_b^-$, $y_u^* + y_v^* + z_{u,v}^* \geq 0.75$.

Proof. Recall that E_b^- denotes the set of dissimilar points in V that are placed in the same cluster by $\mathcal{C}^*$. Since, $E_b^- \subseteq E_-$, the optimal solution to LP6 must satisfy the negative edge-constraint (4) for edge (u,v), and so $y_u^* + y_v^* + z_{u,v}^* \geq 1 - x_{u,v}^*$. Now, note that $x_{u,v}^* \leq 0.25$, since u and v belong to the same cluster in $\mathcal{C}^*$ and the diameter of any cluster in $\mathsf{PaddedClustering}(X, d, \Delta)$ is at most Δ from Theorem 24. ◁

▶ **Lemma 27.** *The set of vertices, V_{del}^- satisfies $|V_{\text{del}}^-| \leq 4\sum_{v \in V} y_v^* \leq 4(m + 1/n)$.*

Proof. Recall that V_{del}^- is the set of vertices, $v \in V_b^-$ such that $y_v^* \geq 1/4$. This, combined with the fact that $\{y_v^*\}$ satisfies $\sum_u y_u^* \leq m + 1/n$ from Lemma 23, completes the proof. ◀

▶ **Lemma 28.** *The cost of mis-classified E_- edges* $\mathsf{cost}(\mathsf{alg})^-$ *is at most* $4\sum_{(u,v)\in E^-} z^*_{u,v}$.

Proof. Observe that $\mathsf{cost}(\mathsf{alg})^-$ accrues unit cost only for edges in E_b^- which are not incident on a vertex in V_{del}^-. This implies that $y^*_u \le 0.25$ for all vertices incident on such edges. This, combined with Claim 26 completes the proof. ◀

We now move onto the analysis of $\mathsf{cost}(\mathsf{alg})^+$ and $|V^+_{\text{del}}|$, which are slightly more involved. In this respect, define

$$\hat{z}_{u,v} \stackrel{\text{def}}{=} \begin{cases} \frac{z^*_{u,v}}{x^*_{u,v}}, & v \notin \mathcal{C}^*(u), \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

We demonstrate some useful facts about $\hat{z}_{u,v}$ and $\hat{y}_u$, which recall is defined previously as,

$$\hat{y}_u = 2^r \cdot y^*_u, \quad \text{where, } r \ : \ \frac{1}{2^r} < \min_{v\in V'\setminus\mathcal{C}^*(u)} x^*_{u,v} \le \frac{1}{2^{r-1}}$$

▷ Claim 29. For any edge $(u,v) \in E_b^+$, $\mathbb{E}[\hat{z}_{u,v}] \le \mathcal{O}(\log n) z^*_{u,v}$.

Proof. Observe that if two points belong to different clusters, then we must necessarily have for $\rho = x^*_{u,v}$ that $\text{Ball}_\rho(u) \not\subseteq \mathcal{C}(u)$). Therefore, from Theorem 24,

$$\text{Prob}(u \notin \mathcal{C}^*(v)) \le \mathcal{O}(\log n)\frac{x^*_{u,v}}{0.25}.$$

Therefore, from the definition of $\hat{z}_{u,v}$ in (9), it follows that, $\mathbb{E}[\hat{z}_{u,v}] \le \mathcal{O}(\log n)\frac{x^*_{u,v}}{0.25}\frac{z^*_{u,v}}{x^*_{u,v}} + 0 = \mathcal{O}(\log n) z^*_{u,v}$. ◁

▷ Claim 30. For any vertex $v \in V_b^-$, $\mathbb{E}\left[\hat{y}_u\right] \le \mathcal{O}(\log^2 n)\cdot y^*_u$.

Proof. Observe that $x^*_{u,v} \in [n^{-2}, 1]$. Therefore, r takes values from the set $\{0, 1, 2, \ldots, 2\log n\}$. By definition of $\hat{y}_u$,

$$\begin{aligned} \mathbb{E}[\hat{y}_u] &= \sum_{r=0}^{2\log n} 2^r \left(y^*_u\right) \text{Prob}\left(\frac{1}{2^r} < \min_{v\in V\setminus\mathcal{C}^*(u)} x^*_{u,v} \le \frac{1}{2^{r-1}}\right), \\ &\le \sum_{r=0}^{2\log n} 2^r \left(y^*_u\right) \text{Prob}\left(\min_{v\in V\setminus\mathcal{C}^*(u)} x^*_{u,v} \le \frac{1}{2^{r-1}}\right). \end{aligned} \tag{10}$$

Next, observe that the event $\min_{v\in V'\setminus\mathcal{C}^*(u)} x^*_{u,v} \le 2^{-(r-1)}$ can only occur if the ball of radius $2^{-(r-1)}$ centered at u does not lie entirely within $\mathcal{C}(u)$. Therefore, from Theorem 24,

$$\text{Prob}\left(\min_{v\in V\setminus\mathcal{C}^*(u)} x^*_{u,v} \le \frac{1}{2^{r-1}}\right) \le \mathcal{O}(\log n)\frac{1}{2^{r-1}}.$$

Plugging this into (10) gives, $\mathbb{E}[\hat{y}_u] \le \mathcal{O}(\log n)\sum_{r=0}^{2\log n} y^*_u = \mathcal{O}(\log^2 n)\cdot y^*_u$. ◁

▷ Claim 31. For any edge $(u,v) \in E_b^+$, we have that $\hat{y}_u + \hat{y}_v + \hat{z}_{u,v} \ge 1$.

Proof. Since $E_b^+ \subseteq E_+$, every $(u,v) \in E_b^+$ must satisfy the positive edge-constraint (5) $y^*_u + y^*_v + z^*_{u,v} \ge x^*_{u,v}$. The proof then concludes by dividing both sides by $x^*_{u,v}$, and using the definitions of $\hat{y}_u$ and $\hat{z}_{u,v}$. ◁

▶ **Lemma 32.** *The set of vertices V^+_{del} satisfies,* $\mathbb{E}\left[\left|V^+_{\text{del}}\right|\right] \leq \mathcal{O}(\log^2 n)\ m$.

Proof. Recall that V^+_{del} is defined as the set of vertices $v \in V^+_b$ such that $\hat{y}_v \geq 1/3$. Therefore $|V^+_{\text{del}}| = \sum_{v \in V^+_b} \mathbb{1}(\hat{y}_v \geq 1/3)$. Since $\mathbb{1}(\hat{y}_v \geq 1/3) \leq 3\hat{y}_v$, it follows that $|V^+_{\text{del}}| \leq 3\sum_{v \in V^+_b} \hat{y}_v$. Taking expectation on both sides, and using Claim 30, $\mathbb{E}[|V^+_{\text{del}}|] \leq \mathcal{O}(\log^2 n) \sum_{v \in V^+_b} y^*_v$. The proof concludes by relaxing the summation $v \in V^+_b$ to $v \in V$, and using Lemma 23 to claim that $\sum_{v \in V} y^*_v \leq m + \frac{1}{n} \leq 2m$. ◀

▶ **Lemma 33.** *The expected cost of the mis-classified E_+ edges $\mathbb{E}\left[\mathsf{cost}(\mathsf{alg})^+\right]$ is at most $\mathcal{O}(\log n) \sum_{(u,v) \in E^+} z^*_{u,v}$.*

Proof. $\mathsf{cost}(\mathsf{alg})^+$ is the cost corresponding to edges in E^+_b which are not incident on any vertex in V_{del}. Recall that a vertex $v \in V'$ belongs to V_{del} only if $\hat{y}_v \geq 1/3$. Following a similar proof as Lemma 28, we get that,

$$\mathsf{cost}(\mathsf{alg})^+ \leq \sum_{(u,v) \in E^+_b} \mathbb{1}(\hat{z}_{u,v} \geq 1/3) \leq 3 \sum_{(u,v) \in E^+_b} \hat{z}_{u,v},$$

Taking expectations on both sides, using Claim 29 to upper bound $\mathbb{E}[\hat{z}_{u,v}]$ by $\mathcal{O}(\log n) z^*_{u,v}$, and relaxing the summation to $(u,v) \in E_+$ completes the proof. ◀

References

1. KookJin Ahn, Graham Cormode, Sudipto Guha, Andrew McGregor, and Anthony Wirth. Correlation Clustering in Data Streams. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2237–2246, Lille, France, 2015. PMLR. URL: `http://proceedings.mlr.press/v37/ahn15.html`.
2. Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating Inconsistent Information: Ranking and Clustering. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 684–693, New York, NY, USA, 2005. ACM. `doi:10.1145/1060590.1060692`.
3. Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation Clustering. *Mach. Learn.*, 56(1-3):89–113, June 2004. `doi:10.1023/B:MACH.0000033116.57574.95`.
4. Amir Ben-Dor and Zohar Yakhini. Clustering Gene Expression Patterns. In *Proceedings of the Third Annual International Conference on Computational Molecular Biology*, RECOMB '99, pages 33–42, New York, NY, USA, 1999. ACM. `doi:10.1145/299432.299448`.
5. Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with Qualitative Information. *J. Comput. Syst. Sci.*, 71(3):360–383, October 2005. `doi:10.1016/j.jcss.2004.10.012`.
6. Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for Facility Location Problems with Outliers. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, pages 642–651, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics. URL: `http://dl.acm.org/citation.cfm?id=365411.365555`.
7. Sanjay Chawla and Aristides Gionis. k-means-: A Unified Approach to Clustering and Outlier Detection. In *SDM*, pages 189–197. SIAM, 2013. URL: `http://dblp.uni-trier.de/db/conf/sdm/sdm2013.html#ChawlaG13`.
8. Shuchi Chawla, Konstantin Makarychev, Tselil Schramm, and Grigory Yaroslavtsev. Near Optimal LP Rounding Algorithm for CorrelationClustering on Complete and Complete K-partite Graphs. In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 219–228, New York, NY, USA, 2015. ACM. `doi:10.1145/2746539.2746604`.

9 Jiecao Chen, Erfan Sadeqi Azer, and Qin Zhang. A Practical Algorithm for Distributed Clustering and Outlier Detection. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 2253–2262, 2018. URL: http://papers.nips.cc/paper/7493-a-practical-algorithm-for-distributed-clustering-and-outlier-detection.

10 Ke Chen. A Constant Factor Approximation Algorithm for K-median Clustering with Outliers. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pages 826–835, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1347082.1347173.

11 Flavio Chierichetti, Nilesh Dalvi, and Ravi Kumar. Correlation Clustering in MapReduce. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 641–650, New York, NY, USA, 2014. ACM. doi:10.1145/2623330.2623743.

12 William Cohen and Jacob Richman. Learning to Match and Cluster Entity Names. In *In ACM SIGIR-2001 Workshop on Mathematical/Formal Methods in Information Retrieval*, 2001.

13 William W. Cohen and Jacob Richman. Learning to Match and Cluster Large High-dimensional Data Sets for Data Integration. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 475–480, New York, NY, USA, 2002. ACM. doi:10.1145/775047.775116.

14 Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2):172–187, 2006. Approximation and Online Algorithms. doi:10.1016/j.tcs.2006.05.008.

15 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. Approximating Metrics by Tree Metrics. *SIGACT News*, 35(2):60–70, June 2004. doi:10.1145/992287.992300.

16 Shalmoli Gupta, Ravi Kumar, Kefu Lu, Benjamin Moseley, and Sergei Vassilvitskii. Local Search Methods for k-Means with Outliers. *PVLDB*, 10(7):757–768, 2017. doi:10.14778/3067421.3067425.

17 Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant Approximation for K-median and K-means with Outliers via Iterative Rounding. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 646–659, New York, NY, USA, 2018. ACM. doi:10.1145/3188745.3188882.

18 Shi Li and Xiangyu Guo. Distributed k-Clustering for Data with Heavy Noise. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 7849–7857, 2018. URL: http://papers.nips.cc/paper/8009-distributed-k-clustering-for-data-with-heavy-noise.

19 Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Correlation Clustering with Noisy Partial Information. In Peter Grünwald, Elad Hazan, and Satyen Kale, editors, *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pages 1321–1342, Paris, France, 2015. PMLR. URL: http://proceedings.mlr.press/v40/Makarychev15.html.

20 Andrew McCallum and Ben Wellner. Toward Conditional Models of Identity Uncertainty with Application to Proper Noun Coreference. In *Proceedings of the 2003 International Conference on Information Integration on the Web*, IIWEB'03, pages 79–84. AAAI Press, 2003. URL: http://dl.acm.org/citation.cfm?id=3104278.3104294.

21 Napat Rujeerapaiboon, Kilian Schindler, Daniel Kuhn, and Wolfram Wiesemann. Size Matters: Cardinality-Constrained Clustering and Outlier Detection via Conic Optimization. *SIAM Journal on Optimization*, 2019.

22 Chaitanya Swamy. Correlation Clustering: Maximizing Agreements via Semidefinite Programming. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, volume 15, pages 526–527, January 2004.

23 Anthony Wirth. Correlation Clustering. In *Encyclopedia of Machine Learning*, pages 227–231. Springer, 2010. doi:10.1007/978-0-387-30164-8_176.

A Hardness of Robust-Correlation-Clustering on General Graphs

Firstly, when $m = 0$, Robust-Correlation-Clustering is simply Correlation-Clustering, for which is known NP-hardness of $\Omega(\alpha_{\text{MC}})$ [5]. We show that it is NP-hard to get any (a, b)-approximation for Robust-Correlation-Clustering with finite b when $a < \alpha_{\text{MC}}$, for any $m > 0$.

▶ **Theorem 34.** *It is NP-hard to have an (a, b) bi-criteria approximation to* Robust-Correlation-Clustering *for any finite b and $a < \alpha_{\text{MC}}$.*

Proof. The proof is via a reduction from Minimum-Multicut, similar to the proof for Correlation-Clustering in [5]. Consider the Minimum-Multicut instance problem $\mathcal{I} = \{G(V, E), \{(s_i, t_i), 1 \leq i \leq k\}\}$, where $(s_i, t_i), 1 \leq i \leq k$ represent k source-sink pairs. We construct the Robust-Correlation-Clustering problem instance $\mathcal{I}^*$ as follows. The edges in G become + edges in $\mathcal{I}^*$. For each $i, 1 \leq i \leq k$, we add a negative edge between (s_i, t_i) of weight $-W$, for some large positive integer W, say $W = n^3$. We can make the instance unweighted by replacing a negative edge of weight $-W$ by W parallel length two paths; each path has a fresh intermediate vertex, with one + edge and one − edge. Clearly, the minimum cost clustering must have (s_i, t_i) in different clusters $\forall 1 \leq i \leq k$. In addition, introduce m more vertices which act like outliers, represented by set $U = \{u_1, u_2, \ldots, u_m\}$ in $\mathcal{I}^*$. Connect each $u_i, 1 \leq i \leq m$ to every vertex $q, q \in V(\mathcal{I}^*) \setminus U$ with an edge of weight $-W$ and an edge of weight W. We can make the instance unweighted by replacing the negative edge as described before, and the positive edge of weight W by W parallel length two paths; each path has a fresh intermediate vertex, with both edges +.

Due to the above construction, the vertices $(q, u_i), q \in V(\mathcal{I}^*) \setminus U, 1 \leq i \leq k$ add a high cost irrespective of whether they lie in the same cluster or not.

Hence, the optimal solution to Robust-Correlation-Clustering on the problem instance $\mathcal{I}^*$ removes vertices $u_1, u_2, \cdots, u_m$, and the corresponding optimal cost is same as the Minimum-Multicut optimal cost on instance $\mathcal{I}$. ◀

We next establish that unless the budget of vertices to be removed is violated by a certain factor, it is NP-hard to find any approximation to the cost of the optimal solution to Robust-Correlation-Clustering.

▶ **Theorem 35.** *It is NP-hard to find an (a, b) bi-criteria approximation to* Robust-Correlation-Clustering *for any finite a, and $b < \alpha_{\text{MC}}$.*

Proof. The proof of this result once again follows via a reduction from Minimum-Multicut. Indeed, consider the Minimum-Multicut instance problem $\mathcal{I} = \{G(V, E), \{(s_i, t_i), 1 \leq i \leq k\}\}$, where $(s_i, t_i), 1 \leq i \leq k$ represent k source-sink pairs. We now define an intermediate problem which will simplify our overall reduction. ◀

▶ **Definition 36** (Vertex-Multicut)**.** *Given a problem instance $\mathcal{I} = \{H, \{(s_i, t_i), 1 \leq i \leq k\}\}$, where $(s_i, t_i), 1 \leq i \leq k$ represent k source-sink pairs, the* Vertex-Multicut *problem is to find the minimum set of vertices $S \subseteq V(H)$ such that no source-sink pair lie in the same connected component in the graph induced on $V(H) \setminus S$.*

▶ **Lemma 37.** *There exists an approximation preserving reduction from* Minimum-Multicut *to* Vertex-Multicut*.*

Proof. The idea is to reduce the MINIMUM-MULTICUT problem instance $\mathcal{I}$ to a VERTEX-MULTICUT problem instance $\mathcal{I}' = \{H(V', E'),\ \{(s'_i, t'_i),\ 1 \leq i \leq l\}\}$. Consider the graph $G = (V, E)$ as defined above. Reduce each vertex $v_i \in V$ into a clique of large size, say n, where $n = |V|$. Let clique$(v_i) = \{v_{i1}, v_{i2}, \ldots, v_{in}\}$, where $v_i \in V, 1 \leq i \leq n$ represent the clique in H. For every $(s_i, t_i),\ 1 \leq i \leq k$ source-sink pair in $\mathcal{I}$, let each of $(s_{ia}, t_{ib})\ \forall 1 \leq a, b \leq n$ be a source sink pair in instance $\mathcal{I}'$. Hence, instance $\mathcal{I}'$ will contain kn^2 source-sink pairs in comparison with the k pairs in $\mathcal{I}$. We now define the edges in $\mathcal{I}'$. E' is composed of two components, $\cup_{i \leq n} E_{\text{clique}(v_i)}$ and E_{across}, where $E_{\text{clique}(v_i)} = \{(v_{ia}, v_{ib}),\ 1 \leq i, a, b \leq n, a \neq b\}$, and $E_{\text{across}} = \{(v_{ij}, v_{ji})\ :\ (v_i, v_j) \in E\}$.
We now have a VERTEX-MULTICUT problem instance $\mathcal{I}'$. We claim that the reduction from $\mathcal{I}$ to $\mathcal{I}'$ is an approximation preserving reduction. Let S denote the optimal solution to problem instance $\mathcal{I}'$, that is, S denotes the optimal set of vertices to remove to disconnect the source-sink pairs. Let $v_{ij} \in S,\ 1 \leq i, j \leq n$. Removing the edge $(v_i, v_j) \in E$ in instance $\mathcal{I}$ is equivalent to removing the vertex v_{ij} (or v_{ji}) in $\mathcal{I}'$ where $(u_i, v_j) \in E'$. Hence solving the VERTEX-MULTICUT problem solves MINIMUM-MULTICUT problem as well. ◀

▶ **Lemma 38.** *There exists an approximation preserving reduction from* VERTEX-MULTICUT *to approximating the budget of number of vertices to remove in* ROBUST-CORRELATION-CLUSTERING *problem.*

Proof. Given a VERTEX-MULTICUT problem instance $\mathcal{I}' = \{H, \{(s_i, t_i) 1 \leq i \leq k, \}\}$, we construct a ROBUST-CORRELATION-CLUSTERING problem instance $\mathcal{I}''$. The edges in H becomes positive edges in $\mathcal{I}''$. In addition, add a negative edge between each (s_i, t_i) pair of weight $-W$, for some large positive integer W, say $W = n^3$. The graph can be made unweighted as discussed in the proof to Theorem 34.

Consider the instance $\mathcal{I}''$. The minimum set of vertices R such that the graph induced on remaining vertices has a 0 cost clustering is identical to the optimal solution to the instance $\mathcal{I}'$. From Lemma 37, it follows that if $\mathcal{I}'$ can be solved optimally, the underlying MINIMUM-MULTICUT problem instance $\mathcal{I}$ can be solved optimally. Therefore from Theorem 34 and Lemma 37, it follows that it is NP-hard to violate the budget of number of vertices to remove by a factor $< \alpha_{\text{MC}}$ such that the cost of the output clustering is a finite approximation to the optimal cost. ◀

Counting Independent Sets and Colorings on Random Regular Bipartite Graphs

Chao Liao
Shanghai Jiao Tong University, China
chao.liao.95@gmail.com

Jiabao Lin
Shanghai University of Finance and Economics, China
lin.jiabao@mail.shufe.edu.cn

Pinyan Lu
Shanghai University of Finance and Economics, China
lu.pinyan@mail.shufe.edu.cn

Zhenyu Mao
Shanghai University of Finance and Economics, China
zhenyu.mao.17@gmail.com

Abstract

We give a fully polynomial-time approximation scheme (FPTAS) to count the number of independent sets on almost every Δ-regular bipartite graph if $\Delta \geq 53$. In the weighted case, for all sufficiently large integers Δ and weight parameters $\lambda = \widetilde{\Omega}\left(\frac{1}{\Delta}\right)$, we also obtain an FPTAS on almost every Δ-regular bipartite graph. Our technique is based on the recent work of Jenssen, Keevash and Perkins (SODA, 2019) and we also apply it to confirm an open question raised there: For all $q \geq 3$ and sufficiently large integers $\Delta = \Delta(q)$, there is an FPTAS to count the number of q-colorings on almost every Δ-regular bipartite graph.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Approximate counting, Polymer model, Hardcore model, Coloring, Random bipartite graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.34

Category RANDOM

Related Version A full version of the paper is available at `https://arxiv.org/abs/1903.07531`.

Funding This work is supported by Innovation Program of Shanghai Municipal Education Commission and the Fundamental Research Funds for the Central Universities.

1 Introduction

Counting independent sets on bipartite graphs (#BIS) plays a significant role in the field of approximate counting. A wide range of counting problems in the study of counting CSPs [14, 6, 15] and spin systems [19, 20, 17, 7], have been proved to be #BIS-equivalent or #BIS-hard under approximation-preserving reductions (AP-reductions) [13]. Despite its great importance, it is still unknown whether #BIS admits a fully polynomial-time approximation scheme (FPTAS) or it is as hard as counting the number of satisfying assignments of Boolean formulas (#SAT) under AP-reduction.

In this paper, we consider the problem of approximating #BIS (and its weighted version) on random regular bipartite graphs. Random regular bipartite graphs frequently appear in the analysis of hardness of counting independent sets [34, 12, 38, 39, 17]. Therefore, understanding the complexity of #BIS on such graphs is potentially useful for gaining


© Chao Liao, Jiabao Lin, Pinyan Lu, and Zhenyu Mao;
licensed under Creative Commons License CC-BY

insights into the general case. Let $Z(G,\lambda) = \sum_{I\in\mathcal{I}(G)} \lambda^{|I|}$ where $\mathcal{I}(G)$ is the set of all independent sets of a graph G and $\lambda > 0$ is the weight parameter. This function also arises in the study of the hardcore model of lattice gas systems in statistical mechanics. Hence we usually call $Z(G,\lambda)$ the partition function of the hardcore model with fugacity λ.

In the case where input graphs are allowed to be nonbipartite, the approximability for counting the number of independent sets (#IS) is well understood. Exploiting the correlation decay properties of $Z(G,\lambda)$, Weitz [41] presented an FPTAS for graphs of maximum degree Δ at fugacity $\lambda < \lambda_c(\Delta) = \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}}$. On the hardness side, Sly [38] proved that, unless NP = RP, there is a constant $\varepsilon = \varepsilon(\Delta)$ that no polynomial-time approximation scheme exists for $Z(G,\lambda)$ on graphs of maximum degree Δ at fugacity $\lambda_c(\Delta) < \lambda < \lambda_c(\Delta) + \varepsilon(\Delta)$. Later, this result was improved at any fugacity $\lambda > \lambda_c(\Delta)$ [39, 16]. In particular, these results state that if $\Delta \leq 5$, there is an FPTAS for #IS on graphs of maximum degree Δ, otherwise there is no efficient approximation algorithm unless NP = RP.

The situation is different on bipartite graphs. No NP-hardness result is known even on graphs with unbounded degree. Surprisingly, Liu and Lu [29] designed an FPTAS for #BIS which only requires one side of the vertex partition to be of maximum degree $\Delta \leq 5$. On the other hand, it is #BIS-hard to approximate $Z(G,\lambda)$ at fugacity $\lambda > \lambda_c(\Delta)$ on biparite graphs of maximum degree $\Delta \geq 3$ [7].

Recently, Helmuth, Perkins, and Regts [25] developed a new approach via the polymer model and gave efficient counting and sampling algorithms for the hardcore model at high fugacity on certain finite regions of the lattice $\mathbb{Z}^d$ and on the torus $(\mathbb{Z}/n\mathbb{Z})^d$. Their approach is based on a long line of work [36, 37, 28, 1, 2, 35]. Shortly after that, Jenssen, Keevash, and Perkins [26] designed an FPTAS for the hardcore model at high fugacity on bipartite expander graphs of bounded degree. And they further extended the result to random Δ-regular bipartite graphs with $\Delta \geq 3$ at fugacity $\lambda > (2e)^{250}$. This is the first efficient algorithm for the hardcore model at fugacity $\lambda > \lambda_c(\Delta)$ on random regular bipartite graphs. A natural question is, can we design FPTAS for lower fugacity and in particular the problem #BIS on random regular bipartite graphs? Indeed, we obtain such results. Let $\mathcal{G}^{\text{bip}}_{n,\Delta}$ denote the set of all Δ-regular bipartite graphs with n vertices on both sides.

▶ **Theorem 1.** *For $\Delta \geq 53$ and fugacity $\lambda \geq 1$, with high probability (tending to 1 as $n \to \infty$) for a graph G chosen uniformly at random from $\mathcal{G}^{\text{bip}}_{n,\Delta}$, there is an FPTAS for the partition function $Z(G,\lambda)$.*

▶ **Theorem 2.** *For all sufficiently large integers Δ and fugacity $\lambda = \widetilde{\Omega}\left(\frac{1}{\Delta}\right)$ [1], with high probability (tending to 1 as $n \to \infty$) for a graph G chosen uniformly at random from $\mathcal{G}^{\text{bip}}_{n,\Delta}$, there is an FPTAS for the partition function $Z(G,\lambda)$.*

For notational convenience, we use the term "on almost every Δ-regular bipartite graph" to denote that a property holds with high probability (tending to 1 as $n \to \infty$) for randomly chosen graphs from $\mathcal{G}^{\text{bip}}_{n,\Delta}$.

Counting proper q-colorings on a graph is another extensively studied problem in the field of approximate counting [27, 4, 5, 10, 23, 22, 33, 9, 24, 18, 11, 31, 21]. In general graphs, if the number of colors q is no more than the maximum degree Δ, there may not be any proper coloring over the graph. Therefore, approximate counting is studied in the range that $q \geq \Delta + 1$. It was conjectured that there is an FPTAS if $q \geq \Delta + 1$, but the current best result is $q \geq \alpha\Delta + 1$ with a constant α slightly below $\frac{11}{6}$ [40, 8]. The conjecture was only confirmed for the special case $\Delta = 3$ [30].

[1] This means that $\lambda \geq (c_1 \log^{c_2} \Delta)/\Delta$ for some constants $c_1, c_2 > 0$.

On bipartite graphs, the situation is quite different. For any $q \geq 2$, we know that there always exist proper q-colorings for every bipartite graph. For any $q \geq 3$, it is shown to be #BIS-hard but unknown to be #BIS-equivalent [13]. Using a technique analogous to that for #BIS, we obtain an FPTAS to count the number of q-colorings on random Δ-regular bipartite graphs for all sufficiently large integers $\Delta = \Delta(q)$ for any $q \geq 3$.

▶ **Theorem 3.** *For $q \geq 3$ and $\Delta \geq 100\overline{q}^{10}$ where $\overline{q} = \lceil q/2 \rceil$, with high probability (tending to 1 as $n \to \infty$) for a graph chosen uniformly at random from $\mathcal{G}^{\text{bip}}_{n,\Delta}$, there is an FPTAS to count the number of q-colorings.*

This result confirms a conjecture in [26].

Our Technique

The classical approach to designing approximate counting algorithms is random sampling via Markov chain Monte Carlo (MCMC). However, it is known that the Markov chains are slowly mixing on random bipartite graphs for both independent set and coloring if the degree Δ is not too small. Taking #BIS as an example, a typical independent set of a random regular bipartite graph of degree at least 6 is unbalanced: it either chooses most of its vertices from the left side or the right side. Thus, starting from an independent set with most vertices from the left side, a Markov chain is unlikely to reach an independent set with most of its vertices from the right side in polynomial time.

Even so, a recent beautiful work exactly makes use of the above separating property to design approximate counting algorithms [26]. By making the fugacity $\lambda > (2e)^{250}$ sufficiently large, they proved that largest contribution to the partition function comes from extremely unbalanced independent sets, those which occupy almost no vertices on one side and almost all vertices on the other side. In particular, for a bipartite graph $G = (\mathcal{L}, \mathcal{R}, E)$ with n vertices on both sides, they identified two independent sets $I = \mathcal{L}$ and $I = \mathcal{R}$ as ground states as they have the largest weight λ^n among all the independent sets. They proved that one only needs to sum up the weights of states which are close to one of the ground states, for no state is close to both ground states and the contribution from the states which are far away from both ground states is exponentially small.

However, the ground state idea cannot be directly applied to counting independent sets and counting colorings since each valid configuration is of the same weight. We extend the idea of ground states to ground clusters, which is not a single configuration but a family of configurations. For example, we identify two ground clusters for independent sets, those which are entirely chosen from vertices on the left side and those which are entirely chosen entirely from vertices on the right side. If a set of vertices is entirely chosen from vertices on one side, it is obviously an independent set. Thus each cluster contains 2^n different independent sets. Similarly, we want to prove that we can count the configurations which are close to one of the ground clusters and then add them up. For counting colorings, there are multiple ground clusters indexed by a subset of colors $\emptyset \subsetneq X \subsetneq [q]$: colorings which color $\mathcal{L}$ only with colors from X and color $\mathcal{R}$ only with colors from $[q] \setminus X$.

Unlike the ground states in [26], our ground clusters may overlap with each other and some configurations are close to more than one ground cluster. In addition to proving that the number of configurations which are far away from all ground clusters is exponentially small, we also need to prove that the number of double counted configurations is small.

After identifying ground states and with respect to a fixed ground state, Jenssen, Keevash, and Perkins [26] defined a polymer model representing deviations from the ground state and rewrote the original partition function as a polymer partition function. We follow this

idea and define a polymer model representing deviations from a ground cluster. However, deviation from a ground cluster is much subtler than deviation from a single ground state. For example, if we define polymer as connected components from the deviated vertices in the graph, we cannot recover the original partition function from the polymer partition function. We overcome this by defining polymer as connected components in the graph G^2, where an edge of G^2 corresponds to a path of length at most 2 in the original graph. Here, a compatible set of polymers also corresponds to a family of configurations in the original problem, while it corresponds to a single configuration in [26].

It is much more common in counting problems that most contribution is from a neighborhood of some clusters rather than a few isolated states. So, we believe that our development of the technique makes it suitable for a much broader family of problems.

Organization of the paper

In this 10-page version, we only prove Theorem 1 which already explains the key technique for proving Theorem 2 and Theorem 3. The complete proof (and the modifications necessary) for these two can be found in the full version. In Section 2 we review necessary definitions and facts. In Section 3 we prove Theorem 1, where the proof is divided into four parts. The first part deals with the property of the independent sets on certain graphs. The second part uses the polymer model to approximate the number of independent sets. The third part discusses how to approximate the partition function of the polymer model. The last part puts these things together.

Independent work

Towards the end of this project, we learned that the authors of [26] obtained similar results in their upcoming journal version submission.

2 Preliminaries

2.1 Independent sets and random regular bipartite graphs

All graphs considered in this paper are unweighted, undirected, with no loops but may have multiple edges. Let $G = (V, E)$ be a graph. We use $d_G(u, w)$ to denote the distance between two vertices u, w in the graph G. For $\emptyset \subsetneq U, W \subseteq V$, define $d_G(U, W) = \min_{u\in U, w\in W} d_G(u, w)$. Let $U \subseteq V$ be a nonempty set. Let $N_G(U) = \{v \in V : d_G(\{v\}, U) = 1\}$ to be the neighborhood of U and emphasize that $N_G(U) \cap U = \emptyset$. We use $G[U]$ to denote the induced subgraph of G on U. Let E^2 be the set of unordered pairs (u, v) such that $u \neq v$ and $d_G(u, v) \leq 2$. We define G^2 to be the graph (V, E^2). It is clear that if the maximum degree of G is at most Δ, then the maximum degree of G^2 is at most Δ^2. An independent set of the graph G is a subset $U \subseteq V$ such that $(u, w) \notin E$ for any $u, w \in U$. We use $\mathcal{I}(G)$ to denote the set of all independent sets of G. The weight of an independent set I is $\lambda^{|I|}$ where $\lambda > 0$ is a parameter called fugacity. We use $Z(G, \lambda) = \sum_{I\in\mathcal{I}(G)} \lambda^{|I|}$ to denote the partition function of the graph G. Clearly, $Z(G, 1)$ is the number of indepndent sets of G.

For two positive real numbers a and b, we say a is an ε-relative approximation to b for some $\varepsilon > 0$ if $\exp(-\varepsilon)b \leq a \leq \exp(\varepsilon)b$, or equivalently $\exp(-\varepsilon)a \leq b \leq \exp(\varepsilon)a$. A fully polynomial-time approximation scheme (FPTAS) is an algorithm that for every $\varepsilon > 0$ outputs an ε-relative approximation to $Z(G)$ in time $(|G|/\varepsilon)^C$ for some constant $C > 0$, where $Z(G)$ is some quantity, like the number of independent sets, of graphs G that we would like to compute.

We use $G \sim \mathcal{G}^{\text{bip}}_{n,\Delta}$ to denote sampling a Δ-regular bipartite graph G with n vertices on both sides uniformly at random. We say a Δ-regular bipartite graph $G = (\mathcal{L}, \mathcal{R}, E)$ with n vertices on both sides is an (α, β)-expander if for all subsets $U \subseteq \mathcal{L}$ or $U \subseteq \mathcal{R}$ with $|U| \leq \alpha n$, $|N(U)| \geq \beta|U|$. This property is called the expansion property of G. We use $\mathcal{G}^{\Delta}_{\alpha,\beta}$ to denote the set of all Δ-regular bipartite (α, β)-expanders. It is known that a random regular bipartite graph is an expander with high probability.

2.2 The polymer model

Let G be a graph and Ω be a finite set. A polymer $\gamma = (\overline{\gamma}, \omega_{\overline{\gamma}})$ consists of a support $\overline{\gamma}$ which is a connected subgraph of G and a mapping $\omega_{\overline{\gamma}}$ which assigns to each vertex in $\overline{\gamma}$ some value in Ω. We use $|\overline{\gamma}|$ to denote the number of vertices of $\overline{\gamma}$. There is also a weight function $w(\gamma, \cdot) : \mathbb{C} \to \mathbb{C}$ for each polymer γ. There can be many polymers defined on the graph G and we use $\Gamma^* = \Gamma^*(G)$ to denote the set of all polymers defined on it. However, at the moment we do not give a constructive definition of polymers. Such definitions are presented when they are needed, see Section 3.2. We say two polymers γ_1 and γ_2 are compatible if $d_G(\overline{\gamma_1}, \overline{\gamma_2}) > 1$ and we use $\gamma_1 \sim \gamma_2$ to denote that they are compatible. For a subset $\Gamma \subseteq \Gamma^*$ of polymers, it is compatible if any two different polymers in this set are compatible. We define $\mathcal{S}(\Gamma^*) = \{\Gamma \subseteq \Gamma^* : \Gamma \text{ is compatible}\}$ to be the collection of all compatible subsets of polymers. For $\Gamma \in \mathcal{S}(\Gamma^*)$, we also define $\left|\overline{\Gamma}\right|$ to be the number of vertices of the subgraph $\overline{\Gamma}$ and let $\omega_{\overline{\Gamma}}$ be a mapping which assigns each vertex $v \in \overline{\Gamma}$ the value that $\omega_{\overline{\gamma}}$ assigns to v where γ is the unique polymer whose support contains vertex v. We say (Γ^*, w) is a polymer model defined on the graph G and the partition function of this polymer model is $\Xi(G, z) = \sum_{\Gamma \in \mathcal{S}(\Gamma^*)} \prod_{\gamma \in \Gamma} w(\gamma, z)$, where z is a complex variable and $\prod_{\gamma \in \emptyset} w(\gamma, z) = 1$ by convention. The following theorem states conditions that $\Xi(G, z)$ can be approximated efficiently.

▶ **Theorem 4** ([25], Theorem 2.2). *Fix Δ and let $\mathcal{G}$ be a set of graphs of degree at most Δ. Suppose:*

- *There is a constant C such that for all $G \in \mathcal{G}$, the degree of $\Xi(G, z)$ is at most $C|G|$.*
- *For all $G \in \mathcal{G}$ and $\gamma \in \Gamma^*(G)$, $w(\gamma, z) = a_\gamma z^{|\overline{\gamma}|}$ where $a_\gamma \neq 0$ can be computed in time $\exp(O(|\overline{\gamma}| + \log_2 |G|))$.*
- *For every connected subgraph G' of every $G \in \mathcal{G}$, we can list all polymers $\gamma \in \Gamma^*(G)$ with $\overline{\gamma} = G'$ in time $\exp(O(|G'|))$.*
- *There is a constant $R > 0$ such that for all $G \in \mathcal{G}$ and $z \in \mathbb{C}$ with $|z| < R$, $\Xi(G, z) \neq 0$.*

Then for every z with $|z| < R$, there is an FPTAS for $\Xi(G, z)$ for all $G \in \mathcal{G}$.

The following condition by Kotecký and Preiss (KP-condition) is useful to show that $\Xi(G, z)$ is zero-free in certain regions.

▶ **Lemma 5** ([28]). *Suppose there is a function $a : \Gamma^* \to \mathbb{R}_{>0}$ and for every $\gamma^* \in \Gamma^*$, $\sum_{\gamma : \gamma \not\sim \gamma^*} e^{a(\gamma)} |w(\gamma, z)| \leq a(\gamma^*)$. Then $\Xi(G, z) \neq 0$.*

To verify the KP-condition, usually we need to enumerate polymers and the following lemma is useful to bound the number of enumerated polymers.

▶ **Lemma 6** ([3]). *For any graph $G = (V, E)$ with maximum degree Δ and $v \in V$, the number of connected induced subgraphs of order $k \geq 2$ containing v is at most $(e\Delta)^{k-1}/2$. As a corollary, the number of connected induced subgraphs of order $k \geq 1$ containing v is at most $(e\Delta)^{k-1}$.*

2.3 Some useful lemmas

Throughout this paper, we use $H(x)$ to denote the binary entropy function

$$H(x) = -x\log_2 x - (1-x)\log_2(1-x), \quad x \in (0,1).$$

Moreover, we extend this function to the interval $[0,1]$ by defining $H(0) = H(1) = 0$. This is reasonable since $\lim_{x\to 0^+} H(x) = \lim_{x\to 1^-} H(x) = 0$.

▶ **Lemma 7.** *It holds that $H(x) \le 2\sqrt{x(1-x)} \le 2\sqrt{x}$ for all $0 \le x \le 1$.*

▶ **Lemma 8** ([32, Lemma 10.2])**.** *Suppose that n is a positive integer and $k \in [0,1]$ is a number such that kn is an integer. Then $\frac{2^{H(k)n}}{n+1} \le \binom{n}{kn} \le 2^{H(k)n}$.*

▶ **Lemma 9.** *For $b > a > 0$, the function $f(\lambda) = \lambda^a/(\lambda+1)^b$ is monotonically increasing on $[0, \frac{a}{b-a}]$ and monotonically decreasing on $[\frac{a}{b-a}, +\infty)$.*

3 Counting independent sets for $\lambda \ge 1$

Throughout this section, we consider integers $\Delta \ge 53$, fugacity $\lambda \ge 1$ and set parameters ζ, α, β to be $\zeta = 1.28, \alpha = \frac{2.9}{\Delta}, \beta = \frac{\Delta}{2.9\zeta}$.

▶ **Lemma 10.** *For $\Delta \ge 53$, $\lim_{n\to\infty} \Pr_{G\sim\mathcal{G}^{\mathrm{bip}}_{n,\Delta}}\left[G \in \mathcal{G}^{\Delta}_{\alpha,\beta}\right] = 1$.*

The reader can find the detailed proof of the lemma above in the full version of the paper.

In the rest of this section, whenever possible, we will simplify notations by omitting superscripts, subscripts and brackets with the symbols between (but this will not happen in the statement of lemmas and theorems). For example, $Z(G,\lambda)$ may be written as Z if G and λ are clear from context.

3.1 Approximating $Z(G,\lambda)$

For all $G = (\mathcal{L}, \mathcal{R}, E) \in \mathcal{G}^{\Delta}_{\alpha,\beta}$, $\mathcal{X} \in \{\mathcal{L}, \mathcal{R}\}$ and $\lambda \ge 1$, let $\mathcal{I}_{\mathcal{X}}(G) = \{I \in \mathcal{I}(G) : |I \cap \mathcal{X}| < \alpha n\}$ and $Z_{\mathcal{X}}(G,\lambda) = \sum_{I\in\mathcal{I}_{\mathcal{X}}(G)} \lambda^{|I|}$. The main result in this part is that we can use $Z_{\mathcal{L}}(G,\lambda) + Z_{\mathcal{R}}(G,\lambda)$ to approximate $Z(G,\lambda)$.

▶ **Lemma 11.** *For $\Delta \ge 53$ and $\lambda \ge 1$, there are constants $C = C(\Delta) > 1$ and $N = N(\Delta)$ so that for all $G \in \mathcal{G}^{\Delta}_{\alpha,\beta}$ with $n > N$ vertices on both sides, $Z_{\mathcal{L}}(G,\lambda) + Z_{\mathcal{R}}(G,\lambda)$ is a C^{-n}-relative approximation to $Z(G,\lambda)$.*

Proof. Apply Lemma 12 and Lemma 13. ◀

▶ **Lemma 12.** *For $\Delta \ge 3$ and $\lambda \ge 1$, there are constants $C = C(\Delta) > 1$ and $N = N(\Delta)$ so that for all $G \in \mathcal{G}^{\Delta}_{\alpha,\beta}$ with $n > N$ vertices on both sides, $\sum_{I\in\mathcal{I}_{\mathcal{L}}(G)\cup\mathcal{I}_{\mathcal{R}}(G)} \lambda^{|I|}$ is a C^{-n}-relative approximation to $Z(G,\lambda)$.*

Proof. Let $\mathcal{B} = \mathcal{I} \setminus (\mathcal{I}_{\mathcal{L}} \cup \mathcal{I}_{\mathcal{R}})$. For any $I \in \mathcal{B}$, it follows from the definition of $\mathcal{B}$ that $|I \cap \mathcal{L}| \ge \alpha n$ and $|I \cap \mathcal{R}| \ge \alpha n$. Using the expansion property, we obtain $|N(I \cap \mathcal{L})| \ge \beta\lfloor \alpha n \rfloor$ and thus $|I \cap \mathcal{R}| \le n - |N(I \cap \mathcal{L})| \le (1 - 1/t)n$ where $1/t = \beta\lfloor \alpha n \rfloor/n \ge \alpha\beta - \beta/n$. Analogously, it holds that $|I \cap \mathcal{L}| \le (1 - 1/t)n$. In the following, we assume $n \ge N_1$ for some $N_1 = N_1(\Delta) > 0$, such that $1 - 1/t \le 1 - \alpha\beta + \beta/n = 1 - 1/\zeta + \beta/n \le 0.219$. We obtain an upper bound of $\sum_{I\in\mathcal{B}} \lambda^{|I|}$ as follows:

(a) Consider an independent set $I \in \mathcal{B}$. Recall that $\alpha n \le |I \cap \mathcal{L}| \le (1-1/t)n$. We first enumerate a subset $U \subseteq \mathcal{L}$ with $\alpha n \le |U| \le (1-1/t)n$ and then enumerate all independent sets I with $I \cap \mathcal{L} = U$. Since $1-1/t < 1/2$, there are at most $n\binom{n}{\lfloor (1-1/t)n \rfloor} \le n2^{H(1-1/t)n}$ ways to enumerate such a set U, where the inequality follows from Lemma 8.

(b) Now fix a set $U \subseteq \mathcal{L}$. Recall that every independent set $I \in \mathcal{B}$ satisfies $|I \cap \mathcal{R}| \le (1-1/t)n$. Therefore $\sum_{I\in\mathcal{B}:\,|I\cap\mathcal{L}|=U} \lambda^{|I|} = \lambda^{|U|} \sum_{I\in\mathcal{B}:\,|I\cap\mathcal{L}|=U} \lambda^{|I\cap\mathcal{R}|} \le \lambda^{(1-1/t)n}(\lambda+1)^{(1-1/t)n}$.

(c) Combining the first two steps we obtain $\sum_{I\in\mathcal{B}} \lambda^{|I|} \le n2^{H(1-1/t)n}\lambda^{(1-1/t)n}(\lambda+1)^{(1-1/t)n} = n2^{H(1-1/t)n}(\lambda^2+\lambda)^{(1-1/t)n}$.

Using $\sum_{I\in\mathcal{I}_\mathcal{L}\cup\mathcal{I}_\mathcal{R}} \lambda^{|I|} \ge (\lambda+1)^n$ and the upper bound above, we obtain

$$\frac{\sum_{I\in\mathcal{B}} \lambda^{|I|}}{\sum_{I\in\mathcal{I}_\mathcal{L}\cup\mathcal{I}_\mathcal{R}} \lambda^{|I|}} \le \frac{n2^{H(1-1/t)n}(\lambda^2+\lambda)^{(1-1/t)n}}{(\lambda+1)^n} = n(f(\lambda))^n, \tag{1}$$

where $f(\lambda) = 2^{H(1-1/t)} \cdot \frac{\lambda^{1-1/t}}{(\lambda+1)^{1/t}}$. Since $1-1/t < 1/t$, it follows from Lemma 9 that $f(\lambda) \le f(1) = 2^{H(1-1/t)-1/t} < 1$ for all $\lambda \ge 1$. So there exists some constant $C > 1$ such that Equation (1) $\le n(f(1))^n < C^{-n}$ for all $n > N \ge N_1$ where $N = N(\Delta)$ is another sufficiently large constant. ◀

▶ **Lemma 13.** *For $\Delta \ge 53$ and $\lambda \ge 1$, there are constants $C > 1$ and N so that for all $G \in \mathcal{G}^\Delta_{\alpha,\beta}$ with $n > N$ vertices on both sides, $Z_\mathcal{L}(G,\lambda) + Z_\mathcal{R}(G,\lambda)$ is a C^{-n}-relative approximation to $\sum_{I\in\mathcal{I}_\mathcal{L}(G)\cup\mathcal{I}_\mathcal{R}(G)} \lambda^{|I|}$.*

Proof. For any $I \in \mathcal{I}_\mathcal{L} \cap \mathcal{I}_\mathcal{R}$, it holds that $|I \cap \mathcal{L}| < \alpha n$ and $|I \cap \mathcal{R}| < \alpha n$. Clearly $\sum_{I\in\mathcal{I}_\mathcal{L}\cup\mathcal{I}_\mathcal{R}} \lambda^{|I|} \ge (\lambda+1)^n$. Therefore

$$\frac{\sum_{I\in\mathcal{I}_\mathcal{L}\cap\mathcal{I}_\mathcal{R}} \lambda^{|I|}}{\sum_{I\in\mathcal{I}_\mathcal{L}\cup\mathcal{I}_\mathcal{R}} \lambda^{|I|}} \le (\lambda+1)^{-n} \left(\sum_{k=0}^{\lfloor \alpha n \rfloor} \binom{n}{k} \lambda^k\right)^2 \le n^2 \left(\frac{4^{H(\alpha)}\lambda^{2\alpha}}{\lambda+1}\right)^n, \tag{2}$$

where the last inequality follows from Lemma 8. Recall that $\alpha = 2.9/\Delta$ and $\Delta \ge 53$. Then $\left.\frac{4^{H(\alpha)}\lambda^{2\alpha}}{\lambda+1}\right|_{\lambda=1} \le 0.76 < 1$. It follows from Lemma 9 that $4^{H(\alpha)}\lambda^{2\alpha}/(\lambda+1)$ is monotonically decreasing in λ on $[1,\infty)$ for all fixed $\Delta \ge 53$. Thus Equation (2) $\le \left(1/\left(0.76n^{2/n}\right)\right)^{-n} < C^{-n}$ for some constant $C > 1$ and for all $n > N$ where N is a sufficiently large constant. ◀

3.2 Approximating $Z_\mathcal{X}(G,\lambda)$

In this subsection, we discuss how to approximate $Z_\mathcal{X}(G,\lambda)$ for any graph $G \in \mathcal{G}^\Delta_{\alpha,\beta}, \mathcal{X} \in \{\mathcal{L},\mathcal{R}\}$ and $\lambda \ge 1$. We will use the polymer model (see Section 2.2). First we constructively define the polymers we need. For any $I \in \mathcal{I}_\mathcal{X}(G)$, we can partition the graph $(G^2)[I \cap \mathcal{X}]$ into connected components $U_1, U_2, \ldots, U_k$ for some $k \ge 0$ (trivially $k = 0$ if $I \cap \mathcal{X} = \emptyset$). There are no edges in G^2 between U_i and U_j for any $1 \le i \ne j \le k$. If $k > 0$, let $p(I) = \{(U_1, \mathbf{1}_{U_1}), (U_2, \mathbf{1}_{U_2}), \ldots, (U_k, \mathbf{1}_{U_k})\}$ where $\mathbf{1}_{U_i}$ is the unique mapping from U_i to $\{1\}$. If $k = 0$, let $p(I) = \emptyset$. We define the set of all polymers to be $\Gamma^*_\mathcal{X}(G) = \bigcup_{I\in\mathcal{I}_\mathcal{X}(G)} p(I)$ and each element in this set is called a polymer. When the graph G and $\mathcal{X}$ are clear from the context, we simply denote by Γ^* the set of polymers. Clearly, p is a mapping from $\mathcal{I}_\mathcal{X}(G)$ to the set $\left\{\Gamma \in \mathcal{S}(\Gamma^*_\mathcal{X}(G)) : \left|\overline{\Gamma}\right| < \alpha n\right\}$ since $\left|\overline{p(I)}\right| = |I \cap \mathcal{X}| < \alpha n$ for all $I \in \mathcal{I}_\mathcal{X}(G)$. For each polymer γ, define its weight function $w(\gamma,\cdot)$ as $w(\gamma,z) = \lambda^{|\overline{\gamma}|}(\lambda+1)^{-|N(\overline{\gamma})|}z^{|\overline{\gamma}|}$, where z is a complex variable. The weight function can be computed in polynomial time in $|\overline{\gamma}|$. The partition function of the polymer model (Γ^*, w) on the graph G^2 is the following sum: $\Xi(z) = \sum_{\Gamma\in\mathcal{S}(\Gamma^*)} \prod_{\gamma\in\Gamma} w(\gamma,z)$. Recall that two polymers γ_1 and γ_2 are compatible if $d_{G^2}(\overline{\gamma_1}, \overline{\gamma_2}) > 1$ and this condition is equivalent to $d_G(\overline{\gamma_1}, \overline{\gamma_2}) > 2$.

▶ **Lemma 14.** *For all bipartite graphs $G = (\mathcal{L}, \mathcal{R}, E)$ with n vertices on both sides, $\mathcal{X} \in \{\mathcal{L}, \mathcal{R}\}$ and $\lambda \geq 0$,*

$$Z_{\mathcal{X}}(G, \lambda) = (\lambda+1)^n \sum_{\Gamma \in \mathcal{S}(\Gamma^*_{\mathcal{X}}(G)):\, |\overline{\Gamma}| < \alpha n} \prod_{\gamma \in \Gamma} w(\gamma, 1).$$

Proof. In the definition of polymers, p is a mapping from $\mathcal{I}_{\mathcal{X}}$ to $\left\{\Gamma \in \mathcal{S}(\Gamma^*) : \left|\overline{\Gamma}\right| < \alpha n\right\}$. Thus $Z_{\mathcal{X}}(G, \lambda) = \sum_{I \in \mathcal{I}_{\mathcal{X}}} \lambda^{|I|} = \sum_{\Gamma \in \mathcal{S}(\Gamma^*):\, |\overline{\Gamma}| < \alpha n} \sum_{I \in \mathcal{I}_{\mathcal{X}}:\, p(I) = \Gamma} \lambda^{|I|}$. Fix $\Gamma \in \mathcal{S}(\Gamma^*)$ with $\left|\overline{\Gamma}\right| < \alpha n$. It holds that

$$\sum_{I \in \mathcal{I}_{\mathcal{X}}:\, p(I) = \Gamma} \lambda^{|I|} = \sum_{I \in \mathcal{I}_{\mathcal{X}}:\, I \cap \mathcal{X} = \overline{\Gamma}} \lambda^{|I|} = \lambda^{|\overline{\Gamma}|} (\lambda+1)^{\left|(\mathcal{L} \sqcup \mathcal{R}) \setminus (\mathcal{X} \sqcup N_G(\overline{\Gamma}))\right|}, \tag{3}$$

where the last equality follows from $\left|\overline{\Gamma}\right| < \alpha n$. Since Γ is compatible, $N_G(\overline{\Gamma}) = \sqcup_{\gamma \in \Gamma} N_G(\overline{\gamma})$ and $\left|(\mathcal{L} \sqcup \mathcal{R}) \setminus (\mathcal{X} \sqcup N_G(\overline{\Gamma}))\right| = n - \sum_{\gamma \in \Gamma} |N_G(\overline{\gamma})|$. Thus Equation (3) $= \lambda^{\sum_{\gamma \in \Gamma} |\overline{\gamma}|} (\lambda + 1)^{n - \sum_{\gamma \in \Gamma} N(\overline{\gamma})} = (\lambda+1)^n \prod_{\gamma \in \Gamma} \lambda^{|\overline{\gamma}|} (\lambda+1)^{-|N(\overline{\gamma})|} = (\lambda+1)^n \prod_{\gamma \in \Gamma} w(\gamma, 1)$. ◀

▶ **Lemma 15.** *For $\Delta \geq 53$ and $\lambda \geq 1$, there are constants $C > 1$ and N so that for all $G = (\mathcal{L}, \mathcal{R}, E) \in \mathcal{G}^{\Delta}_{\alpha,\beta}$ with $n > N$ vertices on both sides and $\mathcal{X} \in \{\mathcal{L}, \mathcal{R}\}$,*

$$(\lambda+1)^n \Xi(1) = (\lambda+1)^n \sum_{\Gamma \in \mathcal{S}(\Gamma^*_{\mathcal{X}}(G))} \prod_{\gamma \in \Gamma} w(\gamma, 1)$$

is a C^{-n}-relative approximation to $Z_{\mathcal{X}}(G, \lambda)$.

Proof. It is clear that $Z_{\mathcal{X}}(G, \lambda) \geq (\lambda+1)^n$. Then using Lemma 14 and Lemma 16 we obtain

$$\rho = \frac{(\lambda+1)^n \Xi(1) - Z_{\mathcal{X}}(G, \lambda)}{Z_{\mathcal{X}}(G, \lambda)} \leq \sum_{\Gamma \in \mathcal{S}(\Gamma^*):\, |\overline{\Gamma}| \geq \alpha n} \prod_{\gamma \in \Gamma} w(\gamma, 1) \leq \sum_{\Gamma \in \mathcal{S}(\Gamma^*):\, |\overline{\Gamma}| \geq \alpha n} 2^{-\beta |\overline{\Gamma}|}. \tag{4}$$

To enumerate each $\Gamma \in \mathcal{S}(\Gamma^*)$ with $\left|\overline{\Gamma}\right| \geq \alpha n$ at least once, we first enumerate an integer $\alpha n \leq k \leq n$, then since $\overline{\Gamma} \subseteq \mathcal{X}$, we choose k vertices from $\mathcal{X}$. Therefore, from Equation (4) we have

$$\rho \leq \sum_{k = \lceil \alpha n \rceil}^{n} \binom{n}{k} 2^{-\beta k} \leq \sum_{k = \lceil \alpha n \rceil}^{n} 2^{H(k/n) n} 2^{-\beta k} \leq \sum_{k = \lceil \alpha n \rceil}^{n} \left(2^{2\sqrt{n/k} - \beta}\right)^k \leq \sum_{k = \lceil \alpha n \rceil}^{n} \left(2^{2\sqrt{1/\alpha} - \beta}\right)^k,$$

where the inequalities follow from Lemma 8 and Lemma 7. Recall that $\zeta = 1.28$, $\alpha = 2.9/\Delta, \beta = \Delta/(2.9\zeta)$ and $\Delta \geq 53$. Let $f(\Delta) = 2\sqrt{1/\alpha} - \beta = 2\sqrt{\Delta/2.9} - \Delta/(2.9\zeta)$. We obtain $\rho \leq \frac{2^{f(\Delta)\alpha n}}{1 - 2^{f(\Delta)}} = \frac{\left(2^{2\sqrt{2.9/\Delta} - 1/\zeta}\right)^n}{1 - 2^{f(\Delta)}}$. Since $f(\Delta)$ is monotonically decreasing in Δ on $[53, +\infty)$, $\rho \leq \frac{\left(2^{2\sqrt{2.9/53} - 1/1.28}\right)^n}{1 - 2^{2\sqrt{53/2.9} - 53/(2.9 \times 1.28)}} \leq 0.81^n / 0.98 < C^{-n}$ for some constant $C > 1$ and for all $n > N$ where N is a sufficiently large constant. ◀

▶ **Lemma 16.** *For all polymers $\gamma \in \Gamma^*$ defined by $G = (\mathcal{L}, \mathcal{R}, E) \in \mathcal{G}^{\Delta}_{\alpha,\beta}$, $\mathcal{X} \in \{\mathcal{L}, \mathcal{R}\}$ and $\lambda \geq 1$, $|w(\gamma, z)| \leq (2^{-\beta}|z|)^{|\overline{\gamma}|}$. As a corollary, $w(\gamma, 1) \leq 2^{-\beta|\overline{\gamma}|}$ and for all compatible $\Gamma \subseteq \Gamma^*(G)$, $\prod_{\gamma \in \Gamma} w(\gamma, 1) \leq 2^{-\beta|\overline{\Gamma}|}$.*

Proof. Let $n = |\mathcal{L}| = |\mathcal{R}|$ and let γ be any polymer. It follows from the definition of polymers that $|\overline{\gamma}| \leq \alpha n$ and by the expansion property, $|N(\overline{\gamma})| \geq \beta |\overline{\gamma}|$. Thus we have $|w(\gamma, z)| = \lambda^{|\overline{\gamma}|} (\lambda+1)^{-|N(\overline{\gamma})|} |z|^{|\overline{\gamma}|} \leq (\lambda(\lambda+1)^{-\beta})^{|\overline{\gamma}|} |z|^{|\overline{\gamma}|} \leq (2^{-\beta}|z|)^{|\overline{\gamma}|}$ where the last inequality follows from Lemma 9 since $\beta > 1$ and $\lambda \geq 1$. In particular, $w(\gamma, 1) \leq 2^{-\beta|\overline{\gamma}|}$. For any compatible Γ, it holds that $\left|\overline{\Gamma}\right| = \sum_{\gamma \in \Gamma} |\overline{\gamma}|$. Thus $\prod_{\gamma \in \Gamma} w(\gamma, 1) \leq \prod_{\gamma \in \Gamma} 2^{-\beta|\overline{\gamma}|} = 2^{-\beta|\overline{\Gamma}|}$. ◀

3.3 Approximating the partition function of the polymer model

▶ **Lemma 17.** *For $\Delta \geq 53$ and $\lambda \geq 1$, there is an FPTAS for $\Xi(1)$ for all $G = (\mathcal{L}, \mathcal{R}, E) \in \mathcal{G}^{\Delta}_{\alpha,\beta}$ and $\mathcal{X} \in \{\mathcal{L}, \mathcal{R}\}$.*

Proof. Apply the FPTAS in Theorem 4. ◀

To apply Theorem 4, we need to show that for the parameters in Lemma 17, the partition function has no zeros in the entire unit disk centered at 0.

▶ **Lemma 18.** *There is a constant $R > 1$ so that for $\Delta \geq 53$ and $\lambda \geq 1$, $\Xi(z) \neq 0$ for all $G \in \mathcal{G}^{\Delta}_{\alpha,\beta}$, $\mathcal{X} \in \{\mathcal{L}, \mathcal{R}\}$ and $z \in \mathbb{C}$ with $|z| < R$.*

Proof. Set $R = 1.001$. For any $\gamma \in \Gamma^*$, let $a(\gamma) = t|\overline{\gamma}|$ where $t = \left(-1 + \sqrt{1+8e}\right)/(4e) \approx 0.346$. We will verify that the KP-condition $\sum_{\gamma:\gamma \not\sim \gamma^*} e^{t|\overline{\gamma}|}|w(\gamma, z)| \leq t\left|\overline{\gamma^*}\right|$ holds for any $\gamma^* \in \Gamma^*$ and any $|z| < R$. It then follows from Lemma 5 that $\Xi(z) \neq 0$ for any $|z| < R$. Recall that $d_{G^2}(\overline{\gamma}, \overline{\gamma^*}) \leq 1$ for all $\gamma \not\sim \gamma^*$. Thus there is always a vertex $v \in \overline{\gamma} \subseteq \mathcal{X}$ such that $v \in \overline{\gamma^*} \sqcup N_{G^2}(\overline{\gamma^*})$. The number of such vertices v is at most $\Delta^2\left|\overline{\gamma^*}\right|$. So to enumerate each $\gamma \not\sim \gamma^*$ at least once, we can: a) first enumerate a vertex v in $\mathcal{X} \cap (\overline{\gamma^*} \cup N_{G^2}(\overline{\gamma^*}))$; b) then enumerate an integer k from 1 to $\lfloor \alpha n \rfloor$; c) finally enumerate γ with $v \in \overline{\gamma}$ and $|\overline{\gamma}| = k$. Since $\overline{\gamma}$ is connected in G^2, applying Lemma 6 and using Lemma 16 to bound $|w(\gamma, z)|$ we obtain $\sum_{\gamma:\gamma \not\sim \gamma^*} e^{t|\overline{\gamma}|}|w(\gamma, z)| \leq \Delta^2\left|\overline{\gamma^*}\right| \left(e^t 2^{-\beta}|z| + \sum_{k=2}^{\lfloor \alpha n \rfloor} \left(e\Delta^2\right)^{k-1} 2^{-1} e^{tk} 2^{-\beta k}|z|^k\right)$. Let $x = e^{t+1}\Delta^2 2^{-\beta} R$. Since $|z| < R$, we obtain $\sum_{\gamma:\gamma \not\sim \gamma^*} e^{t|\overline{\gamma}|}|w(\gamma, z)| \leq \frac{x}{e}\left|\overline{\gamma^*}\right| \left(1 + \frac{1}{2}\sum_{k=2}^{\infty} x^{k-1}\right) = \frac{x(2-x)}{2e(1-x)} \cdot \left|\overline{\gamma^*}\right|$. Recall that $\zeta = 1.28$, $\beta = \Delta/(2.9\zeta)$ and $\Delta \geq 53$. Since $\Delta^2 2^{-\beta}$ is monotonically decreasing in Δ on $[53, +\infty)$, it holds that $x = e^{t+1}\Delta^2 2^{-\beta} R \leq \left(e^{t+1}\Delta^2 2^{-\beta} R\right)\big|_{\Delta=53} \leq 0.545$, and hence $\frac{x(2-x)}{2e(1-x)} < 0.33 < t$. ◀

3.4 Putting things together

Using the results from previous parts, we obtain our main result for counting independent sets.

▶ **Theorem 1.** *For $\Delta \geq 53$ and fugacity $\lambda \geq 1$, with high probability (tending to 1 as $n \to \infty$) for a graph G chosen uniformly at random from $\mathcal{G}^{\text{bip}}_{n,\Delta}$, there is an FPTAS for the partition function $Z(G, \lambda)$.*

Proof. This theorem follows from Lemma 10 and Lemma 19. ◀

▶ **Lemma 19.** *For $\Delta \geq 53$ and $\lambda \geq 1$, there is an FPTAS for $Z(G, \lambda)$ for all $G \in \mathcal{G}^{\Delta}_{\alpha,\beta}$.*

Proof. First we state our algorithm. See Algorithm 1 for a pseudocode description. The input is a graph $G = (\mathcal{L}, \mathcal{R}, E) \in \mathcal{G}^{\Delta}_{\alpha,\beta}$ and an approximation parameter $\varepsilon > 0$. The output is a number $\widehat{Z}$ to approximate $Z(G, \lambda)$. We use $\Xi_{\mathcal{X}}(z)$ to denote the partition function of the polymer model $(\Gamma^*_{\mathcal{X}}(G), w)$ for $\mathcal{X} \in \{\mathcal{L}, \mathcal{R}\}$. Let N_1, C_2, N_2, C_2 be the constants in Lemma 11 and Lemma 15, respectively. These two lemmas show that $(\lambda+1)^n\left(\Xi_{\mathcal{L}}(1) + \Xi_{\mathcal{R}}(1)\right)$ is a $C_1^{-n} + C_2^{-n} \leq 2\min(C_1, C_2)^{-n} \leq C^{-n}$-relative approximation to $Z(G, \lambda)$ for another constant $C > 1$ and all $n > N \geq \max(N_1, N_2)$ where N is another sufficiently large constant. If $n \leq N$ or $\varepsilon \leq 2C^{-n}$, we use the brute-force algorithm to compute $Z(G, \lambda)$. If $\varepsilon > 2C^{-n}$, we apply the FPTAS in Lemma 17 with approximation parameter $\varepsilon' = \varepsilon - C^{-n}$ to obtain outputs $\widehat{Z}_{\mathcal{L}}$ and $\widehat{Z}_{\mathcal{R}}$ which approximate $\Xi_{\mathcal{L}}(1)$ and $\Xi_{\mathcal{R}}(1)$, respectively. Let $\widehat{Z} = (\lambda+1)^n(\widehat{Z}_{\mathcal{L}} + \widehat{Z}_{\mathcal{R}})$ be the output. It is clear that $\exp(-\varepsilon)\widehat{Z} \leq Z(G, \lambda) \leq \exp(\varepsilon)\widehat{Z}$.

Algorithm 1 Counting independent sets at fugacity $\lambda \geq 1$ for $\Delta \geq 53$.

Input: *A graph* $G = (\mathcal{L}, \mathcal{R}, E) \in \mathcal{G}^{\Delta}_{\alpha,\beta}$ *with* n *vertices on both sides and* $\varepsilon > 0$
Output: $\widehat{Z}$ *such that* $\exp(-\varepsilon)\widehat{Z} \leq Z(G, \lambda) \leq \exp(\varepsilon)\widehat{Z}$
if $n \leq N$ or $\varepsilon \leq 2C^{-n}$ **then**
    Use the brute-force algorithm to compute $\widehat{Z} \leftarrow Z(G, \lambda)$;
    Exit;
end if
$\varepsilon' \leftarrow \varepsilon - C^{-n}$;
Use the FPTAS in Lemma 17 to obtain $\widehat{Z}_{\mathcal{L}}$, an ε'-relative approximation to the partition function $\Xi(z)$ at $z = 1$ of the polymer model $(\Gamma^*_{\mathcal{L}}(G), w)$.
Use the FPTAS in Lemma 17 to obtain $\widehat{Z}_{\mathcal{R}}$, an ε'-relative approximation to the partition function $\Xi(z)$ at $z = 1$ of the polymer model $(\Gamma^*_{\mathcal{R}}(G), w)$.
$\widehat{Z} \leftarrow (\lambda + 1)^n \left(\widehat{Z}_{\mathcal{L}} + \widehat{Z}_{\mathcal{R}}\right)$;

Then we show that Algorithm 1 is indeed an FPTAS. It is required that the running time of our algorithm is bounded by $(n/\varepsilon)^{C_3}$ for some constant C_3 and for all $n > N_3$ where N_3 is a constant. Let $N_3 = N$. If $\varepsilon \leq 2C^{-n}$, the running time of the algorithm would be $2.1^n \leq (nC^n/2)^{C_3} \leq (n/\varepsilon)^{C_3}$ for sufficient large C_3. If $\varepsilon > 2C^{-n}$, the running time of the algorithm would be $(n/\varepsilon')^{C_4} = (n/(\varepsilon - C^{-n}))^{C_4} \leq (2n/\varepsilon)^{C_4} \leq (n/\varepsilon)^{C_3}$ for sufficient large C_3, where C_4 is a constant from the FPTAS in Lemma 17. ◀

References

1 Alexander I. Barvinok. *Combinatorics and Complexity of Partition Functions*, volume 30 of *Algorithms and combinatorics*. Springer, 2016. doi:10.1007/978-3-319-51829-9.

2 Alexander I. Barvinok and Pablo Soberón. Computing the partition function for graph homomorphisms with multiplicities. *J. Comb. Theory, Ser. A*, 137:1–26, 2016. doi:10.1016/j.jcta.2015.08.001.

3 Christian Borgs, Jennifer T. Chayes, Jeff Kahn, and László Lovász. Left and right convergence of graphs with bounded degree. *Random Struct. Algorithms*, 42(1):1–28, 2013. doi:10.1002/rsa.20414.

4 Russ Bubley and Martin Dyer. Path coupling: A technique for proving rapid mixing in Markov chains. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science (FOCS'97)*, pages 223–231. IEEE, 1997.

5 Russ Bubley, Martin Dyer, Catherine Greenhill, and Mark Jerrum. On approximately counting colorings of small degree graphs. *SIAM Journal on Computing*, 29(2):387–400, 1999.

6 Andrei A. Bulatov, Martin E. Dyer, Leslie Ann Goldberg, Mark Jerrum, and Colin McQuillan. The expressibility of functions on the Boolean domain, with applications to counting CSPs. *J. ACM*, 60(5):32:1–32:36, 2013. doi:10.1145/2528401.

7 Jin-Yi Cai, Andreas Galanis, Leslie Ann Goldberg, Heng Guo, Mark Jerrum, Daniel Štefankovič, and Eric Vigoda. #BIS-hardness for 2-spin systems on bipartite bounded degree graphs in the tree non-uniqueness region. *J. Comput. Syst. Sci.*, 82(5):690–711, 2016. doi:10.1016/j.jcss.2015.11.009.

8 Sitan Chen, Michelle Delcourt, Ankur Moitra, Guillem Perarnau, and Luke Postle. Improved Bounds for Randomly Sampling Colorings via Linear Programming. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2216–2234, 2019.

9 Martin Dyer, Abraham D Flaxman, Alan M Frieze, and Eric Vigoda. Randomly coloring sparse random graphs with fewer colors than the maximum degree. *Random Structures & Algorithms*, 29(4):450–465, 2006.

10 Martin Dyer and Alan Frieze. Randomly coloring graphs with lower bounds on girth and maximum degree. *Random Structures & Algorithms*, 23(2):167–179, 2003.

11 Martin Dyer, Alan Frieze, Thomas P Hayes, and Eric Vigoda. Randomly coloring constant degree graphs. *Random Structures & Algorithms*, 43(2):181–200, 2013.

12 Martin E. Dyer, Alan M. Frieze, and Mark Jerrum. On Counting Independent Sets in Sparse Graphs. *SIAM J. Comput.*, 31(5):1527–1541, 2002. doi:10.1137/S0097539701383844.

13 Martin E. Dyer, Leslie Ann Goldberg, Catherine S. Greenhill, and Mark Jerrum. The Relative Complexity of Approximate Counting Problems. *Algorithmica*, 38(3):471–500, 2004. doi:10.1007/s00453-003-1073-y.

14 Martin E. Dyer, Leslie Ann Goldberg, and Mark Jerrum. An approximation trichotomy for Boolean #CSP. *J. Comput. Syst. Sci.*, 76(3-4):267–277, 2010. doi:10.1016/j.jcss.2009.08.003.

15 Andreas Galanis, Leslie Ann Goldberg, and Kuan Yang. Approximating Partition Functions of Bounded-Degree Boolean Counting Constraint Satisfaction Problems. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 27:1–27:14, 2017. doi:10.4230/LIPIcs.ICALP.2017.27.

16 Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Inapproximability of the Partition Function for the Antiferromagnetic Ising and Hard-Core Models. *Combinatorics, Probability & Computing*, 25(4):500–559, 2016. doi:10.1017/S0963548315000401.

17 Andreas Galanis, Daniel Štefankovič, Eric Vigoda, and Linji Yang. Ferromagnetic Potts Model: Refined #BIS-hardness and Related Results. *SIAM J. Comput.*, 45(6):2004–2065, 2016. doi:10.1137/140997580.

18 David Gamarnik and Dmitriy Katz. Correlation decay and deterministic FPTAS for counting colorings of a graph. *Journal of Discrete Algorithms*, 12:29–47, 2012.

19 Leslie Ann Goldberg and Mark Jerrum. Approximating the partition function of the ferromagnetic potts model. *J. ACM*, 59(5):25:1–25:31, 2012. doi:10.1145/2371656.2371660.

20 Leslie Ann Goldberg and Mark Jerrum. A complexity classification of spin systems with an external field. *Proceedings of the National Academy of Sciences of the United States of America*, 43(112):13161–13166, 2015.

21 Heng Guo, Chao Liao, Pinyan Lu, and Chihao Zhang. Counting hypergraph colourings in the local lemma regime. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 926–939, 2018. doi:10.1145/3188745.3188934.

22 Thomas P Hayes. Randomly coloring graphs of girth at least five. In *Proceedings of the 35th Annual ACM Symposium on Symposium on Theory of Computing (STOC'03)*, pages 269–278. ACM, 2003.

23 Thomas P Hayes and Eric Vigoda. A non-Markovian coupling for randomly sampling colorings. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*, pages 618–627. IEEE, 2003.

24 Thomas P Hayes and Eric Vigoda. Coupling with the stationary distribution and improved sampling for colorings and independent sets. *The Annals of Applied Probability*, 16(3):1297–1318, 2006.

25 Tyler Helmuth, Will Perkins, and Guus Regts. Algorithmic Pirogov-Sinai Theory. *CoRR*, abs/1806.11548, 2018. arXiv:1806.11548.

26 Matthew Jenssen, Peter Keevash, and Will Perkins. Algorithms for #BIS-hard problems on expander graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2235–2247, 2019. doi:10.1137/1.9781611975482.135.

27 Mark Jerrum. A very simple algorithm for estimating the number of k-colorings of a low-degree graph. *Random Structures and Algorithms*, 7(2):157–166, 1995.

28 R. Kotecký and D. Preiss. Cluster expansion for abstract polymer models. *Communications in Mathematical Physics*, 103(3):491–498, September 1986. doi:10.1007/BF01211762.

29 Jingcheng Liu and Pinyan Lu. FPTAS for #bis with degree bounds on one side. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 549–556, 2015. doi:10.1145/2746539.2746598.

30 Pinyan Lu, Kuan Yang, Chihao Zhang, and Minshen Zhu. An FPTAS for Counting Proper Four-Colorings on Cubic Graphs. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1798–1817, 2017.

31 Pinyan Lu and Yitong Yin. Improved FPTAS for multi-spin systems. In *Proceedings of APPROX-RANDOM*, pages 639–654. Springer, 2013.

32 Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, 2017.

33 Michael Molloy. The Glauber dynamics on colorings of a graph with high girth and maximum degree. *SIAM Journal on Computing*, 33(3):721–737, 2004.

34 Elchanan Mossel, Dror Weitz, and Nicolas Wormald. On the hardness of sampling independent sets beyond the tree threshold. *Probability Theory and Related Fields*, 143(3):401–439, 2009. doi:10.1007/s00440-007-0131-9.

35 Viresh Patel and Guus Regts. Deterministic polynomial-time approximation algorithms for partition functions and graph polynomials. *Electronic Notes in Discrete Mathematics*, 61:971–977, 2017. doi:10.1016/j.endm.2017.07.061.

36 S. A. Pirogov and Ya. G. Sinai. Phase diagrams of classical lattice systems. *Theoretical and Mathematical Physics*, 25(3):1185–1192, December 1975. doi:10.1007/BF01040127.

37 S. A. Pirogov and Ya. G. Sinai. Phase diagrams of classical lattice systems continuation. *Theoretical and Mathematical Physics*, 26(1):39–49, January 1976. doi:10.1007/BF01038255.

38 Allan Sly. Computational Transition at the Uniqueness Threshold. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 287–296, 2010. doi:10.1109/FOCS.2010.34.

39 Allan Sly and Nike Sun. The Computational Hardness of Counting in Two-Spin Models on d-Regular Graphs. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 361–369, 2012. doi:10.1109/FOCS.2012.56.

40 Eric Vigoda. Improved bounds for sampling colorings. *Journal of Mathematical Physics*, 41(3):1555–1569, 2000.

41 Dror Weitz. Counting independent sets up to the tree threshold. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 140–149. ACM, 2006. doi:10.1145/1132516.1132538.

The Expected Number of Maximal Points of the Convolution of Two 2-D Distributions

Josep Diaz
Department of CS, UPC, Barcelona, Spain
diaz@cs.upc.edu

Mordecai Golin
CSE Department, Hong Kong UST
golin@cse.ust.hk

Abstract

The *Maximal* points in a set S are those that are not *dominated* by any other point in S. Such points arise in multiple application settings and are called by a variety of different names, e.g., maxima, Pareto optimums, skylines. Their ubiquity has inspired a large literature on the *expected* number of maxima in a set S of n points chosen IID from some distribution. Most such results assume that the underlying distribution is uniform over some spatial region and strongly use this uniformity in their analysis.

This research was initially motivated by the question of how this expected number changes if the input distribution is perturbed by random noise. More specifically, let $\mathbf{B}_p$ denote the uniform distribution from the 2-dimensional unit ball in the metric L_p. Let $\delta\mathbf{B}_q$ denote the 2-dimensional L_q-ball, of radius δ and $\mathbf{B}_p + \delta\mathbf{B}_q$ be the convolution of the two distributions, i.e., a point $v \in \mathbf{B}_p$ is reported with an error chosen from $\delta\mathbf{B}_q$. The question is how the expected number of maxima change as a function of δ. Although the original motivation is for small δ, the problem is well defined for any δ and our analysis treats the general case.

More specifically, we study, as a function of n, δ, the expected number of maximal points when the n points in S are chosen IID from distributions of the type $\mathbf{B}_p + \delta\mathbf{B}_q$ where $p, q \in \{1, 2, \infty\}$ for $\delta > 0$ and also of the type $\mathbf{B}_\infty + \delta\mathbf{B}_q$ where $q \in [1, \infty)$ for $\delta > 0$.

For fixed p, q we show that this function changes "smoothly" as a function of δ but that this smooth behavior sometimes transitions unexpectedly between different growth behaviors.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases maximal points, probabilistic geometry, perturbations, Minkowski sum

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.35

Category RANDOM

Related Version https://arxiv.org/abs/1807.06845

Funding *Josep Diaz*: TIN2017-86727-C2-1R

1 Introduction

Let S be a set of 2-dimensional points. The "largest" points in S are the *maximal points* of S and are a well-studied object. More formally

▶ **Definition 1.** *For $u \in \Re^2$ let $u.x$ ($u.y$) denote the x (y) coordinate of u. For $u, v \in \Re^2$, u is* dominated *by v if $u \neq v$, $u.x \leq v.x$ and $u.y \leq v.y$. If $S \subset \Re^2$ then*

$$\mathrm{MAX}(S) = \{u \in S : u \text{ is not dominated by any point in } S \setminus \{u\}\}.$$

$\mathrm{MAX}(S)$ *are the* maximal points *of S. See Fig. 1.*


© Josep Diaz and Mordecai Golin;
licensed under Creative Commons License CC-BY
Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh; Article No. 35; pp. 35:1–35:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Figure 1 The diagram shows $\mathrm{MAX}(S_n)$ for two point sets S_n. In both (a) and (b) the circles denote the points in S_n and the (red) filled circles are $\mathrm{MAX}(S_n)$. If the points are considered as being drawn from region D, $P(v)$, as introduced in Def. 2, denotes the region in D that dominates v. In (a), D is the dotted square; in (b), D is the dotted circle.

The problems of finding and estimating the number of maximal points of a set in $\Re^2$ appear very often in many fields under different names: *maximal vectors*, *skylines*, *Pareto frontier/points* and others, see e.g. [5, 12, 15, 17, 18] for a more exhaustive history of the problems, uses in Computer Science and further references, Sections 1 and 2 in [7].

Let S_n denote a set of n points chosen Independently Identically Distributed (IID) from some 2-D distribution $\mathbf{D}$ and $M_n = |\mathrm{MAX}(S_n)|$ be the random variable counting the number of maximal points in S_n. Because maxima are so ubiquitous, understanding the expected number of maxima has been important in different areas and many properties of M_n have been studied. More specifically, if $\mathbf{D}$ is the uniform distribution drawn from an L_p ball with $p \geq 1$, then it is well known [2, 6, 12, 14], that

- If $p = \infty$, then $\mathbf{E}[M_n] = H_n \sim \ln n$.
  The same result holds if the points are drawn from some distribution $\mathbf{D} = (\mathbf{X}, \mathbf{Y})$ where $\mathbf{X}$ and $\mathbf{Y}$ are any two 1-dimensional distributions that are independent of each other.
- If $p \geq 1$, then $\lim_{n\to\infty} \frac{\mathbf{E}[M_n]}{\sqrt{n}} = C_p$, where C_p is a constant dependent only upon p.
- Similar upper bounds to the above, i.e., that $\mathbf{E}[M_n] = O(\sqrt{n})$, derived using similar techniques, are known if $\mathbf{D}$ is a *uniform* distribution from ANY convex region [11].

It is also known [16] that if the n points are chosen IID from a 2-D Gaussian distribution then $\mathbf{E}[M_n] \sim \ln n$. There are also generalizations of these results (both the $\mathbf{B}_p$ ones and the Gaussian one) to higher dimensions. See [14] for a table containing most known results.

Surprisingly, given the importance of the problem, not much is known for other distributions. The motivation for this work is to extend the family of distributions for which $\mathbf{E}[M_n]$ can be derived.

Consider a point u originally generated from a uniform distribution over a unit L_p ball but measured or reported with an error, in the L_q metric, of at most δ. The actual reported point can be equivalently considered as being chosen from a new distribution which we denote by $\mathbf{B}_p + \delta\mathbf{B}_q$ (the next section provides formal definitions). The support of this distribution is the Minkowksi sum of the two balls but the distribution is not uniform over this support. Fig. 2 shows the support of $\mathbf{B}_p + \delta\mathbf{B}_q$, for different values of p and q.

Although the problem described above originally assumed small δ, it is well defined for all $\delta > 0$, which is the problem analyzed in this paper. More specifically, the motivation for the present work is twofold:

- Explain how $\mathbf{E}[M_n]$ changes when the distribution is perturbed.
  (Note: the perturbation size δ may be specified as a function of the sample size n.)
- Increase the families of distributions for which $\mathbf{E}[M_n]$ is understood.

The idea of analyzing how quantities change under perturbations could also be considered from the perspective of *smoothed analysis* [20, 21]. In the classic setting, smoothed analysis of the number of maxima would mean analyzing how, given a *fixed* set S_n, $\mathbf{E}[M_n]$ would change under small perturbations (as a function of the original set S_n). This was the approach in [9, 8] (see similar work for convex hulls in [10]). This paper differs in that it is the *Distribution* that is being smoothed (or convoluted) and not the point set. This paper also differs from recent work [22, 1] on the *most-likely* skyline and convex hull problems. Those papers assume each point has a given probability distribution and are attempting to find the subset of points that has the highest probability of being the skyline (or convex hull).

Outline of the paper. The next section defines the problem and states and explains our results. Sec. 3 describes key technical and conceptual ideas and tools used to achieve the main result. Sec. 4 describes how these tools are used to derive the result. Sec. 5 provides a review and a collection of open problems and possible extensions.

Due to space limitations, the proofs of many of the lemmas and theorems are not included. For the full proofs, please see the extended version of this paper [13] posted on Arxiv.

2 Definitions and Results

Let "$p \in [1,\infty)$" and "$p \geq 1$" both denote that p is a finite real number ≥ 1. $p = \infty$ also being permitted will be denoted by $p \in [1,\infty]$.

Recall: Let $\delta \geq 0$.
For $u \in \Re^2$, $\delta u = (\delta \cdot u.x,\ \delta \cdot u.y)$. For $u, v \in \Re^2$, $u + v = (u.x + v.x, u.y + v.y)$.
If $D \subseteq \Re^2$, $\delta D = \{(\delta u\ :\ u \in D\}$.
For $D_1, D_2 \subseteq \Re^2$, $D_1 + D_2 = \{u_1 + u_2\ :\ u_1 \in D_1,\ u_2 \in D_2\}$ will denote the *Minkowski sum* of D_1 and D_2.
For $u \in \Re^2$, $u + D$ will denote $\{u\} + D$.

Balls and Unit Balls: Let $u \in \Re^2$, $r > 0$ and $p \in [1,\infty)$. Define:

- The L_p *ball of radius r around u* as $B_p(u,r) = \{(x,y)\ :\ |x - u.x|^p + |y - u.y|^p \leq r^p\}$.
- The L_∞ *ball of radius r around u* as $B_\infty(u,r) = \{(x,y)\ :\ \max(|x - u.x|,\ |y - u.y|) \leq r\}$.
- The respective *unit balls* as $B_p = B_p((0,0),1)$ and $B_\infty = B_\infty((0,0),1)$.

Set $a_p = \text{Area}(B_p)$ to be the area of the L_p unit ball. Then $a_\infty = 4, a_1 = 2, a_2 = \pi$. We use the fact that $a_p = \Theta(1)$.

Generation of a probability distribution: Let $\mathbf{D}$ be a distribution with support $D \subset \Re^2$. Then

- If $\delta \geq 0$, the distribution $\delta\mathbf{D}$ is generated by choosing a point u using $\mathbf{D}$ and then returning the point δu.
- Let $\mathbf{D}_1, \mathbf{D}_2$ be two distributions over $\Re^2$. Generate the *convolution* $\mathbf{D}_1 + \mathbf{D}_2$ by choosing a point u_1 from $\mathbf{D}_1$ and a point u_2 from $\mathbf{D}_2$ and returning $u_1 + u_2$.
- A set $S_n = \{u_1, \ldots, u_n\}$ is said to be *chosen from* $\mathbf{D}$ if each u_i is generated *independently and identically distributed* (IID) using the distribution $\mathbf{D}$.

Uniform distribution on unit balls: For all $p \in [1, \infty]$, $\mathbf{B}_p$ will denote the uniform distribution that selects a point uniformly from B_p. This distribution has support B_p with uniform density $1/a_p$ within B_p.

Convolution of two distributions: Let $\mathbf{B}_p + \delta\mathbf{B}_q$ be the convolution of distributions $\mathbf{B}_p$ and $\delta\mathbf{B}_q$.
$(\mathbf{B}_p + \delta\mathbf{B}_q)$'s *support* of this distribution is the Minkowski sum $B_p + \delta B_q$. Observe that the density of $\mathbf{B}_p + \delta\mathbf{B}_q$ is **not** uniform in $B_p + \delta B_q$. It is this non-uniformity that will cause complications in calculating $\mathbf{E}[M_n]$. The main result of this paper is

▶ **Theorem 1.** *Fix p, q so that either $p, q \in \{1, 2, \infty\}$ or $p = \infty$ and $q \geq 1$.*
Let S_n be n points chosen from the distribution $\mathbf{B}_p + \delta\mathbf{B}_q$ and $M_n = |\mathrm{MAX}(S_n)|$.
Let $\delta \geq 0$ be a function of n. Then $\mathbf{E}[M_n]$ behaves as below:

	(a)	(b)	(c)	(d)	(e)	(f)
	$\mathbf{D} =$	$0 \leq \delta$				$\delta = 1$
(i)	$\mathbf{B}_\infty + \delta\mathbf{B}_\infty$	$\Theta(\ln n)$				$\Theta(\ln n)$
		$\delta \leq \frac{1}{\sqrt{n}}$	$\frac{1}{\sqrt{n}} \leq \delta \leq 1$	$1 \leq \delta \leq \sqrt{n}$	$\sqrt{n} \leq \delta$	
(ii)	$\mathbf{B}_1 + \delta\mathbf{B}_1$	$\Theta(\sqrt{n})$	$\Theta\left(\frac{n^{1/3}}{\delta^{1/3}}\right)$	$\Theta\left(\delta^{1/3} n^{1/3}\right)$	$\Theta(\sqrt{n})$	$\Theta\left(n^{1/3}\right)$
(iii)	$\mathbf{B}_2 + \delta\mathbf{B}_2$	$\Theta(\sqrt{n})$	$\Theta\left(\frac{n^{2/7}}{\delta^{3/7}}\right)$	$\Theta\left(\delta^{3/7} n^{2/7}\right)$	$\Theta(\sqrt{n})$	$\Theta\left(n^{2/7}\right)$
		$\delta \leq \frac{1}{\sqrt{n}}$	$\frac{1}{\sqrt{n}} \leq \delta \leq \sqrt{n}$		$\sqrt{n} \leq \delta$	
(iv)	$\mathbf{B}_\infty + \delta\mathbf{B}_q$	$\Theta(\ln n)$	$\Theta\left(\ln n + \sqrt{\delta} n^{1/4}\right)$		$\Theta(\sqrt{n})$	$\Theta\left(n^{1/4}\right)$
		$\delta \leq \frac{1}{\sqrt{n}}$	$\frac{1}{\sqrt{n}} \leq \delta \leq n^{1/26}$	$n^{1/26} \leq \delta \leq \sqrt{n}$	$\sqrt{n} \leq \delta$	
(v)	$\mathbf{B}_1 + \delta\mathbf{B}_2$	$\Theta(\sqrt{n})$	$\Theta\left(\frac{n^{2/7}}{\delta^{3/7}}\right)$	$\Theta\left(\sqrt{\delta} n^{1/4}\right)$	$\Theta(\sqrt{n})$	$\Theta\left(n^{2/7}\right)$

Interpretation of the table:

1. When $p = q = \infty$, M_n has exactly the same distribution as if S_n were chosen from $\mathbf{B}_\infty$, so row (i) is an uninteresting case, only included for completeness.
2. When δ is small enough ($\leq 1/\sqrt{n}$), $\mathbf{E}[M_n]$ behaves almost as if S_n were chosen from $\mathbf{B}_p$ and when δ is large enough ($\geq \sqrt{n}$) $\mathbf{E}[M_n]$ behaves almost as if S_n were chosen from $\mathbf{B}_q$. This is reflected in columns (b) and (e).
3. Lemma 8 states that M_n has the same distribution for S_n chosen from both $\mathbf{B}_p + \delta\mathbf{B}_q$ and $\mathbf{B}_q + \frac{1}{\delta}\mathbf{B}_p$. Thus row (iv) gives the behavior for $\mathbf{B}_q + \delta\mathbf{B}_\infty$ for any $q \geq 1$ and row (v) the behavior for $\mathbf{B}_2 + \delta\mathbf{B}_1$.
4. When $p = q \in \{1, 2\}$, $\mathbf{E}[M_n]$ starts at $\Theta(\sqrt{n})$, smoothly decreases until reaching $\delta = 1$ and then increases again until reaching $\Theta(\sqrt{n})$. The behavior at $\delta = 1$ is different for $p = q = 1$ and $p = q = 2$. In both cases there is symmetry between δ and $1/\delta$ (from Lemma 8).
5. When $p = 1$, $q = 2$ there is no symmetry. The behavior starts at $\Theta(\sqrt{n})$, decreases to $\Theta\left(n^{7/26}\right)$ at $\delta = n^{1/26}$ and then increases again *at a different rate* to $\Theta(\sqrt{n})$.
6. When $p = \infty$, the behavior is asymptotically equivalent for all $q \in [1, \infty)$, not just $q = 1, 2$. The only difference is in the value of the constant hidden by the Θ. The behavior starts at $\Theta(\ln n)$, stays there for a short while and then smoothly increases to $\Theta(\sqrt{n})$.

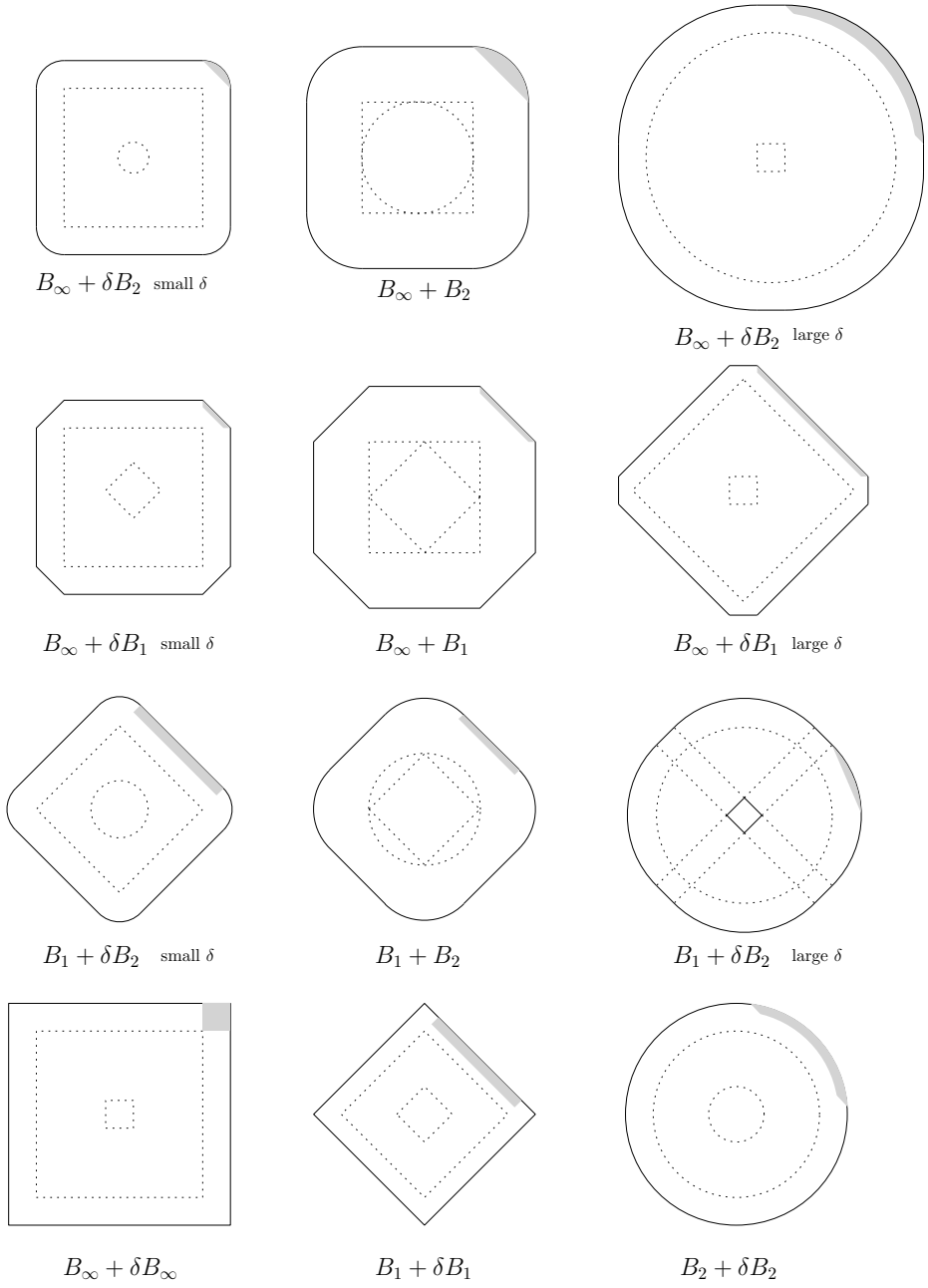


Figure 2 Illustrations of the supports of some of the different distributions in the form $\mathbf{B}_p + \delta\mathbf{B}_q$ examined in Theorem 1. The dotted lines denote the B_p and δB_q balls centred at 0. Note that in all cases the density is uniform near the centre of the support but then decreases to 0 as the boundary is approached. The grey areas denote, approximately, where the maxima of S_n are concentrated.

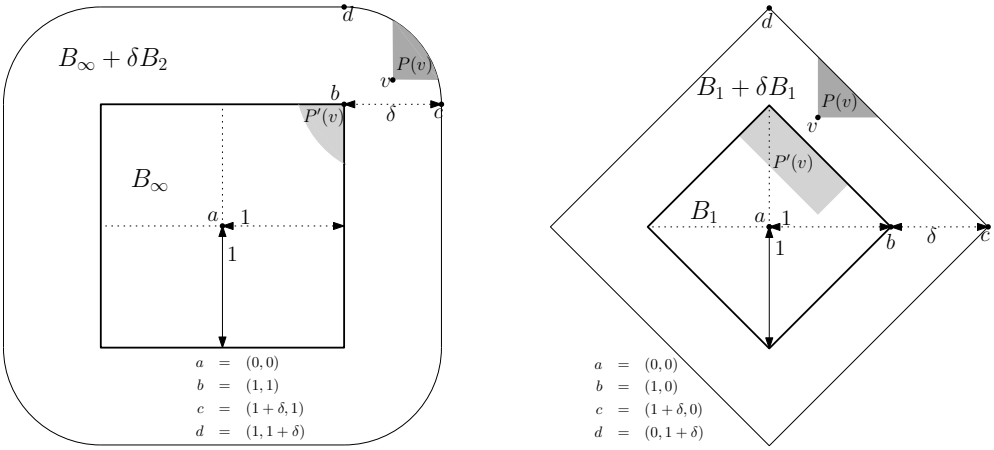


Figure 3 Illustration of definitions of $P(v)$ and $P'(v)$ for $B_p + \delta B_q$. Left side is is $B_\infty + \delta B_2$; right is $B_1 + \delta B_1$. In both diagrams the interior ball (heavy boundary) is the B_p ball centered at the origin a. $P(v)$ is the set of points in $B_p + \delta B_q$ that dominate v and $P'(v)$ is the preimage of v in B_p.

3 Basic Lemmas

The following collection of Lemmas comprise the basic toolkit used to derive Theorem 1.

Recall: Let $\mathbf{D}$ be a distribution over $\Re^2$, $x \in \Re^2$ and $A \subset \Re^2$ a measurable region. Then $f_{\mathbf{D}}(x)$ will denote the *density function* of $\mathbf{D}$, and $\mu_{\mathbf{D}}(A) = \int_A f_{\mathbf{D}}(x)dx$ will denote the *measure* of A under distribution $\mathbf{D}$. If $\mathbf{D}$ is understood, we often simply write $f(x)$ and $\mu(A)$.

▶ **Definition 2.** *(See Fig. 3)*
Let $D \subseteq \Re^2$, $v \in D$ and $A \subseteq D$.
Define: $P(v) = \{u \in D : u \text{ dominates } v\} \cup \{v\}$, and $P(A) = \bigcup_{v \in A} P(v)$.
Say that A is dominant in D *or a* dominant region in D, *if $P(A) = A$.*

Note that, by definition, $\forall v \in D$, $P(v)$ is a dominant region in D. It is straightforward to see that

▶ **Lemma 1.** *Let v and S_n be chosen from $\mathbf{D}$ and $A \subseteq D$. Then*

$$\begin{array}{lrcl} (a) & \Pr(v \in A) & = & \mu(A). \\ (b) & \mathbf{E}\left[|A \cap S_n|\right] & = & n\mu(A). \\ (c) & \Pr(A \cap S_n = \emptyset) & = & (1-\mu(A))^n. \end{array}$$

The following observation will be used to prove most of our lower bounds.

▶ **Lemma 2** (Lower Bound)**.** *Let S_n be chosen from $\mathbf{D}$. Further let $A_1, A_2, \ldots, A_m$ be a collection of pairwise disjoint dominant regions in D with $\mu(A_i) = \Omega(1/n)$ for all i. Then*

$$\mathbf{E}\left[M_n\right] \geq \mathbf{E}\left[\left|\text{MAX}\left(S_n \cap \bigcup_{i=1}^{m} A_i\right)\right|\right] = \Omega(m).$$

Proof. From Lemma 1, $\Pr(S_n \cap A_i = \emptyset) = (1-\mu(A_i))^n$. Thus $\mu(A_i) = \Omega(1/n)$ implies

$$\Pr(S_n \cap A_i \neq \emptyset) = 1 - \Pr(S_n \cap A_i = \emptyset) = \Omega(1).$$

If region A is dominant then points in A can only be dominated by other points in A then $A \cap \text{MAX}(S_n) = \text{MAX}(S_n \cap A)$. Since each A_i is dominant, this implies

$$\mathbf{E}\left[|\text{MAX}(S_n) \cap A_i|\right] \geq \Pr(S_n \cap A_i \neq \emptyset) = \Omega(1).$$

Since the A_i are pairwise disjoint,

$$\mathbf{E}\left[|MAX(S_n)|\right] \geq \mathbf{E}\left[\left|MAX(S_n) \cap \left(\bigcup_i A_i\right)\right|\right] \geq \sum_{i=1}^{m} \Omega(1) = \Omega(m). \quad \blacktriangleleft$$

▶ **Definition 3.** *(See Fig. 3)*
Let $D = B_p + \delta B_q$. For $v \in D$ define the preimage *of v in B_p as*

$$P'(v) = B_q(v, \delta) \cap B_p = (v + \delta B_q) \cap B_p.$$

▶ **Lemma 3.** *Fix $p, q \in [1, \infty]$. Let $\mathbf{D} = \mathbf{B}_p + \delta \mathbf{B}_q$ and let v be a point chosen from $\mathbf{D}$. Let $A \subseteq \Re^2$. Then*

$$f(v) = \frac{1}{a_p a_q} \frac{\text{Area}(\{u \in B_p \,:\, v - u \in \delta B_q\})}{\delta^2} = \frac{1}{a_p a_q} \frac{\text{Area}(P'v)}{\delta^2} \tag{1}$$

$$\mu(A) = \frac{1}{a_p a_q} \int_{u \in B_p} \frac{\text{Area}((u + \delta B_q) \cap A)}{\delta^2} du. \tag{2}$$

Proof. Note that for $u \in B_p$, $f_{\mathbf{B}_p}(u) = \frac{1}{a_p}$ and for $u' \in \delta B_q$, $f_{\delta \mathbf{B}_q}(u') = \frac{1}{a_q \delta^2}$. To see Eq. 2,

$$\mu(A) = \int_{u \in \mathbf{B}_p} \left(\int_{\substack{w \in \delta B_q \\ u+w \in A}} f_{\delta \mathbf{B}_q}(w) dw \right) f_{\mathbf{B}_p}(u) du = \frac{1}{a_p a_q} \int_{u \in B_p} \frac{\text{Area}\left((u + \delta B_q) \cap A\right)}{\delta^2} du.$$

For Eq. 1, use a change of variables $v = u + w$,

$$\begin{aligned} \mu(A) = & \quad \frac{1}{a_p a_q \delta^2} \int_{u \in B_p} \left(\int_{\substack{w \in \delta B_q \\ u+w \in A}} dw \right) du \\ = & \quad \frac{1}{a_p a_q \delta^2} \int_{u \in B_p} \left(\int_{\substack{v \in u + \delta B_q \\ v \in A}} dv \right) du = \quad \frac{1}{a_p a_q} \int_{v \in A} \frac{\text{Area}\left\{u \in B_p \,:\, v - u \in \delta B_q\right\}}{\delta^2} dv. \end{aligned}$$

Differentiating around v yields Eq. 1. ◀

▶ **Lemma 4.** *Fix $p, q \in [1, \infty]$. Let $\mathbf{D} = \mathbf{B}_p + \delta \mathbf{B}_q$ and $\kappa > 0$ be any constant. Then*

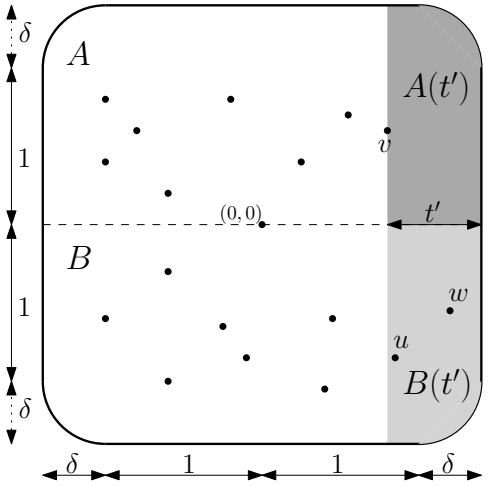
(a)	$v \in D$	$\Rightarrow$	$f(v) = O(1)$.
(b)	$v \in B_p$ and $\delta \leq \kappa$	$\Rightarrow$	$f(v) = \Theta(1)$.
(c)	$A \subseteq D$	$\Rightarrow$	$\mu(A) = O(\text{Area}(A))$.
(d)	$A \subseteq B_p$ and $\delta \leq \kappa$	$\Rightarrow$	$\mu(A) = \Theta(\text{Area}(A))$.

The constants implicit in the $O()$ in (a) and (c) are only dependent upon p, q, while the constants implicit in the $\Theta()$ in (b) and (d) are only dependent upon p, q, κ.

Proof.
(a) Use the fact that, for $\forall u \in B_p$,

$$\text{Area}(B_p \cap (u + \delta B_q))) \leq \text{Area}(u + \delta B_q) = a_q \delta^2,$$

so from Eq. 1, $f(v) = O(1)$.



Distribution is $\mathbf{D} = \mathbf{B}_\infty + \delta\mathbf{B}_2$.

A is D above the x-axis.
B is D below the x-axis.

$$\begin{aligned} A(t) &= \{u \in A : u.x \geq 1 + \delta - t\} \\ B(t) &= \{u \in B : u.x \geq 1 + \delta - t\} \end{aligned}$$

$\forall t$, $A(t)$ and $B(t)$ have the same measure.

$S_n \cap B(t') = \{u, w\}$.

$$X(t') = |S_n \cap B(t')| = 2$$

Any point in $A(t)$ dominates all points in $B \setminus B(t)$.

$$\begin{aligned} \Rightarrow \quad \mathrm{MAX}(S_n) \cap B &\subseteq (S_n \cap B(t')) \\ &= X(t') \\ \Rightarrow \quad |\mathrm{MAX}(S_n) \cap B| &\leq 2 \end{aligned}$$

Figure 4 Illustration of Lemmas 5 and 6. The regions A and B are each swept by parameter t and it is required that $\mu(B(t)) = O(\mu(A(t))$. In the case above, by the symmetry of distribution $\mathbf{D}$, $\mu(B(t)) = \mu(A(t))$ trivially. t' is the first time a point in $A(t)$ is found. Since every point in $A(t)$ dominates all points in $B \setminus B(t)$, all maxima in $S_n \cap B$ must be in $B(t')$. The definition of t' intuitively implies that $\mu(A(t')) \sim \frac{1}{n}$ so, also intuitively, the expectation of $|S_n \cap B_n|$ should be $n\mu(B(t')) \sim 1$. This is proven formally in the text.

(b) If $u \in B_p$ then

$$\mathrm{Area}(B_p \cap (u + \delta B_q))) \geq c\mathrm{Area}(u + \delta B_q) = ca_q\delta^2,$$

where c is only dependent upon p, q, κ. Thus, from Eq. 1, $f(v) = \Theta(1)$.

The proofs for (c) and (d) follow from plugging (a) and (b) into Eq. 2. ◀

▶ **Lemma 5.** *(See Fig. 4)*
Let $\mathbf{D}$ *be any distribution with a continuous density function* $f(u)$ *and* S_n *a set of points chosen from* $\mathbf{D}$. *Let* A, B *be two disjoint regions in the support* D *that are parameterized by* $t \in [0, T]$ *and satisfy:*

- $\mu(A(0)) = \emptyset$.
- $A(T) = A$; $B(T) = B$.
- *(Monotonicity in t)* $\forall t_1 < t_2$, $A(t_1) \subseteq A(t_2)$ *and* $B(t_1) \subseteq B(t_2)$.
- $\mu(B(t)), \mu(A(t))$ *are both continuous in* t.
- *(Asymptotic dominance in measure)* $\forall t$, $\mu(B(t)) = O(\mu(A(t))$.

Define the random variables

$$X = |S_n \cap B(t')|, \qquad t' = \begin{cases} \min\{t : A(t) \cap S_n \neq \emptyset\} & \text{if } A \cap S_n \neq \emptyset, \\ T & \text{if } A \cap S_n = \emptyset. \end{cases}$$

Then, $\mathbf{E}[X] = O(1)$. (3)

Proof. W.l.o.g. rescale t so that $\mu(A(t)) = t$, and $T = \mu(A)$.

The proof's intuition is that since the "first" point in A appears at t', then $\mu(A(t')) \sim \frac{1}{n}$. As B is asymptotically dominated by A, $\mu(B(t'))$=$O(1/n)$ and $\mathbf{E}[X(t')] = n\mu(B(t')) = O(1)$.

Formally, by the continuity of the measure, $\Pr(|S_n \cap A(t')| = 1) = 1$. So we may assume that $|D \setminus A(t')| = n - 1$.

Conditioned on known t', the remaining $n-1$ points in S_n are chosen from $D \setminus A(t')$ with the associated conditional distribution. If u is one of those $n-1$ points,

$$\Pr\left(u \in B(t') \;\middle|\; t'\right) = \frac{\mu(B(t'))}{\mu(D \setminus A(t'))} = \frac{\mu(B(t'))}{1 - \mu(A(t'))}.$$

Thus, conditioning on t', and applying Lemma 1(b)

$$\mathbf{E}\left[X \;\middle|\; t'\right] = (n-1)\frac{\mu(B(t'))}{1 - \mu(A(t'))},$$

$$\text{therefore } \mathbf{E}[X] = \mathbf{E}\left[\mathbf{E}\left[X \;\middle|\; t'\right]\right] = \mathbf{E}\left[(n-1)\frac{\mu(B(t'))}{1 - \mu(A(t'))}\right].$$

From the definition of t' and Lemma 1 (c), $\mu(A(t')) > 1/2$ with exponentially low probability. Therefore, recalling that $\mu(A(t)) = t$,

$$\mathbf{E}[X] = (n-1)\mathbf{E}[O(\mu(B(t')))] = (n-1)\mathbf{E}[O(\mu(A(t')))] = (n-1)O(\mathbf{E}[t']).$$

$$\text{Using Lemma 1 (c) : } \mathbf{E}[t'] = \int_{\alpha=0}^{T} \Pr(t' \geq \alpha) d\alpha = O\left(\tfrac{1}{n-1}\right). \quad \blacktriangleleft$$

▶ **Lemma 6** (Sweep). *(See Fig. 4)*
Let $\mathbf{D}$ be any distribution with a continuous density function $f(u)$, and let S_n be a set of points chosen from $\mathbf{D}$.

Let A, B be two disjoint regions in the support D that are parameterized by $t \in [0, T]$, satisfy conditions 1-3 of Lemma 5 and, in addition satisfy that

$$\forall t \in [0,T], \quad \text{if } u \in A(t) \text{ and } v \in B \setminus B(t) \text{ then } u \text{ dominates } v.$$

In such a case we say that A continuously dominates *B. Then*

$$\mathbf{E}[|\mathrm{MAX}(S_n) \cap B|] = O(1). \tag{4}$$

Proof. By the definition of t', $|A(t') \cap S_n| \geq 1$. Since all points in $B \setminus B(t')$ are dominated by all points in $A(t')$, $\mathrm{MAX}(S_n) \cap (B \setminus B(t')) = \emptyset$. Thus from Lemma 5,

$$\mathbf{E}[|\mathrm{MAX}(S_n) \cap B|] = \mathbf{E}[|\mathrm{MAX}(S_n) \cap B(t')|] \leq \mathbf{E}[|S_n \cap B(t')|] = O(1). \quad \blacktriangleleft$$

▶ **Corollary 7.** *Fix $p, q \in [1, \infty]$ and choose S_n from $\mathbf{D} = \mathbf{B}_p + \delta\mathbf{B}_q$. Let Q_1 be the positive (upper-right) quadrant of the plane and O_1 the first octant , i.e., $Q_1 = \{u \in \Re^2 : 0 \leq u.x, 0 \leq u.y\}$ and $O_1 = \{u \in \Re^2 : 0 \leq u.y \leq u.x\}$. Then*

$$\mathbf{E}[M_n] = \mathbf{E}[|\mathrm{MAX}(S_n)|] = \mathbf{E}[|Q_1 \cap \mathrm{MAX}(S_n)|] + O(1) \tag{5}$$

$$= \Theta\Big(\mathbf{E}[|O_1 \cap \mathrm{MAX}(S_n)|]\Big). \tag{6}$$

Proof. Restrict $t \in [0, 2 + 2\delta]$ and set

$$A = D \cap \{u \in \Re^2 : u.y \geq 0\}, \qquad A(t) = \{u \in A : u.x \geq 1 + \delta - t\},$$
$$B = D \cap \{u \in \Re^2 : u.y < 0\}, \qquad B(t) = \{u \in B : u.x \geq 1 + \delta - t\}.$$

Conditions (1) and (2) of Lemma 5 trivially hold. Condition (3) holds because, by x-axis symmetry, $\mu(B(t)) = \mu(A(t))$. The additional condition of Lemma 6 holds because every point in $B \setminus B(t)$ is below and to the left of every point in $A(t)$. Thus the expected number of maximal points in S_n *below* the x-axis is $O(1)$. Note that this is *independent* of n.

Similarly, the expected number of maximal points to the left of the y-axis is $O(1)$. This proves Eq. 5.

To prove Eq. 6 define the second octant to be $O_2 = \{u \in \Re^2 : 0 \le u.x \le u.y\}$. By the symmetry between the x and y coordinates in the distribution,

$$\mathbf{E}\left[|O_1 \cap \mathrm{MAX}(S_n)|\right] = \mathbf{E}\left[|O_2 \cap \mathrm{MAX}(S_n)|\right].$$

Futhermore, since O_1 and O_2 partition Q_1,

$$\mathbf{E}\left[|Q_1 \cap \mathrm{MAX}(S_n)|\right] = \mathbf{E}\left[|O_1 \cap \mathrm{MAX}(S_n)|\right] + \mathbf{E}\left[|O_2 \cap \mathrm{MAX}(S_n)|\right] = 2\mathbf{E}\left[|O_1 \cap \mathrm{MAX}(S_n)|\right].$$

Thus

$$\mathbf{E}\left[M_n\right] = \mathbf{E}\left[Q_1 \cap |\mathrm{MAX}(S_n)|\right] + O(1) = \Theta\left(\mathbf{E}\left[|O_1 \cap \mathrm{MAX}(S_n)|\right]\right). \quad \blacktriangleleft$$

The fact that for $\delta > 0$, u dominates v if and only if δu dominates δv implies the following result which is used very often in this work,

▶ **Lemma 8** (Scaling). *Fix $p, q \in [1, \infty]$, $\mathbf{D} = \mathbf{B}_p + \delta\mathbf{B}_q$ and $\mathbf{D}' = \mathbf{B}_q + \frac{1}{\delta}\mathbf{B}_p$. Let S_n be n points chosen from $\mathbf{D}$ and let S'_n be n points chosen from $\mathbf{D}'$. Then $|\mathrm{MAX}(S_n)|$ and $|\mathrm{MAX}(S'_n)|$ have exactly the same distribution. In particular, $\mathbf{E}\left[|\mathrm{MAX}(S_n)|\right] = \mathbf{E}\left[|\mathrm{MAX}(S'_n)|\right]$.*

Proof. Let $S_n = \{u_1, \ldots, u_n\}$ be chosen from $\mathbf{D}$. Recall that the process of choosing point u from $\mathbf{D}$ is to choose w from $\mathbf{B}_p$, v from $\mathbf{B}_q$ and return $u = w + \delta v$. Choosing a point u' from $\mathbf{D}'$ is the same except that it returns $u' = v + \frac{1}{\delta}w = \frac{1}{\delta}u$. Thus the distribution of choosing $S_n = \{u_1, \ldots, u_n\}$ from $\mathbf{D}$ is exactly the same as choosing $S_n = \{\frac{1}{\delta}u_1, \ldots, \frac{1}{\delta}u_n\}$ from $\mathbf{D}'$.

Finally, note that dominance is invariant under multiplication by a scalar, i.e., p_i dominates p_j if and only if $\frac{1}{\delta}p_i$ dominates $\frac{1}{\delta}p_j$.

Thus $|\mathrm{MAX}(S_n)|$ and $|\mathrm{MAX}(S'_n)|$ have the same distribution, so $\mathbf{E}\left[|\mathrm{MAX}(S_n)|\right] = \mathbf{E}\left[|\mathrm{MAX}(S'_n)|\right]$. ◀

The next lemma formalizes the intuition that for small values of δ, the value of $\mathbf{E}\left[M_n\right]$ for $\mathbf{B}_p + \delta\mathbf{B}_q$ is the same as the value for $\mathbf{B}_p$.

▶ **Lemma 9** (Limiting Behavior). *Let $p \in [1, \infty]$, $q \in [1, \infty)$, $\delta = O(1/\sqrt{n})$ and S_n chosen from $\mathbf{D} = \mathbf{B}_p + \delta\mathbf{B}_q$. Then*

$$\mathbf{E}\left[M_n\right] = \begin{cases} \Theta(\ln n) & \text{if } p = \infty, \\ \Theta(\sqrt{n}) & \text{if } p \neq \infty. \end{cases}$$

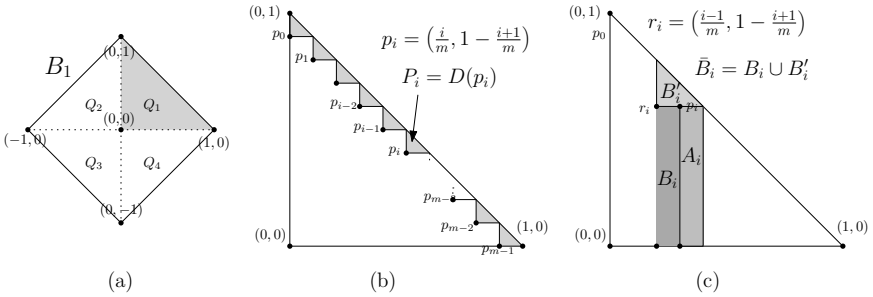
4 General approach to proving Theorem 1

Note that if u is chosen from $\mathbf{B}_\infty$, then $u.x$ and $u.y$ are independent random variables. Thus, for any $\delta > 0$ if v is chosen from $\mathbf{D} = \mathbf{B}_\infty + \delta\mathbf{B}_\infty$, $v.x$ and $v.y$ are independent random variables. As noted in the introduction, this means that if S_n is chosen from $\mathbf{D}$, $\mathbf{E}\left[M_n\right]$ is exactly the same as if S_n was chosen from $\mathbf{B}_\infty$, i.e., $\mathbf{E}\left[M_n\right] = \Theta(\ln n)$, proving row (i).

Lemma 9 combined with Lemma 8 imply the limiting behavior in columns (b) and (e) of the table in Theorem 1. Note too that for rows (ii) and (iii), column (d) follows directly from applying Lemma 8 to column (c).

Thus, proving Theorem 1 reduces to proving cells (ii) c, (iii) c, (iv) c, d and (v) c, d.

Proving Theorem 1 will require case-by-case analyses of $\mathbf{D} = \mathbf{B}_p + \delta\mathbf{B}_q$ for the different pairs p, q. The analysis for each pair will all follow the same 4 step pattern:



(a) (b) (c)

Figure 5 Illustration of proof $\mathbf{E}[M_n] = \Theta(\sqrt{n})$ when S_n is chosen from $\mathbf{B}_1$ All but $O(1)$ maxima will be in quadrant Q_1; (b) and (c) illustrate Q_1. (b) illustrates the lower bound and (c) the upper.

4.1 A Simple Example: $\mathbf{D} = \mathbf{B}_1$

Before sketching our results it is instructive to see how the Lemmas in the previous section can be used to re-derive that fact that, if $\mathbf{D} = \mathbf{B}_1$ then $\mathbf{E}[M_n] = \Theta(\sqrt{n})$. See Fig. 5.

Even though the behavior for $\mathbf{D} = \mathbf{B}_1$ is already well known we provide this to illustrate the generic steps for deriving $\mathbf{E}[M_n]$. These are exactly the same steps that are needed when $\mathbf{D} = \mathbf{B}_p + \delta\mathbf{B}_q$ and this example permits identifying where the complications can arise in those more general cases. Set $m = \lfloor\sqrt{n}\rfloor$ and let p_i, r_i be the points defined in the figure with $P_i = P(p_i)$ and $B'_i = P(r_i)$. Also set

$$B_i = \left\{(x,y) \,:\, \frac{i-1}{m} \le x \le \frac{i}{m},\ 0 \le y \le 1 - \frac{i+1}{m}\right\}, \quad A_i = \left(\frac{1}{m}, 0\right) + B_i$$

and $\bar{B}_i = B_i \cup B'_i$. Finally, for $0 \le t \le (1+i)/m$ set $B_i(t) = B_i \cap \{(x,y) \,:\, y \le (1+i)/m - t\}$ and $A_i(t) = \left(\frac{1}{m}, 0\right) + B_i(t)$. The steps in the derivation are.

Step 1: *Restricting to first Quadrant:*
Corollary 7 states that $\mathbf{E}[M_n] = \mathbf{E}[|Q_1 \cap \mathrm{MAX}(S_n)|] + O(1)$.

Step 2: *Calculating Density and Measure:*
Because $\mathbf{D}$ has a uniform density, $\mu(A) = \Theta(\mathrm{Area}(A))$ for all regions $A \subseteq D$.

Step 3: *Lower Bound:*
The P_i are a collection of m pairwise disjoint dominant regions with

$$\mu(P_i) = \Theta(\mathrm{Area}(P_i)) = \Theta(m^{-2}) = \Theta(1/n).$$

Thus, from Lemma 2, $\mathbf{E}[M_n] = \Omega(m) = \Omega(\sqrt{n})$.

Step 4: *Upper bound:*
Note that $Q_1 \cap D = \left(\bigcup_{i=1}^{m-1} \bar{B}_i\right) \cup B'_m$ so

$$\mathbf{E}[|\mathrm{MAX}(S_n) \cap Q_1|] = \mathbf{E}\left[\left|\mathrm{MAX}(S_n) \cap \left(\bigcup_{i=1}^{m} \bar{B}_i\right)\right|\right] + \mathbf{E}[|\mathrm{MAX}(S_n) \cap B'_m|],$$

$$\mathbf{E}\left[\left|\mathrm{MAX}(S_n) \cap \left(\bigcup_{i=1}^{m} \bar{B}_i\right)\right|\right] \le \sum_{i=1}^{m} \mathbf{E}[|\mathrm{MAX}(S_n) \cap B_i|] + \sum_{i=1}^{m} \mathbf{E}[|\mathrm{MAX}(S_n) \cap B'_i|].$$

Furthermore, $\forall i$, $\mu(B'_i) = \Theta(\mathrm{Area}(B'_i)) = \Theta(1/n)$. Thus

$$\forall i, \quad \mathbf{E}[|\mathrm{MAX}(S_n) \cap B'_i|] \le \mathbf{E}[|S_n \cap B'_i|] = O(n\mu(B'_i)) = O(1).$$

Since $m = O(\sqrt{n})$ this yields

$$\mathbf{E}\left[|\mathrm{MAX}(S_n) \cap Q_1|\right] \leq \sum_{i=1}^{m} \mathbf{E}\left[|\mathrm{MAX}(S_n) \cap B_i|\right] + O(\sqrt{n}).$$

The crucial observation is that, $\forall i$, A_i continuously dominates B_i as defined in Lemmas 5 and 6. Thus, plugging into Lemma 6 yields $\forall i, \quad \mathbf{E}\left[|\mathrm{MAX}(S_n) \cap B_i|\right] = O(1)$, leading to

$$\mathbf{E}\left[|\mathrm{MAX}(S_n) \cap Q_1|\right] = O(m) + O(\sqrt{n}) = O(\sqrt{n}).$$

Combining the $\mathbf{E}\left[|\mathrm{MAX}(S_n) \cap Q_1|\right] = \Omega(\sqrt{n})$ from step (3) with the $\mathbf{E}\left[|\mathrm{MAX}(S_n) \cap Q_1|\right] = O(\sqrt{n})$ from step (4) with step (1) gives the final result

$$\mathbf{E}\left[M_n\right] = \mathbf{E}\left[|\mathrm{MAX}(S_n) \cap Q_1|\right] + O(1) = \Theta(\sqrt{n}) + O(1) = \Theta(\sqrt{n}).$$

4.2 The general approach for $\mathrm{D} = \mathrm{B}_p + \delta\mathrm{B}_q$

For each p, q pair the proof of Theorem 1 follows the same four steps as the analysis of $\mathbf{D} = \mathbf{B}_1$ above.

Step 1: *Restricting to first Quadrant:*
Corollary 7 again states that $\mathbf{E}\left[M_n\right] = \mathbf{E}\left[|Q_1 \cap \mathrm{MAX}(S_n)|\right] + O(1)$.

Step 2: *Calculating Density $f(u)$ and Measure $\mu(A)$:*
This step is often quite technical. In the example $\mathbf{D} = \mathbf{B}_1$ case above, the density was constant. For general $\mathbf{D}$ this is no longer true. The density is constant in some region in the center of the support but decreases to zero as the boundary is approached. While Lemma 3 provides an integral formula for general $\mathbf{D}$ this, in many cases, is unusable. A substantial amount of technical work is involved in finding usable functional representations for the densities/measures in different parts of the support.

Step 3: *Lower Bounding $\mathbf{E}\left[M_n\right]$:*
For most cases this is a relatively straightforward application of Lemma 2 using the results of Step 2. In the general case, it is still necessary to identify a region that contains an asymptotically dominant number of maxima. It is then necessary to partition this region into pairwise disjoint dominant regions, all of which have measure $\Theta(1/n)$. Note that, unlike in the example $\mathbf{D} = \mathbf{B}_1$ case, these regions might no longer all have the same shape or size.

Step 4: *Upper bounding $\mathbf{E}\left[M_n\right]$:*
This is the most delicate part of the proof. It is proven using the Sweep Lemma (Lemma 6) with the major difficulties arising from how to decompose the support into regions that continuously dominate each other. This decomposition strongly depends upon *how* the measure/density is *represented* in Step 2 and can be very differently structured in different parts of the support. In particular, in the case $\mathbf{D} = \mathbf{B}_1 + \delta\mathbf{B}_2$, there are two different parts of the support that require two different decompositions and the decompositions must be designed so that the two upper bounds derived match each other.

More broadly, the density/measure representations developed for $\mathbf{D}_1 = \mathbf{B}_1 + \delta\mathbf{B}_1$ and $\mathbf{D}_2 = \mathbf{B}_2 + \delta\mathbf{B}_2$ are quite different. The analysis of $\mathbf{D}_3 = \mathbf{B}_1 + \delta\mathbf{B}_2$ which is the most delicate, combines the approaches developed for $\mathbf{D}_1$, $\mathbf{D}_2$. The analysis of $\mathbf{D}_4 = \mathbf{B}_\infty + \delta\mathbf{B}_q$ is different from the first three, but much more straightforward.

5 Conclusion

This paper developed a suite of tools for deriving the expected number of maximal points in a set of n points chosen IID from $\mathbf{B}_p + \delta\mathbf{B}_q$, which is the convolution of two distributions.

The results presented here seem to be the first general analysis of $\mathbf{E}[M_n]$ for non-uniform and non-Gaussian distributions. This paper is only a first step. Obvious next steps are

- The results in the paper were only proven for $p, q \in \{1, 2, \infty\}$ and $p = \infty, q \in [1, \infty.]$ The next step would be to attempt to extend the results to *all* pairs $p, q, \in [1, \infty]$.
- There is a rich literature stretching back more than fifty years on the average number of points on the *convex hull* of points chosen IID from a uniform distribution in a planar region or a Gaussian distribution, e.g., [14, 19]. It would be interesting to see how the convex hull evolves in the convoluted distributions $\mathbf{B}_p + \delta\mathbf{B}_q$.
  Such an analysis would require a much tighter understanding of how the distribution behaves "close" to the boundary of its support $B_p + \delta B_q$. One approach might be to introduce some form of measure weighting to the definition of *Macbeath-regions* [3] (which are a known technique for characterizing this boundary region).
- Finally, we note that the results on $\mathbf{E}[M_n]$ for n points chosen IID from a uniform distribution over an L_p ball have analogues in higher dimensions, i.e., $\Theta\left(\log^{d-1} n\right)$ if $p = \infty$ and $\Theta\left(n^{1-\frac{1}{d}}\right)$ if $p \in [1, \infty)$ [4, 14]. The next step would be to attempt to extend the results in this paper to higher dimensions.

References

1 Akash Agrawal, Yuan Li, Jie Xue, and Ravi Janardan. The most-likely skyline problem for stochastic points. *Proc. 29th CCCG*, pages 78–83, 2017.

2 Zhi-Dong Bai, Luc Devroye, Hsien-Kuei Hwang, and Tsung-Hsi Tsai. Maxima in hypercubes. *Random Struct. Algorithms*, 27(3):290–309, 2005.

3 I Bárány. The technique of M-regions and cap-coverings: a survey. *Rendiconti di Palermo*, 65:21–38, 2000.

4 Yuri Baryshnikov. On expected number of maximal points in polytopes. In *Discrete Mathematics and Theoretical Computer Science*, pages 247–258, 2007.

5 Stephan Börzsönyi, Donald Kossmann, and Konrad Stocker. The Skyline Operator. In *Proceedings of the 17th International I.C.D.E.,*, pages 421–430. IEEE Computer Society, 2001.

6 Christian Buchta. On the average number of maxima in a set of vectors. *Information Processing Letters*, 33:63–65, 1989.

7 Wei-Mei Chen, Hsien-Kuei Hwang, and Tsung-Hsi Tsai. Maxima-finding algorithms for multi-dimensional samples: A two-phase approach. *Comput. Geometry: Theory and Applications*, 45(1-2):33–53, 2012.

8 Valentina Damerow. *Average and smoothed complexity of geometric structures.* PhD thesis, University of Paderborn, Germany, 2006.

9 Valentina Damerow and Christian Sohler. Extreme Points Under Random Noise. In *Algorithms – ESA 2004*, pages 264–274, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

10 Olivier Devillers, Marc Glisse, Xavier Goaoc, and Rémy Thomasse. Smoothed complexity of convex hulls by witnesses and collectors. *Journal of Computational Geometry*, 7(2):101–144, 2016.

11 Luc Devroye. *Lecture notes on bucket algorithms.* Birkhauser Boston, 1986.

12 Luc Devroye. Records, the maximal layer, and uniform distributions in monotone sets. *Computers Math. Applic.*, 25(5):19–31, 1993.

13 Josep Diaz and Mordecai Golin. Smoothed Analysis of the Expected Number of Maximal Points in Two Dimensions. *arXiv preprint*, 2018. `arXiv:1807.06845`.

14 R A Dwyer. Kinder, gentler average-case analysis for convex hulls and maximal vectors. *SIGACT News*, 21(2):64–71, 1990.

15 Marc Geilen, Twan Basten, Bart Theelen, and Ralph Otten. An algebra of Pareto points. *Fundamenta Informaticae*, 78(1):35–74, 2007.

16 V. M. Ivanin. Asymptotic estimate for the mathematical expectation of the number of elements in the Pareto set. *Cybernetics*, 11(1):108–113, 1975.

17 J.L.Bentley, H.T. Kung, M. Schkolnick, and C.D. Thompson. On the average number of maxima in a set of vectors and its applications. *Jour. ACM*, 25(4):536–543, 1978.

18 H. T. Kung, Fabrizio Luccio, and Franco P. Preparata. On Finding the Maxima of a Set of Vectors. *J. ACM*, 22(4):469–476, 1975.

19 Alfréd Rényi and Rolf Sulanke. Über die konvexe hülle von n zufällig gewählten punkten. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 2(1):75–84, 1963.

20 Daniel A Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM (JACM)*, 51(3):385–463, 2004.

21 Daniel A Spielman and Shang-Hua Teng. Smoothed analysis: an attempt to explain the behavior of algorithms in practice. *Communications of the ACM*, 52(10):76–84, 2009.

22 Subhash Suri, Kevin Verbeek, and Hakan Yildiz. On the most likely convex hull of uncertain points. In *European Symposium on Algorithms*, pages 791–802. Springer, 2013.

On a Connectivity Threshold for Colorings of Random Graphs and Hypergraphs

Michael Anastos
Carnegie Mellon University, Pittsburgh PA 15213, USA
manastos@andrew.cmu.edu

Alan Frieze
Carnegie Mellon University, Pittsburgh PA 15213, USA
alan@random.math.cmu.edu

Abstract

Let $\Omega_q = \Omega_q(H)$ denote the set of proper $[q]$-colorings of the hypergraph H. Let Γ_q be the graph with vertex set Ω_q where two vertices are adjacent iff the corresponding colorings differ in exactly one vertex. We show that if $H = H_{n,m;k}$, $k \geq 2$, the random k-uniform hypergraph with $V = [n]$ and $m = dn/k$ hyperedges then w.h.p. Γ_q is connected if d is sufficiently large and $q \gtrsim (d/\log d)^{1/(k-1)}$. This is optimal to the first order in d. Furthermore, with a few more colors, we find that the diameter of Γ_q is $O(n)$ w.h.p, where the hidden constant depends on d. So, with this choice of d, q, the natural Glauber Dynamics Markov Chain on Ω_q is ergodic w.h.p.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Random Graphs, Colorings, Ergodicity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.36

Category RANDOM

Funding *Alan Frieze*: Research supported in part by NSF grant DMS1661063.

1 Introduction

In this paper, we will discuss a structural property of the set Ω_q of proper $[q]$-colorings of the random hypergraph $H = H_{n,m;k}$, where $m = dn/k$ for some large constant d. Here H has vertex set $V = V(H) = [n]$ and an edge set $E = E(H)$ consisting of m randomly chosen k-sets from $\binom{[n]}{k}$. Note that in the graph case where $k = 2$ we have $H_{n,m;2} = G_{n,m}$. A proper $[q]$-coloring is a map $\sigma : [n] \to [q]$ such that $|\sigma(e)| \geq 2$ for all $e \in E$ i.e. no edge is mono-chromatic. Then let us define $\Gamma_q = \Gamma_q(H)$ to be the graph with vertex set Ω_q and an edge $\{\sigma, \tau\}$ iff $h(\sigma, \tau) = 1$ where $h(\sigma, \tau)$ is the Hamming distance $|\{v \in [n] : \sigma(v) \neq \tau(v)\}|$.

Notation. $f(d) \gtrsim g(d)$ if there exists a function $\varepsilon(d) > 0$ such that $\lim_{d\to\infty} \varepsilon(d) = 0$ and $f(d) \geq (1 + \varepsilon(d))g(d)$ for d large.

Then let

$$\alpha = \left(\frac{(k-1)d}{\log d - 5(k-1)\log\log d}\right)^{\frac{1}{k-1}}, \quad \beta = 3\log^{3k} d. \tag{1}$$

We prove the following.

▶ **Theorem 1.** *Suppose that $k \geq 2$ and $p = \frac{d}{\binom{n-1}{k-1}}$ and $m = \binom{n}{k}p$ and that $d = O(1)$ is sufficiently large. Then*

(i) *If $q \geq \alpha + \beta + 1$ then w.h.p. Γ_q is connected.*

(ii) *If $q \geq \alpha + 2\beta + 1$ then the diameter of Γ_q is $O(\alpha\beta n)$ w.h.p.*

© Michael Anastos and Alan Frieze;
licensed under Creative Commons License CC-BY

Note that Γ_q connected implies that the Glauber Dynamics, which in essence is a random walk on Γ_q, is ergodic. At the moment we only know that Glauber Dynamics is rapidly mixing w.h.p. when $q \geq (1.76\ldots)d$, see Efthymiou, Hayes, Štefankovič and Vigoda [6]. So, it would seem that the connectivity of Γ_q is not likely to be a barrier to randomly sampling colorings of sparse random graphs.

In the Statistical Physics literature the definition of Γ_q may be that colorings σ, τ are connected by an edge in Γ_q whenever $h(\sigma, \tau) = o(n)$. Our theorem holds a fortiori if this is the case.

We note that the lower bound for q is close to where the greedy coloring algorithm succeeds w.h.p. For the case $k = 2$ this follows from Shamir and Upfal [15]. For $k \geq 3$, the authors could not find relevant literature. Nevertheless the claim follows (partially) from the current paper. In particular, Lemmas 13 and 14 show that the greedy coloring algorithm uses at most $\alpha + \beta$ colors. Furthermore, a simple argument based on the size of an independent set selected by the greedy algorithm shows that the number of colors required is close to α.

We should note that it has shown, for $k = 2$ by Molloy [14] and for $k \geq 3$ by Ayre and Greenhill [3], that w.h.p. there is no giant component in Γ_q if $q \lesssim \frac{d}{\log d}$. . It is somewhat surprising therefore that w.h.p. Γ_q jumps very quickly from having no giant to being connected. One might have expected that $q \gtrsim \frac{d}{\log d}$ would simply imply the existence of a giant component. In Physics terminology, this implies a short *non-reconstruction* phase between *uniqueness* and *reconstruction*.

Prior to this paper, it was shown in [13] that w.h.p. $\Gamma_q, q \geq d + 2$ is connected. The diameter of the *reconfiguration graph* $\Gamma_q(G)$ for graphs G has been studied in the graph theory literature, see Bousquet and Perarnau [5] and Feghali [7]. They show that if the maximum sub-graph density of a graph is at most $d - \varepsilon$ and $q \geq d + 1$ then $\Gamma_q(G)$ has polynomial diameter. Using Theorem 1 of [5] we can show a linear bound on the diameter with a small increase in the number of colors, See (ii) of Theorem 1.

Theorem 1 falls into the area of "Structural Properties of Solutions to Random Constraint Satisfaction Problems". This is a growing area with connections to Computer Science and Theoretical Physics. In particular, much of the research on the graph Γ_q has been focussed on the structure near the *colorability threshold*, e.g. Bapst, Coja-Oghlan, Hetterich, Rassman and Vilenchik [4], or the *clustering threshold*, e.g. Achlioptas, Coja-Oghlan and Ricci-Tersenghi [1], Molloy [14] or the *condensation threshold*, e.g. Ayre, Coja-Oghlan and Greenhill [2] or the *rigidity threshold*, e.g. Ayre and Greenhill [3]. Other papers heuristically identify a sequence of phase transitions in the structure of Γ_q, e.g., Krząkala, Montanari, Ricci-Tersenghi, Semerjian and Zdeborová [11], Zdeborová and Krząkala [16] or Gabrié, Dani, Semerjian and Zdeborová [9]. The existence of these transitions has been shown rigorously for some other CSPs. One of the most spectacular examples is due to Ding, Sly and Sun [10] who rigorously showed the existence of a sharp satisfiability threshold for random k-SAT.

Section 3 describes a property (α, β)-greedy-colorability such that if H has this property then $q \geq \alpha + \beta + 1$ implies that Γ_q is connected. Section 4 proves that $H_{n,m;k}, k \geq 2$, is (α, β)-greedy-colorable for α, β defined in (1).

The paper uses some of the ideas from [12] which showed there is a giant component in $\Gamma_q(G_{n,m})$, $m = dn/2$ w.h.p. when $q \geq cd/\log d$ for $c > 3/2$.

2 Outline argument

We show that with the values $\alpha \approx ((k-1)d/\log d)^{1/(k-1)} \gg \beta$ given in (1) then w.h.p. $H = H_{n,m;k}$ has the property that **any** greedy coloring of H will need at most α maximal independent sets before being left with a graph without a β-core. (See Lemma 13.) We call the colorings found in this way, *good greedy colorings* and we refer to this property as

(α,β)-greedy-colorability. Any good (α,β)-coloring uses at most $\alpha+\beta$ colors. It follows from this, basically using the argument from [12], that if $\sigma\in\Omega_q$ and $q\geq\alpha+\beta+1$ then there is a *good* path in Γ_q to some good greedy coloring σ_1.

Suppose now that σ_1,τ_1 are good greedy colorings. If $q\geq\alpha+\beta+1$ then there is a color c that is not used by σ_1. From σ_1 we move to σ_2 by re-coloring vertices colored 1 in σ_1 by c. Then we move from σ_2 to σ_3 by coloring with color 1, all vertices that have color 1 in τ_1. At this point, σ_3 and τ_1 agree on color 1. σ_3 may use more than $\alpha+\beta$ colors and so we move by a good path from σ_3 to a coloring σ_4 that uses at most $\alpha+\beta$ colors and does not change the color of any vertex currently with color 1. Here we use the fact that $H_{n,m;k}$ is (α,β)-greedy-colorable. After this, it is induction that completes the proof.

3 (α,β)-Greedy-Colorability

The degree of a vertex $v\in V$ in a hypergraph $H=(V,E)$ is the number of edges $e\in E$ such that $v\in e$. (For completeness, we will state several things in this short paper that one might think can be taken for granted.)

Let $H=(V,E)$. A β-core of H is a maximal subgraph of H in which every vertex has degree at least β. For every $U\subset V$, if the subgraph $H[U]$ of H induced by U does not have a β-core then there is an ordering $\{u_1,u_2,...,u_{|U|}\}$ of the vertices in U such that every vertex in $u_\ell\in U$ has degree at most $\beta-1$ in the sub-hypergraph induced by $\{u_1,u_2,\dots,u_\ell\}$. (Here, we mean by induced that the edges of $H[U]$ are the edges of H contained entirely in U.)

If a hypergraph H does not have a β-core then we can color it with at most β colors. Let $v_1,v_2,...,v_n$ be an ordering on V where for every i, there are at most $\beta-1$ edges that contain v_i and are contained in $\{v_1,v_2,\dots,v_i\}$. Such an ordering must exist when there is no β-core. We can color the vertices in the order $v_1,v_2,\dots,v_n$ and assign to v_i a color that is not *blocked* by the neighbors that precede it. A color c is blocked for vertex v by vertices $w_1,w_2,\dots,w_{k-1}$ if $e=\{v,w_1,\dots,w_{k-1}\}\in E(H)$ and $w_1,w_2,\dots,w_{k-1}$ have already been given color c.

Next let $V_1,V_2,\dots,V_\alpha$ be a sequence of disjoint independent sets of H such that for each $j\geq1$, V_j is maximal in the sub-hypergraph H_j induced by $V\setminus\bigcup_{1\leq i<j}V_i$. (A set of vertices is independent if it contains no edges.) We say that such a sequence is a *maximally independent sequence of length* α. Note that we allow $V_j=\emptyset$ here, in order to make our sequences of length exactly α.

▶ **Definition 2.** *We say that a hypergraph H is (α,β)-greedy-colorable if there* ***does not exist*** *a maximally independent sequence of length α such that $V\setminus\bigcup_{i\leq\alpha}V_i$ has a β-core.*

The main result of this section is the following.

▶ **Theorem 3.** *Let H be (α,β)-greedy-colorable. If $q\geq\alpha+\beta+1$ then, $\Gamma_q(H)$ is connected. In addition if $q\geq\alpha+2\beta+1$ then, the diameter of $\Gamma_q(H)$ is $O(\alpha\beta n)$.*

Later, in Section 4 we will show that $H_{n,m;k},k\geq2$ is (α,β)-greedy-colorable, for suitable values of m,α,β, viz. the values given in (1).

▶ **Lemma 4.** *Let $H=(V,E)$ be an (α,β)-greedy-colorable hypergraph and $V_1\subseteq V$ be a maximal independent set of V. Set $V'=V\setminus V_1$ and let H' be the subgraph of H induced by V'. Then H' is $(\alpha-1,\beta)$-greedy-colorable.*

Proof. Assume that H' is not $(\alpha-1,\beta)$-greedy-colorable. Then there exists a partition of V' into $V'_1,...,V'_{\alpha-1}$ such that for $j\in[\alpha-1]$, V'_j is a maximal independent set of $V'\setminus\bigcup_{\ell<j}V'_\ell$ and $W'=V'\setminus\bigcup_{\ell\le\alpha-1}V'_\ell$ has a β-core. For $j\in[\alpha-1]$ set $V_{j+1}=V'_j$. Furthermore set $W=V\setminus(\bigcup_{1\le\ell\le\alpha}V_\ell)=V'\setminus(\bigcup_{\ell\le\alpha-1}V'_\ell)=W'$. Then $V_1,...,V_\alpha$ is a maximal independent sequence of length α and W has a β-core which contradicts the fact that H is (α,β)-greedy-colorable. ◀

▶ **Lemma 5.** *Let H be a hypergraph, $\alpha,\beta\ge 0$ and $q\ge\alpha+\beta+1$. Let $W\subseteq V$ be such that the subgraph of H induced by W has no β-core. Furthermore let σ and τ be two colorings of H such that*

- **(i)** *They agree on $V\setminus W$.*
- **(ii)** *They use only α colors on the vertices in $V\setminus W$.*
- **(iii)** *τ uses at most β colors on W that are distinct from the ones it uses on $V\setminus W$.*

Then there exists a path from σ to τ in $\Gamma_q(H)$.

Proof. Without loss of generality we may assume that σ and τ use $[\alpha]$ to color $V\setminus W$. The proof that follows is an adaptation to hypergraphs of the proof in [12] that $\Gamma_q(G)$ is connected when a graph G has no q-core. Because W has no β-core there exists an ordering of its vertices, $v_1,v_2,...,v_r$, such that for $i\in[r]$, v_i has at most $\beta-1$ neighbors in $v_1,v_2,\dots,v_{i-1}$. For $0\le i\le r$ let τ_i be the coloring that agrees with τ on $\{v_1,...,v_i\}$ and with σ on $W\setminus\{v_1,...,v_i\}$. On $V\setminus W$ it agrees with both. Thus $\tau_0=\sigma$ and $\tau_r=\tau$. We note that $\tau_1,\tau_2,\dots,\tau_{r-1}$ may not be proper colorings.

We proceed by induction on i to show that there is a sequence of colorings Σ_i from σ to τ_i such that (i) going from one coloring to the next in Σ_i only re-colors one vertex and (ii) all colorings in the sequence Σ_i are proper for the hypergraph induced by $V\setminus\{v_{i+1},...,v_r\}$. We **do not** claim that the colorings in $\Sigma_i, i<r$ are proper for H. On the other hand, taking $i=r$ we get a sequence of H-proper colorings that starts with σ, ends with τ, such that the consecutive pairs of proper colorings differ on a single vertex. Clearly, such a sequence corresponds to a path from σ to τ in $\Gamma_q(H)$.

The case $i=1$ is trivial as we have assumed that σ,τ agree on $V\setminus W$ and so we can give v_1 the color $\tau(v_1)$. Assume that the assertion is true for $i=\ell\ge 1$ and let $\sigma=\psi_0,\psi_1,\dots,\psi_s=\tau_\ell$ be a sequence of colorings promised by the inductive assertion. Let (w_j,c_j) denote the $(vertex,color)$ change defining the move from ψ_{j-1} to ψ_j. We construct a sequence of colorings of length at most $2s+1$ that yields the assertion for $i=\ell+1$. For $j=1,2,\dots,s$, we will re-color w_j to color c_j, unless there exists a set X such that $X\cup\{w_j\}\in E$ and $\psi_{j-1}(x)=c_j, x\in X\subseteq\{v_1,v_2,\dots,v_{\ell+1}\}$. The fact that ψ_j is a proper coloring of $V\setminus\{v_{\ell+1},...,v_r\}$ implies that $v_{\ell+1}\in X$. Because $v_{\ell+1}$ has at most $\beta-1$ neighbors in $\{v_1,...,v_\ell\}$ and τ only uses colors in $[\alpha]$ to color $V\setminus W$, there exists a color $c'\ne c_j$ for $v_{\ell+1}$ in $[\alpha+\beta+1]\setminus[\alpha]$ which is not blocked by a subset of $\{v_1,v_2,\dots,v_\ell\}$ and is different from its current color. We first re-color $v_{\ell+1}$ to c' and then we re-color w_j to c_j, completing the inductive step. At the very end, i.e. after at most $2s+1$ steps, we give $v_{\ell+1}$ its color in τ. ◀

▶ **Definition 6.** *A coloring with color sets $V_1,V_2,\dots,V_{\alpha+\beta}$ is said to be a* good greedy coloring *if (i) $V_1,V_2,\dots,V_\alpha$ is a maximally independent sequence of length α and (ii) $V\setminus\bigcup_{\ell\le\alpha}V_\ell$ has no β-core.*

We prove Theorem 3 in two steps. In Lemma 7, we show that if $q\ge\alpha+\beta+1$ and H is (α,β)-greedy-colorable then we can reach a good greedy coloring in $\Gamma_q(H)$ starting from any coloring. Then in Lemma 9, we show that if $q\ge\alpha+\beta+1$ then any good greedy coloring τ can be reached in $\Gamma_q(H)$ from any other good greedy coloring σ.

▶ **Lemma 7.** *Let H be an (α,β)-greedy-colorable hypergraph, $q \geq \alpha+\beta+1$ and σ be a $[q]$-coloring of H. Then there exists a good greedy coloring τ of H such that there exists a path in $\Gamma_q(H)$ from σ to τ.*

Proof. We generate the coloring τ as follows. Let $C_1, C_2, \ldots, C_q$ be the color classes of σ. Then let $V_1 \supseteq C_1$ be a maximal independent set containing C_1. In general, having defined $V_1, V_2, \ldots, V_{\ell-1}$ we let $V_{<\ell} = \bigcup_{1\leq i<\ell} V_i$ and then we let V_ℓ be a maximal independent set in $V \setminus V_{<\ell}$ that contains $C_\ell \setminus V_{<\ell}$. Thus $V_1, V_2, \ldots, V_\alpha$ is a maximal independent sequence of length α. We now describe how we transform the coloring σ vertex by vertex into a coloring σ' in which vertices in V_i get color i for $1 \leq i \leq \alpha$. We first re-color the vertices in $V_1 \setminus C_1$ by giving them color 1, one vertex at a time. The coloring stays proper, as V_1 is an independent set. In general, having re-colored $V_1, V_2, \ldots, V_{\ell-1}$ we re-color the vertices in $V_\ell \setminus C_\ell$ with color ℓ. Again, the coloring stays proper, as V_ℓ is an independent set, containing all vertices in C_ℓ that have not been re-colored. We observe that each re-coloring of a vertex v done while turning σ into σ' can be interpreted as moving from a coloring in $\Gamma_q(H)$ to a neighboring coloring.

Let $W = V \setminus \bigcup_{1\leq i\leq\alpha} V_i$. Because H is (α,β)-greedy-colorable, we find that W has no β-core. Because W has no β-core there exists a proper coloring τ' of the subgraph of H induced by W that uses only colors in $[\alpha+\beta] \setminus [\alpha]$. Set τ to be the coloring that agrees with σ' on $V \setminus W$ and with τ' on W.

Lemma 5 implies that there is a path from σ' to τ. Hence there is a path from σ to τ. ◀

▶ Remark 8. In the proof of Lemma 7 we see that each vertex is re-colored at most twice before we apply Lemma 5. Thus this part of the proof yields at most $O(\alpha n)$ vertex recolorings.

▶ **Lemma 9.** *Let H be an (α,β)-greedy-colorable hypergraph, $q \geq \alpha+\beta+1$ and let σ,τ be two good greedy colorings. Then there exists a path from σ to τ in $\Gamma_q(H)$.*

Proof. We proceed by induction on α. For $\alpha = 0$, H is $(0,\beta)$-greedy-colorable and so it does not have a β-core. Thus the base case follows directly from Lemma 5 by taking $W = V$.

Assume that the statement of the Lemma is true for $\alpha = \ell-1$ and let $\alpha = \ell$. There exists a maximal independent sequence $V_1, V_2, \ldots, V_\ell$ of length ℓ such that if $V' = V \setminus \bigcup_{1\leq i\leq\ell} V_i$ then (i) for $i \in [\ell]$, τ assigns the color i to $v \in V_i$ and (ii) τ assigns only colors in $[\ell+\beta] \setminus [\ell]$ to vertices in V'.

Let c be a color not assigned by σ. There is one as $q \geq \ell+\beta+1$. Starting from σ we recolor all vertices that are colored 1 by color c to create a coloring $\bar{\sigma}$. Then we continue from $\bar{\sigma}$ by recoloring all the vertices in V_1 by color 1 and we let σ' be the resulting coloring. Clearly there is a path P_1 from σ to σ' in $\Gamma_q(H)$.

We now set $H_1 = H \setminus V_1$, and set σ'_1, τ_1 to be the restrictions of σ', τ on H_1. Observe that since V_1 is a maximal independent set, Lemma 4 implies that H_1 is $(\ell-1,\beta)$-greedy-colorable and in addition that τ_1 is a good greedy coloring of H_1. Lemma 7 implies that in $\Gamma_{q-1}(H_1)$ there is a path P_2 from σ'_1 to some good greedy coloring σ_1 that uses only $\ell-1+\beta$ colors from $[q] \setminus \{1\}$. The induction hypothesis implies that in $\Gamma_{q-1}(H_1)$ that there is a path P_3 from σ_1 to τ_1.

Color 1 is not used in σ'_1, τ_1 or in any of colorings found in the path P_2, P_3. Thus the path P_2, P_3 corresponds to a path P_4 in $\Gamma_q(H)$ from σ' to τ. Consequently the colorings σ, τ are connected in $\Gamma_q(H)$ by the path $P_1 + P_4$. ◀

Proof of Theorem 3. Let H be (α,β)-greedy-colorable, $q \geq \alpha+\beta+1$, and let σ_1,σ_2 be two colorings of H. Lemma 7 implies that in $\Gamma_q(H)$, there is path P_i from σ_i to a good greedy coloring τ_i for $i=1,2$. Lemma 9 implies that there is a path in $\Gamma_q(H)$ from τ_1 to τ_2.

When $q \geq \alpha+2\beta+1$ Remark 8 shows that while traversing between any pair of proper colorings we perform $O(\alpha n)$ vertex recolorings in the context of Lemma 7. In addition we recolor $\alpha+2$ times with $2\beta+1$ colors a hypergraph with no β-core. Theorem 1 of [5] implies that we need $(\alpha+2)\cdot O(\beta n)$ vertex re-colorings to do this. Thus, there will be $O(\alpha\beta n)$ re-colorings overall and this proves the second part of the theorem. ◀

4 Random Hypergraphs

Theorem 1 follows from

▶ **Lemma 10.** *Let $k \geq 2$ and suppose that $q \geq \alpha+\beta+1$ and that d is sufficiently large. If $p = \frac{d}{\binom{n-1}{k-1}}$ and $m = \binom{n}{k}p$ then w.h.p. $\Gamma_q(H_{n,m;k})$ is connected.*

In the following we will assume for simplicity of notation that $d = O(1)$, so that $O(f(d)/n) = O(1/n)$. We do not know if there is an upper bound needed for the growth rate of d, but we doubt it.

To prove Lemma 10 we use Lemmas 11, 13, 14 (below) in order to deduce that w.h.p. $H_{n,m:k}$ is (α,β)-greedy-colorable. Then we apply Theorem 3. (Lemmas 11 and 14 are hardly new or best possible, but we prove them here for completeness.)

We will do our calculations on the random graph $H_{n,p;k}, p = d/\binom{n-1}{k-1}$ and use the fact for any hypergraph property $\mathcal{P}$, we have (see [8])

$$\mathbf{Pr}(H_{n,m;k} \in \mathcal{P}) \leq O(m^{1/2})\,\mathbf{Pr}(H_{n,p;k} \in \mathcal{P}). \tag{2}$$

▶ **Lemma 11.** *Let $p = \frac{d}{\binom{n-1}{k-1}}$ and $k \geq 2$ and d sufficiently large. Then, w.h.p. $H = H_{n,p;k}$ does not contain an independent set of size $\left(\frac{2k\log d}{(k-1)d}\right)^{\frac{1}{k-1}} n$.*

Proof. Let $u = \left(\frac{2k\log d}{(k-1)d}\right)^{\frac{1}{k-1}} n$. The probability that there exists an independent set of size u in H is bounded by

$$\begin{aligned}
\binom{n}{u}(1-p)^{\binom{u}{k}} &\leq \left(\frac{en}{u}\right)^u \exp\left\{-\frac{d}{\binom{n-1}{k-1}}\cdot\binom{u}{k}\right\}\\
&\leq \left(\frac{en}{u}\right)^u \exp\left\{-\frac{du}{k}\left(\frac{u}{n}\right)^{k-1}\left(1+O\left(\frac{1}{n}\right)\right)\right\}\\
&= \left(e^{k-1}\frac{(k-1)d}{2k\log d}\cdot\exp\left\{-2\log d\left(1+O\left(\frac{1}{n}\right)\right)\right\}\right)^{u/(k-1)}\\
&= \left(\frac{e^{k-1}(k-1)}{2kd\log d}\left(1+O\left(\frac{1}{n}\right)\right)\right)^{u/(k-1)} \qquad (3)\\
&= o(1).
\end{aligned}$$

◀

▶ **Notation 12.** *We let*

$$m_0 = \frac{n}{\alpha} \text{ and } n_0 = 16m_0\log^2 d.$$

Furthermore, for $t \le d$ we let

$$S_t = \left\{(s_1, s_2, ..., s_t) \in \left[\left(\frac{2k\log d}{(k-1)d}\right)^{\frac{1}{k-1}} n\right]^t : \sum_{j=1}^{t} s_i \le \min\{tm_0, n - n_0\}\right\}.$$

▶ **Lemma 13.** *If $k \ge 2$ and d is sufficiently large then, w.h.p.* ***there does not exist*** *$1 \le t \le d$ and disjoint sets $V_1, ..., V_t \subset V$ such that:*

(i) *$V_1, V_2, \dots, V_t$ is a maximal independent sequence of length t in $H = H_{n,p;k}$.*

(ii) *$\big(|V_1|, |V_2|, ..., |V_t|\big) \in S_t$.*

Proof. Fix $t \in [d]$, $(s_1, ..., s_t) \in S_t$ and let $\bar{s} = \frac{1}{t}\sum_{i\in[t]} s_i$. Since $(s_1, ..., s_t) \in S_t$ we have that $\bar{s} \le \frac{1}{t} \cdot tm_0 = m_0$. There are $\binom{n}{s_1, s_2, ..., s_t, n-t\bar{s}}$ ways to pick disjoint sets $V_1, V_2, ..., V_t \subseteq V$ of sizes $s_1, ..., s_t$ respectively. So $V_1, ..., V_t$ satisfy condition (i) of Lemma 13 only if for every $i \in [t]$ and every $v \in V \setminus \bigcup_{j\in[i]} V_j$, there exist $u_1, ..., u_{k-1} \in V_i$ such that $\{u_1, ..., u_{k-1}, v\} \in E(H)$. So, given $V_1, ..., V_t$ the probability that we have (i) is at most

$$p_1 = \prod_{i=1}^{t}(1-(1-p)^{\binom{s_i}{k-1}})^{n-\sum_{j=1}^{i} s_j} \le \exp\left\{-\sum_{i=1}^{t}\left((1-p)^{\binom{s_i}{k-1}}\left(n - \sum_{j=1}^{i} s_j\right)\right)\right\}. \quad (4)$$

Now let $t' = \max\left\{i : \sum_{j\le i} s_j \le n - \frac{n}{\log^2 d}\right\}$ and $s' = \sum_{i=1}^{t'} s_i$ and $\bar{s}' = \frac{s'}{t'}$. We consider 2 cases.

Case 1: $t' \ge (1 - \frac{1}{\log d})t$. Now $t\bar{s} \ge t'\bar{s}'$ and so $\bar{s}' - \bar{s} \le \frac{t-t'}{t}\bar{s}' \le \frac{\bar{s}'}{\log d}$, which implies that $\bar{s}' \le \bar{s}\left(1 - \frac{1}{\log d}\right)^{-1} \le m_0\left(1 + \frac{2}{\log d}\right)$. Then,

$$\begin{aligned}
&\sum_{i=1}^{t}\left((1-p)^{\binom{s_i}{k-1}}\left(n - \sum_{j=1}^{i} s_j\right)\right)\\
&\ge \sum_{i=1}^{t'}\left((1-p)^{\binom{s_i}{k-1}}\left(n - \sum_{j=1}^{i} s_j\right)\right)\\
&\ge \frac{n}{\log^2 d}\sum_{i=1}^{t'}(1-p)^{\binom{s_i}{k-1}} \ge \frac{nt'}{\log^2 d}(1-p)^{\binom{\bar{s}'}{k-1}} \ge \frac{nt}{2\log^2 d}(1-p)^{\binom{m_0\left(1+\frac{2}{\log d}\right)}{k-1}}\\
&\ge \frac{nt}{2\log^2 d}\exp\left\{-(p+p^2)\binom{\left(\frac{(\log d - 5(k-1)\log\log d)}{(k-1)d}\right)^{1/(k-1)}\left(1+\frac{2}{\log d}\right)n}{k-1}\right\}\\
&\ge \frac{nt}{2\log^2 d}\exp\left\{-\frac{\log d - 5(k-1)\log\log d}{k-1}\cdot\left(1+\frac{3(k-1)}{\log d}\right)\right\}\\
&\ge \frac{nt\log^2 d}{d^{1/(k-1)}}.
\end{aligned}$$

Now

$$\binom{n}{s_1, ..., s_t, n-t\bar{s}} \le \binom{n}{\bar{s}, ..., \bar{s}, n-t\bar{s}} \le \prod_{i=1}^{t}\binom{n}{\bar{s}} \le \left(\frac{en}{\bar{s}}\right)^{t\bar{s}} \le \left(\frac{en}{m_0}\right)^{tm_0}.$$

Thus the probability that for some $t \le d$ there exist $V_1, ..., V_t$ satisfying conditions (i), (ii) of Lemma 13 and the condition of Case 1 is bounded by

$$\begin{aligned}
&\sum_{t=1}^{d} \sum_{(s_1,\dots,s_t)\in S_t} \binom{n}{s_1, s_2, .., s_t, n - \sum_{i\in[t]} s_i} p_1 \\
&\le \sum_{t=1}^{d} \sum_{(s_1,\dots,s_t)\in S_t} \left(\frac{en}{m_0}\right)^{tm_0} \exp\left\{-\frac{nt\log^2 d}{d^{1/(k-1)}}\right\} \\
&= \sum_{t=1}^{d} \sum_{(s_1,\dots,s_t)\in S_t} (ea)^{tm_0} \left(\frac{1}{d^{\log d}}\right)^{nt/d^{1/(k-1)}} \\
&\le \sum_{t=1}^{d} n^t \cdot (ea)^{tm_0} \cdot \left(\frac{1}{d^{\log d}}\right)^{nt/d^{1/(k-1)}} \le \sum_{t=1}^{d} n^t \left(\frac{(e\alpha)^{(\log d)^{1/(k-1)}}}{d^{\log d}}\right)^{nt/d^{1/(k-1)}} = o(1).
\end{aligned}$$

At the last equality we used that when d is sufficiently large then the term in the parenthesis is smaller than 1.

Case 2: $t' < \left(1 - \frac{1}{\log d}\right)t$. Thus $t - t' \ge \frac{t}{\log d}$. Observe that from Lemma 11 we can assume that

$$t \ge t' \ge \left(\left(1 - \frac{1}{\log^2 d}\right) \Big/ \left(\frac{2k\log d}{(k-1)d}\right)^{\frac{1}{k-1}}\right) - 1 \ge \frac{1}{4}\left(1 - \frac{1}{\log^2 d}\right)\left(\frac{d}{\log d}\right)^{\frac{1}{k-1}}. \quad (5)$$

For (5) we are using Lemma 11 to argue that we need at least this many independent sets to partition a set of size $n\left(1 - \frac{1}{\log^2 d}\right)$. The -1 comes from the fact that the upper bound in the definition of t' may not be tight.

Thus,

$$\begin{aligned}
u = \frac{1}{t-t'} \sum_{i=t'+1}^{t} s_i &\le \frac{\log d}{t} \cdot n \left(\frac{1}{\log^2 d} + \left(\frac{2k\log d}{(k-1)d}\right)^{\frac{1}{k-1}}\right) \\
&\le 4\left(1 + \frac{2}{\log^2 d}\right)\left(\frac{\log d}{d}\right)^{\frac{1}{k-1}} \cdot \frac{n}{\log d} \quad (6)
\end{aligned}$$

and now with p_1 as defined in (4) we have

$$\begin{aligned}
p_1 &\le \prod_{i=t'+1}^{t} (1 - (1-p)^{\binom{s_i}{k-1}})^{n - \sum_{j=1}^{i} s_j} \le \prod_{i=t'+1}^{t} (1 - (1-p)^{\binom{s_i}{k-1}})^{n_0} \\
&\le \exp\left\{-n_0 \sum_{i=t'+1}^{t} (1-p)^{\binom{s_i}{k-1}}\right\} \le \exp\left\{-n_0(t-t')\left(\prod_{i=t'+1}^{t} (1-p)^{\binom{s_i}{k-1}}\right)^{\frac{1}{t-t'}}\right\} \\
&\le \exp\left\{-n_0(t-t')\exp\left\{-(p+p^2)\left[\sum_{i=t'+1}^{t}\binom{s_i}{k-1}\right]\cdot\frac{1}{t-t'}\right\}\right\}
\end{aligned}$$

$$\begin{aligned}
\ldots &\leq \exp\left\{-n_0(t-t')\exp\left\{-(p+p^2)\binom{u}{k-1}\right\}\right\}\\
&\leq \exp\left\{-n_0(t-t')\exp\left\{-d\left(\frac{u}{n}\right)^{k-1}\right\}\left(1+O\left(\frac{1}{n}\right)\right)\right\}\\
&\leq \exp\left\{-n_0(t-t')\exp\left\{-\frac{4^k\left(1+\frac{2}{\log^2 d}\right)^k}{\log^{k-2} d}\right\}\right\}\\
&\leq e^{-(t-t')n_0/2}.
\end{aligned}$$

Thus the probability that for some $t \leq d$ there exist $V_1, \ldots, V_t$ satisfying conditions (i), (ii) of Lemma 13 and the condition of Case 2 is bounded by

$$\begin{aligned}
P = \sum_{t=1}^{d} \sum_{(s_1,\ldots,s_t)\in S_t} \left[\prod_{i=1}^{t'} \binom{n-\sum_{j=1}^{i-1} s_j}{s_i} \prod_{i=t'+1}^{t} \binom{n-\sum_{j=1}^{i-1} s_j}{s_i}\right] p_1 \\
\leq \sum_{t=1}^{d} \sum_{(s_1,\ldots,s_t)\in S_t} \left(\frac{en}{\bar{s}'}\right)^{t'\bar{s}'} \left(\frac{en}{u}\right)^{(t-t')u} e^{-(t-t')n_0/2}.
\end{aligned}$$

For sufficiently large d, (6) implies $u \leq m_0$ and we also have that $n_0 = 16m_0 \log^2 d$. Therefore

$$\left(\frac{en}{u}\right)^{(t-t')u} e^{-(t-t')n_0/4} \leq \left(\frac{en}{m_0}\right)^{(t-t')m_0} e^{-4(t-t')m_0 \log^2 d} \leq e^{-3(t-t')m_0 \log^2 d} \leq e^{-3tm_0 \log d}.$$

Furthermore, Lemma 11 implies that $\bar{s}' \leq \left(\frac{2k\log d}{(k-1)d}\right)^{\frac{1}{k-1}} n \leq 3m_0$. Thus

$$\begin{aligned}
\left(\frac{en}{\bar{s}'}\right)^{t'\bar{s}'} e^{-(t-t')n_0/4} \leq \left(\frac{en}{3m_0}\right)^{3tm_0} e^{-4(t-t')m_0 \log^2 d} \leq \\
\left(\frac{en}{3m_0}\right)^{3tm_0} e^{-4tm_0 \log d} \leq e^{-tm_0 \log d}.
\end{aligned}$$

So,

$$P \leq dn^d e^{-4tm_0 \log d} = dn^d e^{-4t\alpha n \log d} = o(1). \qquad \blacktriangleleft$$

▶ **Lemma 14.** *If $k \geq 2$ and d is sufficiently large then w.h.p. every set $S \subset V$ of size at most n_0 spans fewer than $3|S|\log^{3k} d$ edges in H. Hence no subset of size at most n_0 contains a $3\log^{3k} d$ core.*

Proof. Let $L = 3\log^{3k} d$. The probability that there exists $S \subset V$ of size at most n_0 that spans at least $t = L|S|$ edges is bounded by

$$\begin{aligned}
\sum_{s=1}^{n_0} \binom{n}{s}\binom{\binom{s}{k}}{t} p^t \leq \sum_{s=1}^{n_0} \left(\left(\frac{en}{s}\right)^{s/t} \cdot \frac{e\binom{s}{k}}{t} \cdot \frac{d}{\binom{n-1}{k-1}}\right)^t \leq \sum_{s=1}^{n_0} \left(\left(\frac{en}{s}\right)^{1/L} \frac{eds}{t} \left(\frac{s}{n}\right)^{k-1}\right)^t \\
= \sum_{s=1}^{n_0} \left(\left(\frac{s}{n}\right)^{k-1-1/L} \frac{e^{1+1/L}d}{L}\right)^t = o(1). \qquad \blacktriangleleft
\end{aligned}$$

Proof of Theorem 1. Let α, β be as in (1). We argue next that the properties given by Lemmas 11, 13 and 14 imply that $H_{n,p;k}$ is (α, β)-greedy-colorable for d sufficiently large. That is for any sequence of sets $V_1, V_2, \ldots, V_\alpha$ such that V_i is maximally independent in $[n] \setminus \bigcup_{j<i} V_j$ for $j \leq \alpha$ we have that $[n] \setminus \bigcup_{i \leq \alpha} V_i$ does not have a β-core. Lemma 10 then follows directly from (2) and Theorem 3.

Consider such a sequence of sets $V_1, V_2, \ldots, V_\alpha$. It follows from Lemma 13 that because $\alpha m_0 = n$, we must have $\sum_{i=1}^{\alpha} |V_i| \geq n - n_0$ and then Lemma 14 implies that $[n] \setminus \bigcup_{i \leq \alpha} V_i$ does not have a β-core. ◀

References

1 D. Achlioptas, A. Coja-Oghlan, and F. Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. *Random Structures and Algorithms*, 38:251–268, 2010.

2 P. Ayre, A. Coja-Oghlan, and C. Greenhill. Hypergraph coloring up to condensation. *arxiv:1508.01841*, 2019. `arXiv:1508.01841`.

3 P. Ayre and C. Greenhill. Rigid colourings of hypergraphs and contiguity. *arxiv:1812.03195*, 2019. `arXiv:1812.03195`.

4 V. Bapst, A. Coja-Oghlan, S. Hetterich, F. Rassmann, and D. Vilenchik. The condensation phase transition in random graph coloring. *Communications in Mathematical Physics*, 341:543–606, 2016.

5 N. Bousquet and G. Perarnau. Fast recoloring of sparse graphs. *European Journal of Combinatorics*, 52:1–11, 2016.

6 C. Efthymiou, T. Hayes, D. Štefankovič, and E. Vigoda. Sampling Colorings of Sparse random Graphs. In *SODA*, 2018.

7 C. Feghali. Paths between colourings of sparse graphs. *European Journal of Combinatorics*, 75:169–171, 2019.

8 A.M. Frieze and M. Karoński. *Introduction to Random Graphs*. Cambridge University Press, 2015.

9 M. Gabrié, V. Dani, G. Semerjian, and L. Zdeborová. Phase transitions in the q-coloring of random hypergraphs. *Journal of Physics A: Mathematical Theory*, 50, 2017.

10 A. Sly J. Ding and N. Sun. Proof of the satisfiability conjecture for large k. *arxiv:1411.0650*, 2019. `arXiv:1411.0650`.

11 F. Krzakąla, A. Montanari, F. Ricci-Tersenghi, G. Semerijian, and L. Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceedings of the National Academy of Sciences*, 104:10318–10323, 2007.

12 Anastos M., A.M. Frieze, and W. Pegden. Constraining the clustering transition for colorings of sparse random graphs. *Electronic Journal of Combinatorics*, 2018.

13 and A. Flaxman M. Dyer, A.M. Frieze, and E. Vigoda. Randomly coloring sparse random graphs with fewer colors than the maximum degree. *Random Structures and Algorithms*, 29:450–465, 2006.

14 M. Molloy. The freezing threshold for k-colourings of a random graph. In *STOC*, 2012.

15 E. Shamir and E. Upfal. Sequential and Distributed Graph Coloring Algorithms with Performance Analysis in Random Graph Spaces. *Journal of Algorithms*, 5:488–501, 1984.

16 L. Zdeborová and F. Krzakąla. Phase Transitions in the Coloring of Random Graphs. *Physics Review E*, 76, 2007.

Slow Mixing of Glauber Dynamics for the Six-Vertex Model in the Ordered Phases

Matthew Fahrbach
School of Computer Science, Georgia Institute of Technology, Atlanta, Georgia, USA
matthew.fahrbach@gatech.edu

Dana Randall
School of Computer Science, Georgia Institute of Technology, Atlanta, Georgia, USA
randall@cc.gatech.edu

Abstract

The six-vertex model in statistical physics is a weighted generalization of the ice model on $\mathbb{Z}^2$ (i.e., Eulerian orientations) and the zero-temperature three-state Potts model (i.e., proper three-colorings). The phase diagram of the model represents its physical properties and suggests where local Markov chains will be efficient. In this paper, we analyze the mixing time of Glauber dynamics for the six-vertex model in the ordered phases. Specifically, we show that for all Boltzmann weights in the *ferroelectric phase*, there exist boundary conditions such that local Markov chains require exponential time to converge to equilibrium. This is the first rigorous result bounding the mixing time of Glauber dynamics in the ferroelectric phase. Our analysis demonstrates a fundamental connection between correlated random walks and the dynamics of intersecting lattice path models (or routings). We analyze the Glauber dynamics for the six-vertex model with free boundary conditions in the *antiferroelectric phase* and significantly extend the region for which local Markov chains are known to be slow mixing. This result relies on a Peierls argument and novel properties of weighted non-backtracking walks.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains

Keywords and phrases Correlated random walk, Markov chain Monte Carlo, Six-vertex model

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.37

Category RANDOM

Related Version A full version of the paper is available at `https://arxiv.org/abs/1904.01495`.

Funding *Matthew Fahrbach*: Supported in part by an NSF Graduate Research Fellowship under grant DGE-1650044.
Dana Randall: Supported in part by NSF grants CCF-1637031 and CCF-1733812.

1 Introduction

The *six-vertex model* was first introduced by Pauling in 1935 [33] to study the thermodynamics of crystalline solids with ferroelectric properties, and has since become one of the most compelling models in statistical mechanics. The prototypical instance of the model is the hydrogen-bonding pattern of two-dimensional ice – when water freezes, each oxygen atom must be surrounded by four hydrogen atoms such that two of the hydrogen atoms bond covalently with the oxygen atom and two are farther away. The state space of the six-vertex model consists of orientations of the edges in a finite region of the Cartesian lattice where every internal vertex has two incoming edges and two outgoing edges, also represented as Eulerian orientations of the underlying lattice graph. The model is most often studied on the $n \times n$ square lattice $\Lambda_n \subseteq \mathbb{Z}^2$ with $4n$ additional edges so that each internal vertex has degree 4. There are six possible edge orientations incident to a vertex (see Figure 1).


© Matthew Fahrbach and Dana Randall;
licensed under Creative Commons License CC-BY

We assign Boltzmann weights $w_1, w_2, w_3, w_4, w_5, w_6 \in \mathbb{R}_{>0}$ to the six vertex types and define the partition function as $Z = \sum_{x \in \Omega} \prod_{i=1}^{6} w_i^{n_i(x)}$, where Ω is the set of Eulerian orientations of Λ_n and $n_i(x)$ is the number of type-i vertices in the configuration x.

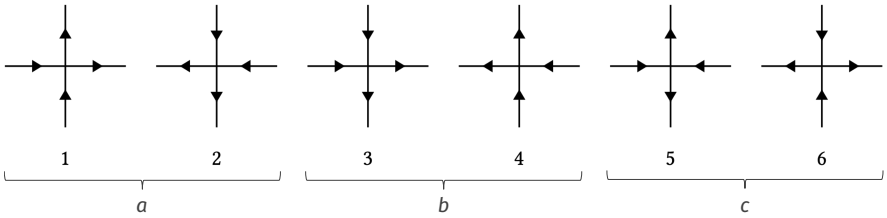
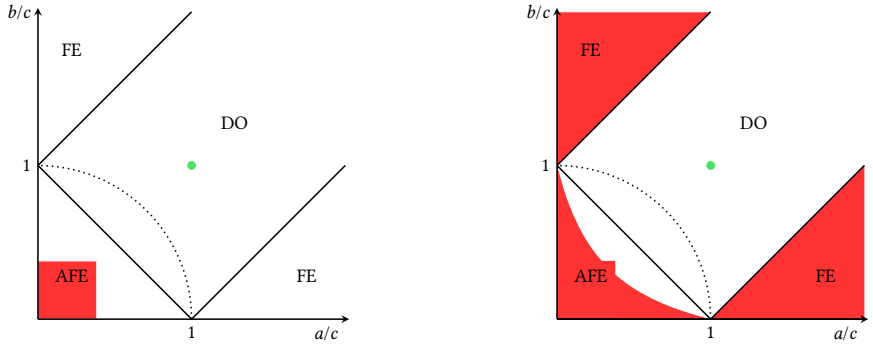


Figure 1 The valid edge orientations for internal vertices in the six-vertex model.

In 1967, Lieb discovered exact solutions to the six-vertex model with periodic boundary conditions for three different parameter regimes [25, 26, 27]. In particular, he famously showed that if all six vertex weights are $w_i = 1$, the energy per vertex is $\lim_{n\to\infty} Z^{1/n^2} = (4/3)^{3/2} = 1.5396007...$ (known as Lieb's square ice constant). His results were immediately generalized to allow for all parameter settings and external electric fields [38, 40]. An equivalence between periodic and free boundary conditions in the limit was established soon after [7], and since then the primary object of study has been the six-vertex model subject to *domain wall boundary conditions*, where the lower and upper boundary edges point into the square and the left and right boundary edges point outwards [20, 22, 6, 3, 4, 5]. There have been several surprisingly profound connections to enumerative combinatorics in this line of work. For instance, Zeilberger gave a sophisticated computer-assisted proof of the *alternating sign matrix conjecture* in 1995 [41]. A year later, Kuperberg [23] produced an elegant and significantly shorter proof using analysis of the partition function of the six-vertex model with domain wall boundary conditions. Other connections of the model to combinatorics and probability include tilings of the Aztec diamond and the arctic circle theorem [11, 14], sampling lozenge tilings [29, 39, 2], and enumerating 3-colorings of lattice graphs [36, 10].

While there has been extraordinary progress in understanding properties of the six-vertex model with periodic or domain wall boundary conditions, remarkably less is known when the model is subject to arbitrary boundary conditions. Sampling configurations using Markov chain Monte Carlo (MCMC) algorithms has been one of the primary means for discovering more general mathematical and physical properties of the six-vertex model [1, 31, 30, 21], and empirically the model is very sensitive to boundary conditions. Numerical studies have often observed slow convergence of local MCMC algorithms under certain parameter settings. For example, according to [30], "it must be stressed that the Metropolis algorithm might be impractical in the antiferromagnetic phase, where the system may be unable to thermalize." However, there are very few rigorous results for natural Markov chains and the computational complexity of sampling from the Boltzmann distribution for various weights and boundary conditions. This motivates our study of Glauber dynamics, the most widely used MCMC sampling algorithm, for the six-vertex model in the *ferroelectric* and *antiferroelectric phases*.

At first glance, there are six degrees of freedom in the model. However, this conveniently reduces to a two-parameter family due to invariants and standard physical assumptions that relate pairs of vertex types. To see this, it is useful to map configurations of the six-vertex model to sets of intersecting lattice paths by erasing all of the edges that are directed south or west and keeping the others [29]. Using this "routing interpretation," it is simple to see that the number of type-5 and type-6 vertices must closely correlated. In addition to revealing invariants, the lattice path representation of configurations turns out to be exceptionally



(a) (b)

Figure 2 Phase diagram of the six-vertex model with (a) previously known and (b) our current slowly mixing regions colored in red. Glauber dynamics is conjectured to be rapidly mixing for the entire disordered phase but has only been shown for the uniform distribution indicated by the green point $(1, 1)$ in both figures.

useful for analyzing Glauber dynamics. Moreover, the total weight of a configuration should remain unchanged if all the edge directions are reversed in the absence of an external electric field, so we let $w_1 = w_2 = a$, $w_3 = w_4 = b$, and $w_5 = w_6 = c$. This complementary invariance is known as the *zero field assumption*, and it is often convenient to exploit the conservation laws of the model [4] to reparameterize the system so that $w_1 = a^2$ and $w_2 = 1$. This allows us to ignore empty sites and focus solely on weighted lattice paths. Furthermore, since our goal is to sample configurations from the Boltzmann distribution, we can normalize the partition function by a factor of c^{-n^2} and consider the weights $(a/c, b/c, 1)$ instead of (a, b, c). We collectively refer to these properties as the invariance of the Gibbs measure for the six-vertex model.

The phase diagram of the six-vertex model represents physical properties of the system and is partitioned into three regions: the *disordered* (DO) phase, the *ferroelectric* (FE) phase, and the *antiferroelectric* (AFE) phase. To establish these regions, we consider the parameter

$$\Delta = \frac{a^2 + b^2 - c^2}{2ab}.$$

The disordered phase is the set of parameters $(a, b, c) \in \mathbb{R}^3_{>0}$ that satisfy $|\Delta| < 1$, and Glauber dynamics is expected to be rapidly mixing in this region because there are no long-range correlations in the system. The ferroelectric phase is defined by $\Delta > 1$, or equivalently when $a > b + c$ or $b > a + c$. We show in this paper that Glauber dynamics can be slow mixing at any point in this region (Figure 2b). The antiferroelectric phase is defined by $\Delta < -1$, or equivalently when $a + b < c$, and our second result significantly extends the antiferroelectric subregion for which Glauber dynamics is known to be slow mixing. The phase diagram is symmetric over the main positive diagonal, which follows from the fact that a and b are interchangeable under the automorphism that rotates each of the six vertex types by ninety degrees clockwise. Under the zero field assumption, this is equivalent to rotating the entire model, so we can assume without loss of generality that if a mixing result holds for one point in the phase diagram, it also holds at the point reflected over the main diagonal.

Cai, Liu, and Lu [9] recently provided strong evidence supporting conjectures about the approximability of the six-vertex model. In particular, they designed a *fully randomized approximation scheme* (FPRAS) for a subregion of the disordered phase that works for all 4-regular graphs via the winding framework for Holant problems [32, 19]. They also showed

that there cannot exist an FPRAS for 4-regular graphs in the ferroelectric or antiferroelectric phases unless $\mathbf{RP} = \mathbf{NP}$. We note that their hardness result uses nonplanar gadgets and the larger class of 4-regular graphs, so it does not reveal anything about the complexity of Glauber dynamics for the six-vertex model on regions of $\mathbb{Z}^2$. A dichotomy theorem for the (exact) computability of the partition function of the six-vertex model on 4-regular graphs was also recently proven in [8]. As for the positive results, Luby, Randall, and Sinclair [29] proved rapid mixing of a Markov chain that leads to a fully polynomial almost uniform sampler for Eulerian orientations on any region of the Cartesian lattice with fixed boundaries (i.e., the unweighted case when $a/c = b/c = 1$). Randall and Tetali [36] then used a comparison technique to argue that Glauber dynamics for Eulerian orientations on lattice graphs is rapidly mixing by relating this Markov chain to the Luby-Randall-Sinclair chain. Goldberg, Martin, and Paterson [16] extended their approach to show that Glauber dynamics is rapidly mixing on rectangular lattice regions with free boundary conditions.

Liu [28] recently gave the first rigorous result that Glauber dynamics is slowly mixing in a subregion of an ordered phase by showing that local Markov chains require exponential time to converge in the antiferroelectric subregion defined by $\max(a, b) < c/\mu$, where $\mu = 2.6381585...$ is the connective constant for self-avoiding walks on the square lattice (Figure 2a). He also showed that the directed loop algorithm mixes slowly in the same antiferroelectric subregion and for all of the ferroelectric region, but this has no bearing on the efficiency of Glauber dynamics in the ferroelectric region. We note that the partition function is exactly computable for all boundary conditions at the free-fermion point when $\Delta = 0$, or equivalently $a^2 + b^2 = c^2$, via a reduction to domino tilings and a Pfaffian computation [14]. There is strong evidence that exact counting is unlikely anywhere else for arbitrary boundary conditions [8].

1.1 Main Results

In this paper we show that there exist boundary conditions for which Glauber dynamics mixes slowly for the six-vertex model in the ferroelectric and antiferroelectric phases. We start by proving that there are boundary conditions that cause Glauber dynamics to be slow for all Boltzmann weights that lie in the ferroelectric region of the phase diagram, where the mixing time is exponential in the number of vertices in the lattice. This is the first rigorous result for the mixing time of Glauber dynamics in the ferroelectric phase and it gives a complete characterization.

▶ **Theorem 1** (Ferroelectric phase). *For any $(a, b, c) \in \mathbb{R}^3_{>0}$ such that $a > b + c$ or $b > a + c$, there exist boundary conditions for which Glauber dynamics mixes exponentially slowly on Λ_n.*

We note that our approach naturally breaks down at the critical line in a way that reveals a trade-off between the energy and entropy of the system. Additionally, our analysis suggests an underlying combinatorial interpretation for the phase transition between the ferroelectric and disordered phases in terms of the adherence strength of intersecting lattice paths and the momentum parameter of correlated random walks.

Our second mixing result builds on the topological obstruction framework developed in [35] to show that Glauber dynamics with free boundary conditions mixes slowly in most of the antiferroelectric region. Specifically, we generalize the recent antiferroelectric mixing result in [28] with a Peierls argument that uses multivariate generating functions for weighted non-backtracking walks instead of the connectivity constant for (unweighted) self-avoiding walks to better account for the discrepancies in Boltzmann weights.

▶ **Theorem 2** (Antiferroelectric phase). *For any $(a, b, c) \in \mathbb{R}^3_{>0}$ such that $ac + bc + 3ab < c^2$, Glauber dynamics mixes exponentially slowly on Λ_n with free boundary conditions.*

We illustrate the new regions for which Glauber dynamics can be slowly mixing in Figure 2. Observe that our antiferroelectric subregion significantly extends Liu's and pushes towards the conjectured threshold.

1.2 Techniques

We take significantly different approaches for our analysis of the ferroelectric and antiferroelectric phases. In the ferroelectric phase, where $a > b + c$ and type-a vertices are preferred to type-b and type-c vertices, we construct boundary conditions that induce polynomially-many paths separated by a critical distance that allows all of the paths to (1) behave independently and (2) simultaneously intersect with their neighbors maximally. (This analysis also covers the case $b > a + c$ by a standard invariant that shows symmetry in the phase diagram over the line $y = x$.) From here, we analyze the dynamics of a single path in isolation as an escape probability, which eventually allows us to bound the conductance of the Markov chain. The dynamics of a single lattice path are equivalent to a those of a *correlated random walk*. In Appendix A we present a new tail inequality for correlated random walks that accurately bounds the probability of large deviations from the starting position. We note that decomposing the dynamics of lattice models into one-dimensional random walks has recently been shown to achieve nearly tight bounds for escape probabilities in a different setting [12].

One of the key technical contributions in this paper is our analysis of the tail behavior of correlated random walks in Appendix A. While there is a simple combinatorial expression for the position of a correlated random walk written as a sum of marginals, it is not immediately useful for bounding the displacement from the origin. To achieve an exponentially small tail bound for these walks, we first construct a smooth function that tightly upper bounds the marginals and then optimize this function to analyze the asymptotics of the log of the maximum marginal. Once we obtain an asymptotic equality for the maximum marginal, we can upper bound the deviation of a correlated random walk, and hence the deviation of a lattice path in a configuration. Ultimately, this allows us to show that there exists a balanced cut in the state space that has an exponentially small escape probability, which implies that the Glauber dynamics are slowly mixing.

In the antiferroelectric phase, on the other hand, the Boltzmann weights satisfy $a + b < c$ so type-c vertices are preferred. It follows that there are two (arrow-reversal) symmetric ground states of maximum probability containing only type-c vertices. To move between configurations that agree predominantly with different ground states, the Markov chain must pass through configurations with a large number of type-a or type-b vertices. Using the idea of *fault lines* introduced in [35], we use self-avoiding walks to characterize such configurations and construct a cut set with exponentially small probability mass that separates the ground states. Liu [28] follows this Peierls argument approach and bounds the weight of the cut by separately considering the minimum energy gain of the corresponding inverse map and the number of preimages (i.e., the entropy). Instead, we directly bound the free energy (rather than as a product of the upper bounds for the energy and entropy terms) and are able to show slow mixing for a much larger region of the phase diagram. Our key observation for accurately bounding the free energy is that when a fault line changes direction, the vertices along it switch from type-a to type-b or vice versa. Therefore, we introduce the notion of *weighted non-backtracking walks* and solve their multivariate generating function by diagonalizing a system of linear recurrences to exactly account for disparities between the weights of a and b along fault lines.

2 Preliminaries

We start by reviewing some necessary background on Markov chains, Glauber dynamics, and correlated random walks.

2.1 Markov Chains and Mixing Times

Let $\mathcal{M}$ be an ergodic, reversible Markov chain with finite state space Ω, transition probability matrix P, and stationary distribution π. The t-step transition probability from states x to y is denoted as $P^t(x, y)$. The total variation distance between the probability distributions μ and ν on Ω is

$$\|\mu - \nu\|_{\text{TV}} = \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \nu(x)|.$$

The *mixing time* of $\mathcal{M}$ is $\tau(1/4) = \min\{t \in \mathbb{Z}_{\geq 0} : \max_{x \in \Omega} \|P^t(x, \cdot) - \pi\|_{\text{TV}} \leq 1/4\}$. We say that $\mathcal{M}$ is rapidly mixing if its mixing time is $O(\text{poly}(n))$, where n is the size of each configuration in the state space. Similarly, we say that $\mathcal{M}$ is slow mixing if its mixing time is $\Omega(\exp(n^c))$ for some constant $c > 0$.

The mixing time of a Markov chain is characterized by its *conductance* (up to polynomial factors). The conductance of a nonempty set $S \subseteq \Omega$ is

$$\Phi(S) = \frac{\sum_{x \in S, y \notin S} \pi(x) P(x, y)}{\pi(S)},$$

and the conductance of the Markov chain is $\Phi^* = \min_{S \subseteq \Omega : 0 < \pi(S) \leq 1/2} \Phi(S)$. It is often useful to view the conductance of a set as an escape probability – starting from stationarity and conditioned on being in S, the conductance $\Phi(S)$ is the probability that $\mathcal{M}$ leaves S in one step.

▶ **Theorem 3** ([24]). *For an ergodic, reversible Markov chain with conductance Φ^*, we have $\tau(1/4) \geq 1/(4\Phi^*)$.*

To show that a Markov chain is slow mixing, it suffices to show that the conductance is exponentially small.

In this paper we the study single-site *Glauber dynamics* for the six-vertex model. This Markov chain makes local moves by (1) choosing an internal cell of the lattice uniformly at random and (2) reversing the orientations of the edges that bound the chosen cell if they form a cycle. In the lattice path interpretation of the model, these dynamics correspond to the mountain-valley Markov chain that flips corners. Transitions between states are made according to the Metropolis-Hastings acceptance probability so that the Markov chain converges to the desired stationary distribution.

2.2 Correlated Random Walks

A key tool in our analysis for the ferroelectric phase are correlated random walks, which generalize simple symmetric random walks by accounting for momentum. A *one-dimensional correlated random walk* with momentum parameter $p \in [0, 1]$ starts at the origin and is defined as follows. Let X_1 be a uniform random variable with support $\{-1, 1\}$. For all subsequent steps $i \geq 2$, the direction of the process is correlated with the direction of the previous step and satisfies

$$X_{i+1} = \begin{cases} X_i & \text{with probability } p, \\ -X_i & \text{with probability } 1 - p. \end{cases}$$

We denote the position of the walk at time t by $S_t = \sum_{i=1}^{t} X_i$. It will often be useful to make the change of variables $p = \mu/(1+\mu)$ when analyzing the six-vertex model. In many cases this also leads to cleaner expressions. We use the following probability density function (PDF) for the position of a correlated random walk to develop a new tail inequality (Lemma 8) that holds for all values of p.

▶ **Lemma 4** ([18]). *For any $n \geq 1$ and $m \geq 0$, the PDF of a correlated random walk is*

$$\Pr(S_{2n} = 2m) = \begin{cases} \frac{1}{2}p^{2n-1} & \text{if } 2m = 2n, \\ \sum_{k=1}^{n-m} \binom{n+m-1}{k-1}\binom{n-m-1}{k-1}(1-p)^{2k-1}p^{2n-1-2k}\left(\frac{n(1-p)+k(2p-1)}{k}\right) & \text{if } 2m < 2n. \end{cases}$$

3 Slow Mixing in the Ferroelectric Phase

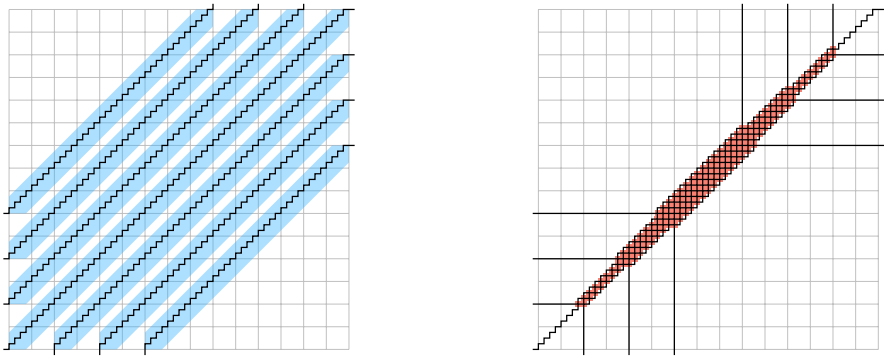
We start with the ferroelectric phase where $a > b+c$ or $b > a+c$, and we give a conductance-based argument to show that Glauber dynamics can be slowly mixing in the entire ferroelectric region. Specifically, we show that there exist boundary conditions that induce an exponentially small, asymmetric bottleneck in the state space, revealing a natural trade-off between the energy and entropy in the system. Viewing the six-vertex model in the intersecting lattice interpretation suggests how to plant polynomially-many paths in the grid that can (1) be analyzed independently, while (2) being capable of intersecting maximally. This path independence makes our analysis tractable and allows us to interpret the dynamics of a path as a correlated random walk, for which we develop an exponentially small tail bound in Appendix A. Since escape probabilities govern mixing times [34], we show how to relate the expected maximum deviation of a correlated walk to the conductance of the Markov chain to prove slow mixing. In addition to showing slow mixing up to the conjectured threshold, a surprising feature of our argument is that it potentially gives a combinatorial explanation for the phase transition from the ferroelectric to disordered phase. In particular, Lemma 9 demonstrates how the parameters of the model delicately balance the probability mass of the Markov chain.

Next, we exploit the invariance of the Gibbs measure and the lattice path interpretation of the six-vertex model to conveniently reparameterize the Boltzmann weights. Specifically, we let $w_1 = \lambda^2$ and $w_2 = 1$ so that we can ignore empty sites. Note that $a = \sqrt{w_1 w_2} = \lambda$. We also let $b = w_2 = w_3 = \mu$ and $c = w_5 = w_6 = 1$ so that the weight of a configuration only comes from straight segments and intersections of neighboring lattice paths.

3.1 Constructing the Boundary Conditions and Cut

We begin with a few colloquial definitions for lattice paths that allow us to easily construct the boundary conditions and make arguments about the conductance of the Markov chain. We call a $2n$-step, north-east lattice path γ starting from $(0,0)$ a *path of length* $2n$, and if the path ends at (n,n) we describe it as *tethered*. If $\gamma = ((0,0),(x_1,y_1),(x_2,y_2),\ldots,(x_{2n},y_{2n}))$, we define the *deviation* of γ to be $\max_{i=0..2n}\|(x_i,y_i) - (i/2,i/2)\|_1$. Geometrically, path deviation captures the (normalized) maximum perpendicular distance of the path to the line $y = x$. We refer to vertices (x_i,y_i) along the path as *corners* or *straights* depending on whether or not the path turned. If two paths intersect at a vertex we call this site a *cross*. Note that this classifies all vertex types in the six-vertex model.

We consider the following *independent paths boundary condition* for an $n \times n$ six-vertex model for the rest of the section. To construct this boundary condition, we consider its lattice path interpretation. First, place a tethered path γ_0 that enters $(0,0)$ horizontally and exits (n,n) horizontally. Next, place $2\ell = \lfloor n^{1/8} \rfloor$ translated tethered paths of varying



(a) (b)

Figure 3 Examples of states with the independent paths boundary condition: (a) is a state in S with the deviation bounds highlighted and (b) is the ground state in the ferroelectric phase.

length above and below the main diagonal, each separated from its neighbors by distance $d = \lfloor 32n^{3/4} \rfloor$. Specifically, the paths $\gamma_1, \gamma_2, \ldots, \gamma_\ell$ below the main diagonal begin at the vertices $(d, 0), (2d, 0), \ldots, (\ell d, 0)$ and end at the vertices $(n, n-d), (n, n-2d), \ldots, (n, n-\ell d)$, respectively. The paths $\gamma_{-1}, \gamma_{-2}, \ldots, \gamma_{-\ell}$ above the main diagonal begin at $(0, d), (0, 2d), \ldots, (0, \ell d)$ and end at $(n-d, n), (n-2d, n), \ldots, (n-\ell d, n)$. The deviation of a translated tethered path is the deviation of the same path starting at $(0, 0)$. To complete the boundary condition, we force the paths below the main diagonal to enter vertically and exit horizontally. Symmetrically, we force the paths above the main diagonal to enter horizontally and exit vertically. See Figure 3a for an illustration of the construction when all paths have small deviation.

Next, we construct an asymmetric cut in the state space induced by this boundary condition in terms of its internal lattice paths. In particular, we analyze a set S of configurations such that every path in a configuration has small deviation. Formally, we let

$$S \stackrel{\text{def}}{=} \left\{ x \in \Omega : \text{the deviation of each path in } x \text{ is less than } 8n^{3/4} \right\}.$$

Observe that by our choice of separation distance $d = \lfloor 32n^{3/4} \rfloor$ and the deviation limit for S, no paths in any configuration of S intersect. It follows that the partition function for S factors into a product of $2\ell + 1$ partition functions, one for each path with bounded deviation. This intuition is useful when analyzing the conductance $\Phi(S)$ as an escape probability from stationarity.

3.2 Lattice Paths as Correlated Random Walks

Now we consider weighting the internal paths according to the six-vertex model. The main result in this subsection is that random tethered paths are exponentially unlikely to deviate past $\omega(n^{1/2})$, even if drawn from a Boltzmann distribution that favors straights (Lemma 5). Let $\Gamma(\mu, n)$ denote the distribution over tethered paths of length $2n$ such that

$$\Pr(\gamma) \propto \mu^{(\#\text{ of straights in } \gamma)}.$$

▶ **Lemma 5.** *Let $\mu, \varepsilon > 0$ and $m = o(n)$. For n sufficiently large and $\gamma \sim \Gamma(\mu, n)$, we have* $\Pr(\gamma \text{ deviates by at least } 2m) \leq e^{-(1-\varepsilon)\frac{m^2}{\mu n}}$.

We defer the proof of Lemma 5 to the full version of the paper [13]. Instead, we sketch its key ideas to demonstrate the connection between biased tethered paths and correlated random walks, and to show how the supporting lemmas interact.

First, observe that there is a natural measure-preserving bijection between biased tethered paths of length $2n$ and correlated random walks of length $2n$ that return to the origin. Concretely, for a correlated random walk $(S_0, S_1, \ldots, S_{2n})$ parameterized by $p = \mu/(1+\mu)$,

$$\Pr(\gamma \text{ deviates by at least } 2m) = \Pr\Big(\max_{i=0..2n} |S_i| \geq 2m \;\Big|\; S_{2n} = 0\Big). \tag{1}$$

Now we present an asymptotic equality that generalizes the return probability of simple symmetric random walks. This allows us to relax the condition in (1) that a correlated random walk returns to the origin, and instead we bound $\Pr(\max_{i=0..2n} |S_i| \geq 2m)$ at the expense of an additional polynomial factor.

▶ **Lemma 6** ([15]). *For any constant $\mu > 0$, the return probability of a correlated random walk is $\Pr(S_{2n} = 0) \sim 1/\sqrt{\mu\pi n}$.*

Another result needed to prove Lemma 5 is that the PDF for correlated random walks is unimodal.

▶ **Lemma 7.** *For any momentum parameter $p \in (0,1)$ and n sufficiently large, the probability of the position of a correlated random walk is unimodal. Concretely, for $m \in \{0, 1, \ldots, n-1\}$, we have $\Pr(S_{2n} = 2m) \geq \Pr(S_{2n} = 2(m+1))$.*

Last, we give an upper bound for the position of a correlated random walk. We fully develop this inequality in Appendix A by analyzing the asymptotic behavior of the PDF in Lemma 4. Observe that Lemma 8 demonstrates exactly how the tail behavior of simple symmetric random walks generalizes to correlated random walks as a function of μ.

▶ **Lemma 8.** *Let $\mu, \varepsilon > 0$ and $m = o(n)$. For n sufficiently large, a correlated random walk satisfies $\Pr(S_{2n} = 2m) \leq e^{-(1-\varepsilon)\frac{m^2}{\mu n}}$.*

To complete the proof sketch of Lemma 5, we start by using Lemma 6 to relax the conditional probability. It follows from Lemma 7 and union bounds that

$$\Pr\Big(\max_{i=0..2n} |S_i| \geq 2m \;\Big|\; S_{2n} = 0\Big) \leq 2\sqrt{\mu\pi n} \cdot 2n^2 \cdot \Pr(S_{2n} = 2m). \tag{2}$$

Applying Lemma 8 to (2) with a smaller error completes the proof. See [13] for more details.

3.3 Bounding the Conductance and Mixing Time

Next, we bound the conductance of the Markov chain by viewing $\Phi(S)$ as an escape probability. We start by claiming that $\pi(S) \leq 1/2$ (as required by the definition of conductance) if and only if the parameters are in the ferroelectric phase. Due to space constraints, we also defer the proof of Lemma 9 to [13]. Then we use the correspondence between tethered paths and correlated random walks (Section 3.2) to prove that $\Phi(S)$ is exponentially small.

▶ **Lemma 9.** *Let $\mu > 0$ and $\lambda > 1 + \mu$ be constants. For n sufficiently large, $\pi(S) \leq 1/2$.*

Our analysis of the escape probability from S critically relies on the fact that paths in any state $x \in S$ are non-intersecting. Combinatorially, we exploit the factorization of the generating function for states in S as a product of $2\ell+1$ independent path generating functions.

▶ **Lemma 10.** *Let $\mu, \varepsilon > 0$ be constants. For n sufficiently large, $\Phi(S) \le e^{-(1-\varepsilon)\mu^{-1}n^{1/2}}$.*

Proof. The conductance $\Phi(S)$ can be understood as the following escape probability. Sample a state $x \in S$ from the stationary distribution π conditioned on $x \in S$, and run the Markov chain from x for one step to get a neighboring state y. The definition of conductance implies that $\Phi(S)$ is the probability that $y \notin S$. Using this interpretation, we can upper bound $\Phi(S)$ by the probability mass of states that are near the boundary of S in the state space, since the process must escape in one step. Therefore, it follows from the independent paths boundary condition and the definition of S that

$$\Phi(S) \le \Pr\Big(\text{there exists a path in } x \text{ deviating by at least } 4n^{3/4} \;\Big|\; x \in S\Big).$$

Next, we use a union bound over the $2\ell+1$ different paths in a configuration and consider the event that a particular path γ_k deviates by at least $4n^{3/4}$. Because all of the paths in S are independent, we only need to consider the behavior of γ_k in isolation. This allows us to rephrase the conditional event. Relaxing the conditional probability of each term in the sum gives

$$\begin{aligned}
\Phi(S) &\le \sum_{k=-\ell}^{\ell} \Pr\Big(\gamma_k \text{ deviates by at least } 4n^{3/4} \;\Big|\; x \in S\Big) \\
&= \sum_{k=-\ell}^{\ell} \Pr\Big(\gamma_k \text{ deviates by at least } 4n^{3/4} \;\Big|\; \gamma_k \text{ deviates by less than } 8n^{3/4}\Big) \\
&\le \sum_{k=-\ell}^{\ell} \frac{\Pr\big(\gamma_k \text{ deviates by at least } 4n^{3/4}\big)}{1 - \Pr\big(\gamma_k \text{ deviates by at least } 8n^{3/4}\big)}.
\end{aligned}$$

For large enough n, the length of every path γ_k is in the range $[n, 2n]$ since we eventually have $n - \ell d \ge n/2$. Therefore, we can apply Lemma 5 with the error $\varepsilon/2$ to each term and use the universal upper bound

$$\frac{\Pr\big(\gamma_k \text{ deviates by at least } 4n^{3/4}\big)}{1 - \Pr\big(\gamma_k \text{ deviates by at least } 8n^{3/4}\big)} \le \frac{e^{-\left(1-\frac{\varepsilon}{2}\right)\frac{16n^{3/2}}{\mu n}}}{1 - e^{-\left(1-\frac{\varepsilon}{2}\right)\frac{64n^{3/2}}{\mu n}}} \le 2e^{-\left(1-\frac{\varepsilon}{2}\right)\frac{16n^{3/2}}{\mu n}}.$$

It follows from the union bound and previous inequality that the conductance $\Phi(S)$ is bounded by

$$\Phi(S) \le (2\ell+1) \cdot 2e^{-\left(1-\frac{\varepsilon}{2}\right)\frac{16n^{3/2}}{\mu n}} \le e^{-(1-\varepsilon)\mu^{-1}n^{1/2}},$$

which completes the proof. ◀

▶ **Theorem 11.** *Let $\mu, \varepsilon > 0$ and $\lambda > 1 + \mu$. For n sufficiently large, $\tau(1/4) \ge e^{(1-\varepsilon)\mu^{-1}n^{1/2}}$.*

Proof. Since $\pi(S) \le 1/2$ by Lemma 9, we have $\Phi^* \le \Phi(S)$. The proof follows from Theorem 3 and the conductance bound in Lemma 10 with a smaller error $\varepsilon/2$. ◀

Last, we restate our main theorem and use Theorem 11 to show that Glauber dynamics for the six-vertex model can be slow mixing for all parameters in the ferroelectric phase.

▶ **Theorem 1** (Ferroelectric phase). *For any $(a, b, c) \in \mathbb{R}^3_{>0}$ such that $a > b + c$ or $b > a + c$, there exist boundary conditions for which Glauber dynamics mixes exponentially slowly on Λ_n.*

Proof. Without loss of generality, we reparameterized the model so that $a = \lambda$, $b = \mu$, and $c = 1$. Therefore, Glauber dynamics with the independent paths boundary condition is slow mixing if $a > b + c$ by Theorem 11. Since the rotational invariance of the six-vertex model implies that a and b are interchangeable parameters, this mixing time result also holds in the case $b > a + c$. ◀

4 Slow Mixing in the Antiferroelectric Phase

Now we consider the mixing time of Glauber dynamics in the antiferroelectric phase, where $c > a + b$ and corners (type-c vertices) are preferred. The main insight behind our slow mixing proof is that when c is sufficiently large, the six-vertex model can behave like the low-temperature hardcore model on $\mathbb{Z}^2$ where configurations predominantly agree with one of two ground states. Liu recently formalized this argument in [28] and showed that Glauber dynamics for the six-vertex model with free boundary conditions requires exponential time when $\max(a, b) < \mu c$, where $\mu \leq 2.639$ is the connective constant of self-avoiding walks on the square lattice [17]. His proof uses a Peierls argument based on topological obstructions introduced by Randall [35] in the context of independent sets. We extend Liu's result to the region depicted in Figure 2b by computing a closed-form multivariate generating function that upper bounds the number of self-avoiding walks and accounts for disparities in their Boltzmann weights induced by the parameters of the six-vertex model.

4.1 Topological Obstruction Framework

We start with a recap of the definitions and framework laid out in [28]. There are two ground states in the antiferroelectric phase such that every interior vertex is a corner: x_{R} (Figure 4a) and x_{G} (Figure 4b). These configurations are edge reversals of each other, so for any $x \in \Omega$ we can color its edges *red* if they are oriented as in x_{R} or *green* if they are oriented as in x_{G}. See Figure 4c for an example. It follows from case analysis of the six vertex types (Figure 1) that the number of red edges incident to any internal vertex is even, and if there are only two red edges then they must be rotationally adjacent to each other. The same property holds for green edges by symmetry. Note that the four edges bounding a cell of the lattice are monochromatic if and only if they are oriented cyclically, and thus reversible by Glauber dynamics. We say that a simple path from a horizontal edge on the left boundary of Λ_n to a horizontal edge on the right boundary is a *red horizontal bridge* if it contains only red edges. We define green horizontal bridges and monochromatic vertical bridges similarly. A configuration has a *red cross* if it contains both a red horizontal bridge and a red vertical bridge, and we define a *green cross* likewise. Let $C_{\mathrm{R}} \subseteq \Omega$ be the set of all states with a red cross, and let $C_{\mathrm{G}} \subseteq \Omega$ be the set of all states with a green cross. We have $C_{\mathrm{R}} \cap C_{\mathrm{G}} = \emptyset$ by Lemma 12.

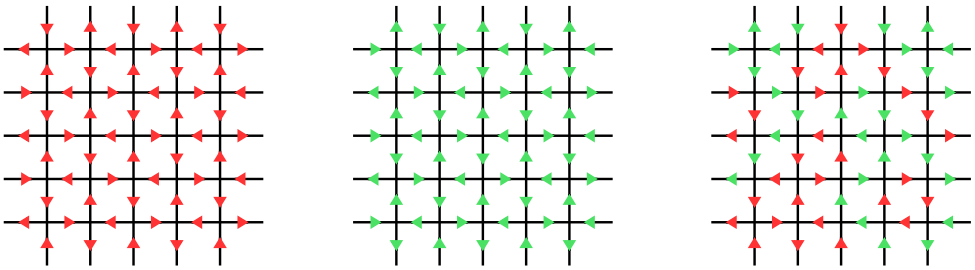


Figure 4 Edge colorings of (a) the red ground state x_{R}, (b) the green ground state x_{G}, and (c) an example configuration with free boundary conditions that does not have a monochromatic cross.

Next, we define the dual lattice L_n to describe configurations in $\Omega \setminus (C_{\mathrm{R}} \cup C_{\mathrm{G}})$. The vertices of L_n are the centers of the cells in Λ_n, including the cells on the boundary that are partially enclosed, and we connect dual vertices by an edge if their corresponding cells are diagonally adjacent. Note that L_n is a union of two disjoint graphs (Figure 5a). For any state $x \in \Omega$ there is a corresponding dual subgraph L_x defined as follows: for each interior vertex v in Λ_n, if v is incident to two red edges and two green edges, then L_x contains the dual edge passing through v that separates the two red edges from the two green edges. This construction is well-defined because the red edges are rotationally adjacent. See Figure 5b for an example. For any $x \in \Omega$, we say that x has a *horizontal fault line* if L_x contains a simple path from a left dual boundary vertex to a right dual boundary vertex. We define horizontal fault lines similarly and let $C_{\mathrm{FL}} \subseteq \Omega$ be the set of all states containing a horizontal or vertical fault line. Observe that fault lines completely separate red and green edges, and hence are topological obstructions that prohibit monochromatic bridges.

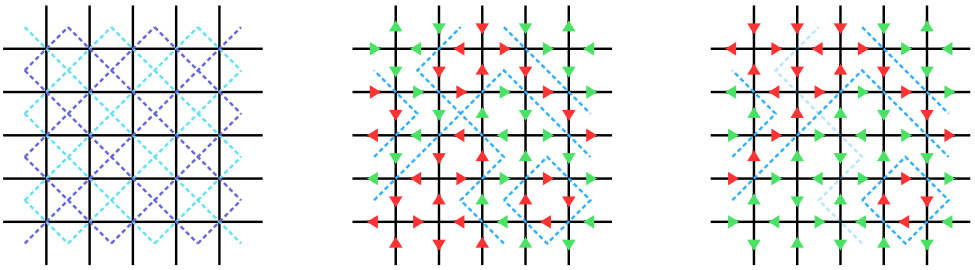


Figure 5 Illustrations of (a) the dual lattice L_n as a union of disjoint cyan and purple subgraphs, (b) an example configuration overlaid with its dual graph, and (c) the example under the injective fault line map.

Last, we extend the notion of fault lines to *almost fault lines*. We say that $x \in \Omega$ has a horizontal almost fault line if there is a simple path in L_n connecting a left dual boundary vertex to a right dual boundary vertex such that all edges except for one are in L_x. We define vertical almost fault lines similarly and let the set $C_{\mathrm{AFL}} \subseteq \Omega$ denote all states containing an almost fault line. Finally, let $\partial C_{\mathrm{R}} \subseteq \Omega$ denote the set of states not in C_{R} that one move away from C_{R} in the state space according to the Glauber dynamics.

▶ **Lemma 12** ([28]). *We can partition the state space into* $\Omega = C_{\mathrm{R}} \cup C_{\mathrm{FL}} \cup C_{\mathrm{G}}$. *Furthermore, we have* $\partial C_{\mathrm{R}} \subseteq C_{\mathrm{FL}} \cup C_{\mathrm{AFL}}$.

4.2 Weighted Non-Backtracking Walks and a Peierls Argument

In this subsection we show that $\pi(C_{\mathrm{FL}} \cup C_{\mathrm{AFL}})$ is an exponentially small bottleneck in the state space Ω. The analysis relies on Lemma 12 and a new multivariate upper bound for weighted self-avoiding walks (Lemma 13). Our key observation is that when a fault line changes direction, the vertices in its path change from type-a to type-b or vice versa. Therefore, our goal in this subsection is to generalize the trivial 3^{n-1} upper bound for the number of self-avoiding walks by accounting for their changes in direction in aggregate. We achieve this by using generating functions to solve a system of linear recurrence relations.

We start by encoding *non-backtracking walks* that start from the origin and take their first step northward using the characters in $\{\mathsf{S}, \mathsf{L}, \mathsf{R}\}$, representing straight, left, and right steps. For example, the walk SLRSSL corresponds uniquely to the sequence

$((0,0),(0,1),(-1,1),(-1,2),(-1,3),(-1,4),(-2,4))$. If a fault line is the same shape as SLRSSL up to a rotation about the origin, then there are only two possible sequences of vertex types through which it can pass: *abaaab* and *babbba*. This follows from the fact that once the first vertex type is determined, only turns in the self-avoiding walk (i.e., the L and R characters) cause the vertex type to switch. We define the weight of a fault line to be the product of the vertex types through which it passes. More generally, we define the weight of a non-backtracking walk that initially passes through a fixed vertex type to be the product of the induced vertex types according to the rule that turns toggle the current type. Formally, we let $g_a(\gamma) : \{\mathsf{S}\} \times \{\mathsf{S},\mathsf{L},\mathsf{R}\}^{n-1} \to \mathbb{R}$ denote the weight of a non-backtracking walk γ that starts by crossing a type-a vertex. We define the function $g_b(\gamma)$ similarly and note that $g_a(\mathsf{SLRSSL}) = a^4b^2$ and $g_b(\mathsf{SLRSSL}) = a^2b^4$. Last, observe that a sequence of vertex types can have many different walks in its preimage. The non-backtracking walk SRRSSR also maps to *abaaab* and *babbba* – in fact, there are $2^3 = 8$ such walks in this example since we can interchange L and R characters.

The idea of enumerating the preimages of a binary string corresponding to sequence of vertex types suggests a recursive approach for computing the sum of weighted non-backtracking walks. This naturally leads to the use of generating functions, so overload the variables x and y to also denote function arguments. For nonempty binary string $s \in \{0,1\}^n$, let $h(s)$ count the number of pairs of adjacent characters that are not equal and let $|s|$ denote the number of ones in s (e.g., if $s = 010001$ then $h(s) = 3$ and $|s| = 2$). The sum of weighted self-avoiding walks is upper bounded by the sum of weighted non-backtracking walks, so we proceed by analyzing the following function:

$$F_n(x,y) \stackrel{\text{def}}{=} \sum_{\gamma \in \{\mathsf{S}\} \times \{\mathsf{S},\mathsf{L},\mathsf{R}\}^{n-1}} g_x(\gamma) + g_y(\gamma) = \sum_{s \in \{0,1\}^n} 2^{h(s)} x^{|s|} y^{n-|s|}. \tag{3}$$

Note that $F_n(1,1) = 2 \cdot 3^{n-1}$ recovers the number of non-backtracking walks that initially cross type-a or type-b vertices. We compute a closed-form solution for $F_n(x,y)$ in the full version [13] by diagonalizing a matrix corresponding to the system of recurrence relations, which allows us to accurately capture the discrepancy between fault lines when the Boltzmann weights a and b differ.

▶ **Lemma 13.** *Let $F_n(x,y)$ be the generating function for weighted non-backtracking walks defined in (3). For any integer $n \geq 1$ and $x, y \in \mathbb{R}_{>0}$, we have*

$$F_n(x,y) \leq 3(x+y)\left(\frac{x+y+\sqrt{x^2+14xy+y^2}}{2}\right)^{n-1}.$$

We are now ready to present our Peierls argument to bound $\pi(C_{\mathrm{FL}} \cup C_{\mathrm{AFL}})$, which gives us a bound on the conductance and allows us to prove Theorem 2. First, we describe which antiferroelectric parameters cause $F_n(a/c, b/c)$ to decrease exponentially fast.

▶ **Lemma 14.** *If $(a,b,c) \in \mathbb{R}^3_{>0}$ is antiferroelectric and $3ab + ac + bc < c^2$, then we have $a + b + \sqrt{a^2 + 14ab + b^2} < 2c$.*

▶ **Lemma 15.** *If $(a,b,c) \in \mathbb{R}^3_{>0}$ is antiferroelectric and $3ab + ac + bc < c^2$, for free boundary conditions we have*

$$\pi\left(C_{\mathrm{FL}} \cup C_{\mathrm{AFL}}\right) \leq \mathrm{poly}(n)\left(\frac{a+b+\sqrt{a^2+14ab+b^2}}{2c}\right)^n.$$

Proof. For any self-avoiding walk γ and dual vertices $s, t \in L_n$ on the boundary, let $\Omega_{\gamma,s,t} \subseteq \Omega$ be the set of states containing γ as a fault line or an almost fault line such that γ starts at s and ends at t. Without loss of generality, assume that the (almost) fault line is vertical. Reversing the direction of all edges on the left side of γ defines the injective map $f_{\gamma,s,t} : \Omega_{\gamma,s,t} \to \Omega \setminus \Omega_{\gamma,s,t}$ such that if γ is a fault line of $x \in \Omega_{\gamma,s,t}$, then the weight of its image $f_{\gamma,s,t}(x)$ is amplified by $c^{|\gamma|}/g_a(\gamma)$ or $c^{|\gamma|}/g_b(\gamma)$. See Figure 5c for an example. Similarly, if γ is an almost fault line, decompose γ into subpaths γ_1 and γ_2 separated by a type-c vertex such that γ_1 starts at s and γ_2 ends at t. In this case, the weight of the images of almost fault lines is amplified by a factor of $\min(a,b)/c \cdot c^{|\gamma_1|+|\gamma_2|}/(g_\alpha(\gamma_1)g_\beta(\gamma_2))$ for some $(\alpha,\beta) \in \{a,b\}^2$. Using the fact that $f_{\gamma,s,t}$ is injective and summing over the states containing γ as a fault line and an almost fault line separately gives us

$$\pi(\Omega_{\gamma,s,t}) \leq \frac{g_a(\gamma)+g_b(\gamma)}{c^{|\gamma|}} + \frac{c}{\min(a,b)} \sum_{\gamma_1+\gamma_2=\gamma} \frac{g_a(\gamma_1)+g_b(\gamma_1)}{c^{|\gamma_1|}} \cdot \frac{g_a(\gamma_2)+g_b(\gamma_2)}{c^{|\gamma_2|}}, \tag{4}$$

where the sum is over all $\Theta(|\gamma|)$ decompositions of γ into γ_1 and γ_2.

Equipped with (4) and Lemma 13, we use a union bound over all pairs of terminals (s,t) and fault line lengths ℓ to upper bound $\pi(C_{\mathrm{FL}} \cup C_{\mathrm{AFL}})$ in terms of our generating function for weighted non-backtracking walks $F_\ell(x,y)$. Since the antiferroelectric weights satisfy $3ab + ac + bc < c^2$, it follows from Lemma 14 that

$$\begin{aligned}
\pi(C_{\mathrm{FL}} \cup C_{\mathrm{AFL}}) &\leq \sum_{(s,t)} \sum_{\ell=n}^{n^2} \left(F_\ell(a/c, b/c) + \frac{c}{\min(a,b)} \sum_{k=0}^{\ell} F_k(a/c,b/c) F_{\ell-k}(a/c,b/c) \right) \\
&\leq \sum_{(s,t)} \sum_{\ell=n}^{n^2} \mathrm{poly}(\ell) \left(\frac{a+b+\sqrt{a^2+14ab+b^2}}{2c} \right)^{\ell} \\
&\leq \mathrm{poly}(n) \left(\frac{a+b+\sqrt{a^2+14ab+b^2}}{2c} \right)^{n}.
\end{aligned}$$

Note that the convolutions in the first inequality generate all *almost* weighted non-backtracking walks. ◀

▶ **Theorem 2** (Antiferroelectric phase). *For any $(a,b,c) \in \mathbb{R}^3_{>0}$ such that $ac + bc + 3ab < c^2$, Glauber dynamics mixes exponentially slowly on Λ_n with free boundary conditions.*

Proof of Theorem 2. Let $\Omega_{\mathrm{MIDDLE}} = C_{\mathrm{FL}} \cup C_{\mathrm{AFL}}$, $\Omega_{\mathrm{LEFT}} = C_{\mathrm{R}} \setminus \Omega_{\mathrm{MIDDLE}}$, and $\Omega_{\mathrm{RIGHT}} = C_{\mathrm{G}} \setminus \Omega_{\mathrm{MIDDLE}}$. It follows from Lemma 12 that $\Omega = \Omega_{\mathrm{LEFT}} \cup \Omega_{\mathrm{MIDDLE}} \cup \Omega_{\mathrm{RIGHT}}$ is a partition with the properties that $\partial\Omega_{\mathrm{LEFT}} \subseteq \Omega_{\mathrm{MIDDLE}}$ and $\pi(\Omega_{\mathrm{LEFT}}) = \pi(\Omega_{\mathrm{RIGHT}})$. Since the partition is symmetric, Lemma 15 implies that $1/4 \leq \pi(\Omega_{\mathrm{LEFT}}) \leq 1/2$, for n sufficiently large. Therefore, we can upper bound the conductance by $\Phi^* \leq \Phi(\Omega_{\mathrm{LEFT}}) \leq 4\pi(\Omega_{\mathrm{MIDDLE}})$. Using Theorem 3 with Lemma 15 and Lemma 14 gives the desired mixing time bound. ◀

References

1 David Allison and Nicolai Reshetikhin. Numerical study of the 6-vertex model with domain wall boundary conditions. *Annales de l'institut Fourier*, 55(6):1847–1869, 2005.

2 Prateek Bhakta, Ben Cousins, Matthew Fahrbach, and Dana Randall. Approximately sampling elements with fixed rank in graded posets. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1828–1838. SIAM, 2017.

3 Pavel Bleher and Vladimir Fokin. Exact solution of the six-vertex model with domain wall boundary conditions. Disordered phase. *Communications in Mathematical Physics*, 268(1):223–284, 2006.

4 Pavel Bleher and Karl Liechty. Exact solution of the six-vertex model with domain wall boundary conditions. Ferroelectric phase. *Communications in Mathematical Physics*, 286(2):777–801, 2009.

5 Pavel Bleher and Karl Liechty. Exact Solution of the Six-Vertex Model with Domain Wall Boundary Conditions: Antiferroelectric Phase. *Communications on Pure and Applied Mathematics*, 63(6):779–829, 2010.

6 N. M. Bogoliubov, A. G. Pronko, and M. B.. Zvonarev. Boundary correlation functions of the six-vertex model. *Journal of Physics A: Mathematical and General*, 35(27):5525, 2002.

7 H. J. Brascamp, H. Kunz, and F. Y. Wu. Some rigorous results for the vertex model in statistical mechanics. *Journal of Mathematical Physics*, 14(12):1927–1932, 1973.

8 Jin-Yi Cai, Zhiguo Fu, and Mingji Xia. Complexity classification of the six-vertex model. *Information and Computation*, 259:130–141, 2018.

9 Jin-Yi Cai, Tianyu Liu, and Pinyan Lu. Approximability of the Six-vertex Model. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2248–2261. SIAM, 2019.

10 Sarah Cannon and Dana Randall. Sampling on lattices with free boundary conditions using randomized extensions. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1952–1971. Society for Industrial and Applied Mathematics, 2016.

11 Henry Cohn, Noam Elkies, and James Propp. Local statistics for random domino tilings of the Aztec diamond. *Duke Mathematics Journal*, 85(1):117–166, October 1996. doi:10.1215/S0012-7094-96-08506-3.

12 David Durfee, Matthew Fahrbach, Yu Gao, and Tao Xiao. Nearly tight bounds for sandpile transience on the grid. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 605–624. SIAM, 2018.

13 Matthew Fahrbach and Dana Randall. Slow Mixing of Glauber Dynamics for the Six-Vertex Model in the Ferroelectric and Antiferroelectric Phases. *arXiv preprint*, 2019. arXiv:1904.01495.

14 Patrik L. Ferrari and Herbert Spohn. Domino tilings and the six-vertex model at its free-fermion point. *Journal of Physics A: Mathematical and General*, 39(33):10297, 2006.

15 J. Gillis. Correlated random walk. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(4):639–651, 1955.

16 Leslie Ann Goldberg, Russell Martin, and Mike Paterson. Random sampling of 3-colorings in $\mathbb{Z}^2$. *Random Structures & Algorithms*, 24(3):279–302, 2004.

17 A. J. Guttmann and A. R. Conway. Square lattice self-avoiding walks and polygons. *Annals of Combinatorics*, 5(3-4):319–345, 2001.

18 J. W. Hanneken and D. R. Franceschetti. Exact distribution function for discrete time correlated random walks in one dimension. *The Journal of Chemical Physics*, 109(16):6533–6539, 1998.

19 Lingxiao Huang, Pinyan Lu, and Chihao Zhang. Canonical paths for MCMC: From art to science. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 514–527. Society for Industrial and Applied Mathematics, 2016.

20 Anatoli G Izergin, David A Coker, and Vladimir E Korepin. Determinant formula for the six-vertex model. *Journal of Physics A: Mathematical and General*, 25(16):4315, 1992.

21 David Keating and Ananth Sridhar. Random tilings with the GPU. *Journal of Mathematical Physics*, 59(9):091420, 2018.

22 Vladimir Korepin and Paul Zinn-Justin. Thermodynamic limit of the six-vertex model with domain wall boundary conditions. *Journal of Physics A: Mathematical and General*, 33(40):7053, 2000.

23 Greg Kuperberg. Another proof of the alternative-sign matrix conjecture. *International Mathematics Research Notices*, 1996(3):139–150, 1996.

24 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times*, volume 107. American Mathematical Society, 2017.

25 Elliott H Lieb. Exact solution of the problem of the entropy of two-dimensional ice. *Physical Review Letters*, 18(17):692, 1967.

26 Elliott H Lieb. Exact Solution of the Two-Dimensional Slater KDP Model of a Ferroelectric. *Physical Review Letters*, 19(3):108, 1967.

27 Elliott H Lieb. Residual Entropy of Square Ice. *Physical Review*, 162(1):162, 1967.

28 Tianyu Liu. Torpid Mixing of Markov Chains for the Six-vertex Model on $\mathbb{Z}^2$. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques* (APPROX/RANDOM). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

29 Michael Luby, Dana Randall, and Alistair Sinclair. Markov chain algorithms for planar lattice structures. *SIAM Journal on Computing*, 31(1):167–192, 2001.

30 Ivar Lyberg, Vladimir Korepin, G. A. P. Ribeiro, and Jacopo Viti. Phase separation in the six-vertex model with a variety of boundary conditions. *Journal of Mathematical Physics*, 59(5):053301, 2018.

31 Ivar Lyberg, Vladimir Korepin, and Jacopo Viti. The density profile of the six vertex model with domain wall boundary conditions. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(5):053103, 2017.

32 Colin McQuillan. Approximating holant problems by winding. *arXiv preprint*, 2013. `arXiv:1301.2880`.

33 Linus Pauling. The structure and entropy of ice and of other crystals with some randomness of atomic arrangement. *Journal of the American Chemical Society*, 57(12):2680–2684, 1935.

34 Yuval Peres and Perla Sousi. Mixing times are hitting times of large sets. *Journal of Theoretical Probability*, 28(2):488–519, 2015.

35 Dana Randall. Slow mixing of Glauber dynamics via topological obstructions. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 870–879. Society for Industrial and Applied Mathematics, 2006.

36 Dana Randall and Prasad Tetali. Analyzing Glauber dynamics by comparison of Markov chains. *Journal of Mathematical Physics*, 41(3):1598–1615, 2000.

37 Eric Renshaw and Robin Henderson. The correlated random walk. *Journal of Applied Probability*, 18(2):403–414, 1981.

38 Bill Sutherland. Exact solution of a two-dimensional model for hydrogen-bonded crystals. *Physical Review Letters*, 19(3):103, 1967.

39 David Bruce Wilson. Mixing times of lozenge tiling and card shuffling Markov chains. *The Annals of Applied Probability*, 14(1):274–325, 2004.

40 C. P. Yang. Exact solution of a model of two-dimensional ferroelectrics in an arbitrary external electric field. *Physical Review Letters*, 19(10):586, 1967.

41 Doron Zeilberger. Proof of the alternating sign matrix conjecture. *Electronic Journal of Combinatorics*, 3(2):R13, 1996.

A Tail Behavior of Correlated Random Walks

In this section we prove Lemma 8, which gives an exponentially small upper bound for the tail of a correlated random walk as a function of its momentum parameter μ. Our proof builds off of the PDF for the position of a correlated random walk given as Lemma 4, which is combinatorial in nature and not readily amenable for tail inequalities. Specifically, the probability $\Pr(S_{2n} = 2m)$ is a sum of marginals conditioned on the number of turns that the walk makes [37].

There are two main ideas in our approach to develop a more useful bound for the position of a correlated random walk $\Pr(S_{2n} = 2m)$. First, we construct a smooth function that upper bounds the marginals as a function of x (a continuation of the number of turns in the walk k), and then we determine its maximum value. Next we show that the log of the maximum value is asymptotically equivalent to $m^2/(\mu n)$ for $m = o(n)$, which gives us desirable bounds

for sufficiently large values of n. We point out that this analysis illustrates precisely how correlated random walks generalize simple symmetric random walks and how the momentum parameter μ controls the exponential decay.

A.1 Upper Bounding the Marginal Probabilities

We start by using Stirling's approximation to construct a smooth function that upper bounds the marginal terms in the sum of the PDF for correlated random walks. For $x \in (0, n-m)$, let

$$f(x) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } x = 0, \\ \frac{(n+m)^{n+m}}{x^x (n+m-x)^{n+m-x}} \cdot \frac{(n-m)^{n-m}}{x^x (n-m-x)^{n-m-x}} \cdot \mu^{-2x} & \text{if } x \in (0, n-m), \\ \mu^{-2(n-m)} & \text{if } x = n-m. \end{cases} \tag{5}$$

It can easily be checked that $f(x)$ is continuous on all of $[0, n-m]$ since $\lim_{x \to 0} x^x = 1$.

▶ **Lemma 16.** *For any integer $m \geq 0$, a correlated random walk satisfies*

$$\Pr(S_{2n} = 2m) \leq \text{poly}(n) \sum_{k=0}^{n-m} \left(\frac{\mu}{1+\mu}\right)^{2n} f(k).$$

Proof. Consider the probability density function for $\Pr(S_{2n} = 2m)$ in Lemma 4. If $2m = 2n$ the claim is clearly true, so we focus on the other case. We start by bounding the rightmost polynomial term in the sum. For all $n \geq 1$, we have $n(1-p) + k(2p-1) \leq 2nk$. Next, we reparameterize the marginals in terms of μ, where $p = \mu/(1+\mu)$, and use a more convenient upper bound for the binomial coefficients. Observe that

$$\begin{aligned} \Pr(S_{2n} = 2m) &\leq 2n \sum_{k=1}^{n-m} \binom{n+m-1}{k-1} \binom{n-m-1}{k-1} \left(\frac{1}{1+\mu}\right)^{2k-1} \left(\frac{\mu}{1+\mu}\right)^{2n-1-2k} \\ &\leq \text{poly}(n) \sum_{k=0}^{n-m} \binom{n+m}{k} \binom{n-m}{k} \left(\frac{\mu}{1+\mu}\right)^{2n} \mu^{-2k}. \end{aligned}$$

Stirling's approximation states that for all $n \geq 1$ we have $e(n/e)^n \leq n! \leq en(n/e)^n$, so we can bound the products of binomial coefficients up to a polynomial factor by

$$\begin{aligned} \binom{n+m}{k} \binom{n-m}{k} &\leq \text{poly}(n) \cdot \frac{\left(\frac{n+m}{e}\right)^{n+m}}{\left(\frac{k}{e}\right)^k \left(\frac{n+m-k}{e}\right)^{n+m-k}} \cdot \frac{\left(\frac{n-m}{e}\right)^{n-m}}{\left(\frac{k}{e}\right)^k \left(\frac{n-m-k}{e}\right)^{n-m-k}} \\ &= \text{poly}(n) \cdot \frac{(n+m)^{n+m}}{k^k (n+m-k)^{n+m-k}} \cdot \frac{(n-m)^{n-m}}{k^k (n-m-k)^{n-m-k}}. \end{aligned}$$

The proof follows the definition of $f(x)$ given in (5). ◀

There are polynomially-many marginal terms in the sum of the PDF, so if the maximum term is exponentially small, then the total probability is exponentially small. Since the marginal terms are bounded above by an expression involving $f(x)$, we can proceed by maximizing $f(x)$ on its support.

▶ **Lemma 17.** *The function $f(x)$ is maximized at the critical point*

$$x^* = \begin{cases} \frac{n^2 - m^2}{2n} & \text{if } \mu = 1, \\ \frac{n}{1-\mu^2} \left(1 - \sqrt{\mu^2 + (1-\mu^2)\frac{m^2}{n^2}}\right) & \text{otherwise.} \end{cases}$$

Proof. We start by showing that $f(x)$ is log-concave on $(0, n-m)$, which implies that it is unimodal. It follows that a local maximum of $f(x)$ is a global maximum. Since n and k are fixed as constants and because the numerator is positive, it is sufficient to show that

$$\begin{aligned} g(x) &= -\log\big(x^x(n+m-x)^{n+m-x} \cdot x^x(n-m-x)^{n-m-x} \cdot \mu^{2x}\big) \\ &= -(2x\log(\mu x) + (n+m-x)\log(n+m-x) + (n-m-x)\log(n-m-x)) \end{aligned}$$

is concave. Observe that the first derivative of $g(x)$ is

$$\begin{aligned} g'(x) &= -2(1+\log(\mu x)) + (1+\log(n+m-x)) + (1+\log(n-m-x)) \\ &= -2\log(\mu x) + \log(n+m-x) + \log(n-m-x), \end{aligned}$$

and the second derivative is

$$g''(x) = -\frac{2}{x} - \frac{1}{n+m-x} - \frac{1}{n-m-x}.$$

Because $g''(x) < 0$ on $(0, n-m)$, the function $f(x)$ is log-concave and hence unimodal.

To identify the critical points of $f(x)$, it suffices to determine where $g'(x) = 0$ since $\log x$ is increasing. Using the previous expression for $g'(x)$, it follows that

$$g'(x) = \log\left[\frac{(n-x)^2 - m^2}{\mu^2 x^2}\right]. \tag{6}$$

Therefore, the critical points are the solutions of $(n-x)^2 - m^2 = \mu^2 x^2$, so we have

$$x^* = \begin{cases} \frac{n^2-m^2}{2n} & \text{if } \mu = 1, \\ \frac{n - \sqrt{n^2 - (1-\mu^2)(n^2-m^2)}}{1-\mu^2} & \text{otherwise.} \end{cases}$$

It remains and suffices to show that x^* is a local maximum since $f(x)$ is unimodal. Observing that $\frac{\partial}{\partial x}\log f(x) = g'(x)$ and differentiating $f(x) = \exp(\log f(x))$ using the chain rule, the definition of x^* gives

$$f''(x^*) = e^{\log f(x^*)}\Big[g''(x^*) + g'(x^*)^2\Big] = f(x^*)g''(x^*).$$

We know $f(x^*) > 0$, so $f''(x^*)$ has the same sign as $g''(x^*) < 0$. Therefore, x^* is a local maximum of $f(x)$. Using the continuity of $f(x)$ on $[0, n-m]$ and log-concavity, $f(x^*)$ is a global maximum. ◀

A.2 Asymptotic Behavior of the Maximum Log Marginal

Now that we have a formula for x^*, and hence an expression for $f(x^*)$, we want to show that

$$\left(\frac{\mu}{1+\mu}\right)^{2n} f(x^*) \le e^{-n^c},$$

for some constant $c > 0$. Because there are polynomially-many marginals in the sum, this leads to an exponentially small upper bound for $\Pr(S_{2n} = 2m)$. Define the *maximum log marginal* to be

$$h(n) \stackrel{\text{def}}{=} -\log\left[\left(\frac{\mu}{1+\mu}\right)^{2n} f(x^*)\right]. \tag{7}$$

Equivalently, we show that $h(n) \ge n^c$ for sufficiently large n using asymptotic equivalences.

▶ **Lemma 18.** *The maximum log marginal $h(n)$ can be symmetrically expressed as*

$$h(n) = (n+m)\log\left[\left(\frac{1+\mu}{\mu}\right)\left(1-\frac{x^*}{n+m}\right)\right] + (n-m)\log\left[\left(\frac{1+\mu}{\mu}\right)\left(1-\frac{x^*}{n-m}\right)\right].$$

Proof. Grouping the terms of $h(n)$ by factors of n, m and x^* gives

$$n\log\left[\left(\frac{1+\mu}{\mu}\right)^2\frac{(n-x^*)^2-m^2}{(n+m)(n-m)}\right] + m\log\left[\frac{(n-m)(n+m-x^*)}{(n+m)(n-m-x^*)}\right] + x^*\log\left[\frac{(\mu x^*)^2}{(n-x^*)^2-m^2}\right].$$

Using (6), observe that the last term is

$$x^*\log\left[\frac{(\mu x^*)^2}{(n-x^*)^2-m^2}\right] = -x^* g'(x^*) = 0.$$

The proof follows by grouping the terms of the desired expression by factors of n and m. ◀

The following lemma is the crux of our argument, as it presents an asymptotic equality for the maximum log marginal in the PDF for correlated random walks. We remark that we attempted to bound this quantity directly using Taylor expansions instead of an asymptotic equivalence, and while this seems possible, the expressions are unruly. Our asymptotic equivalence demonstrates that second derivative information is needed, which makes the earlier approach even more unmanageable.

▶ **Lemma 19.** *For $\mu > 0$ and $m = o(n)$, the maximum log marginal satisfies $h(n) \sim m^2/(\mu n)$.*

Proof. The proof is by case analysis for μ. In both cases we analyze $h(n)$ as expressed in Lemma 18, consider a change of variables, and use L'Hospital's rule twice. In the first case, we assume $\mu = 1$. The value of x^* in Lemma 17 gives us

$$1-\frac{x^*}{n+m} = \frac{2n(n+m)-\left(n^2-m^2\right)}{2n(n+m)} = \frac{n+m}{2n}$$
$$1-\frac{x^*}{n-m} = \frac{2n(n-m)-\left(n^2-m^2\right)}{2n(n-m)} = \frac{n-m}{2n}.$$

It follows that $h(n)$ can be simplified as

$$h(n) = n\log\left[\left(\frac{1+\mu}{\mu}\right)^2\left(\frac{n^2-m^2}{4n^2}\right)\right] + m\log\left(\frac{n+m}{n-m}\right) = n\log\left(1-\frac{m^2}{n^2}\right) + m\log\left(1+\frac{2m}{n-m}\right).$$

To show $h(n) \sim m^2/n$, by the definition of asymptotic equivalence we need to prove that

$$\lim_{n\to\infty}\frac{n\log\left(1-\frac{m^2}{n^2}\right)+m\log\left(1+\frac{2m}{n-m}\right)}{\frac{m^2}{n}} = 1.$$

Make the change of variables $y = m/n$. Since $m = o(n)$, this is equivalent to showing

$$\lim_{y\to 0}\frac{\log\left(1-y^2\right)+y\log\left(1+\frac{2y}{1-y}\right)}{y^2} = 1.$$

Using L'Hospital's rule twice with the derivatives

$$\frac{\partial}{\partial y}\left[\log\left(1-y^2\right)+y\log\left(1+\frac{2y}{1-y}\right)\right] = \log\left(-\frac{y+1}{y-1}\right)$$
$$\frac{\partial^2}{\partial y^2}\left[\log\left(1-y^2\right)+y\log\left(1+\frac{2y}{1-y}\right)\right] = \frac{2}{1-y^2},$$

it follows that

$$\lim_{y\to 0} \frac{\log(1-y^2) + y\log\left(1+\frac{2y}{1-y}\right)}{y^2} = \lim_{y\to 0} \frac{\log\left(-\frac{y+1}{y-1}\right)}{2y} = \lim_{y\to 0} \frac{\frac{2}{1-y^2}}{2} = 1.$$

This completes the proof for $\mu = 1$.

The case when $\mu \neq 1$ is analogous but messier. Making the same change of variables $y = m/n$, it is equivalent to show that

$$\begin{aligned}(1+y)\log\left[\left(\frac{1+\mu}{\mu}\right)\left(1-\frac{1}{1-\mu^2}\cdot\frac{1}{1+y}\cdot\left(1-\sqrt{\mu^2+(1-\mu^2)y^2}\right)\right)\right] \\ +(1-y)\log\left[\left(\frac{1+\mu}{\mu}\right)\left(1-\frac{1}{1-\mu^2}\cdot\frac{1}{1-y}\cdot\left(1-\sqrt{\mu^2+(1-\mu^2)y^2}\right)\right)\right] \sim \mu^{-1}y^2, \quad (8)\end{aligned}$$

because the value of x^* for $\mu \neq 1$ in Lemma 17 gives us

$$1-\frac{x^*}{n+m} = 1-\frac{1}{n+m}\cdot\frac{n}{1-\mu^2}\cdot\left(1-\sqrt{\mu^2+(1-\mu^2)\frac{m^2}{n^2}}\right).$$

Denoting the left-hand side of (8) by $g(y)$, one can verify the first two derivatives of $g(y)$ are

$$g'(y) = \log\left(\frac{\mu^2-\sqrt{\mu^2-\mu^2y^2+y^2}+\left(\mu^2-1\right)y}{(\mu-1)\mu(y+1)}\right) - \log\left(\frac{-\mu^2+\sqrt{\mu^2-\mu^2y^2+y^2}+\left(\mu^2-1\right)y}{(\mu-1)\mu(y-1)}\right)$$

$$g''(y) = \frac{2}{(1+y)(1-y)\sqrt{y^2-\mu^2(y^2-1)}}.$$

Observing that $g(0) = g'(0) = 0$ due to cancellations and using L'Hospital's rule twice,

$$\lim_{y\to 0}\frac{g(y)}{\mu^{-1}y^2} = \lim_{y\to 0}\frac{g'(y)}{2\mu^{-1}y} = \lim_{y\to 0}\frac{2}{(1+y)(1-y)\sqrt{y^2-\mu^2(y^2-1)}}\cdot\frac{\mu}{2} = 1.$$

This completes the proof for all cases of μ. ◀

▶ **Lemma 8.** *Let $\mu, \varepsilon > 0$ and $m = o(n)$. For n sufficiently large, a correlated random walk satisfies* $\Pr(S_{2n} = 2m) \leq e^{-(1-\varepsilon)\frac{m^2}{\mu n}}$.

Proof. For n sufficiently large, the asymptotic equality for $h(n)$ in Lemma 19 gives us

$$h(n) \geq \left(1-\frac{\varepsilon}{2}\right)\frac{m^2}{\mu n}.$$

It follows from our construction of $f(x)$ and the definition of the maximum log marginal that

$$\begin{aligned}\Pr(S_{2n} = 2m) &\leq \mathrm{poly}(n)\cdot\left(\frac{\mu}{1+\mu}\right)^{2n} f(x^*) \\ &= \mathrm{poly}(n)\cdot e^{-h(n)} \\ &\leq \mathrm{poly}(n)\cdot e^{-\left(1-\frac{\varepsilon}{2}\right)\frac{m^2}{\mu n}} \\ &\leq e^{-(1-\varepsilon)\frac{m^2}{\mu n}},\end{aligned}$$

as desired. ◀

Lifted Multiplicity Codes and the Disjoint Repair Group Property

Ray Li
Department of Computer Science, Stanford University, CA, USA
https://www.cs.stanford.edu/~rayyli/
rayyli@cs.stanford.edu

Mary Wootters
Departments of Computer Science and Electrical Engineering, Stanford University, CA, USA
https://sites.google.com/site/marywootters/
marykw@stanford.edu

Abstract

Lifted Reed Solomon Codes (Guo, Kopparty, Sudan 2013) were introduced in the context of locally correctable and testable codes. They are multivariate polynomials whose restriction to any line is a codeword of a Reed-Solomon code. We consider a generalization of their construction, which we call *lifted multiplicity codes*. These are multivariate polynomial codes whose restriction to any line is a codeword of a multiplicity code (Kopparty, Saraf, Yekhanin 2014). We show that lifted multiplicity codes have a better trade-off between redundancy and a notion of locality called the t-disjoint-repair-group property than previously known constructions. More precisely, we show that, for $t \leq \sqrt{N}$, lifted multiplicity codes with length N and redundancy $O(t^{0.585}\sqrt{N})$ have the property that any symbol of a codeword can be reconstructed in t different ways, each using a disjoint subset of the other coordinates. This gives the best known trade-off for this problem for any super-constant $t < \sqrt{N}$. We also give an alternative analysis of lifted Reed Solomon codes using dual codes, which may be of independent interest.

2012 ACM Subject Classification Theory of computation → Error-correcting codes

Keywords and phrases Lifted codes, Multiplicity codes, Disjoint repair group property, PIR code, Coding theory

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.38

Category RANDOM

Related Version https://arxiv.org/abs/1905.02270

Funding *Ray Li*: Research supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE - 1656518.
Mary Wootters: Research partially supported by NSF grants CCF-1657049 and CCF-1844628.

Acknowledgements We thank Eitan Yaakobi for helpful conversations. We thank Julien Lavauzelle for pointing out the reference [26] and also for pointing out an error in an earlier version of this paper. We thank Nikita Polianskii for pointing out an error in an earlier version of this paper. A previous version claimed that a lifted code is exactly the span of all good monomials, but in fact the span of all good monomials only forms a subset of the lifted code (see Remark 18). This does not change our main result, as our lower bound on the number of good monomials still gives the same lower bound on the rate of the lifted code. We thank anonymous reviewers for helpful comments on an earlier draft of this paper.

1 Introduction

In this work we study *lifted multiplicity codes,* and show how they provide improved constructions of codes with the *t-disjoint repair group property* (t-DRGP), a notion of locality in error correcting codes.


© Ray Li and Mary Wootters;
licensed under Creative Commons License CC-BY

An *error correcting code* of length N over an alphabet Σ is a set $\mathcal{C} \subseteq \Sigma^N$. There are several desirable properties in error correcting codes, and in this paper we study the trade-off between two of them. The first is the size of $\mathcal{C}$, which we would like to be as big as possible given N. The second desirable property is *locality.* Informally, a code $\mathcal{C}$ exhibits locality if, given (noisy) access to $c \in \mathcal{C}$, one can learn the i'th symbol c_i of c in sublinear time. As we discuss more below, locality arises in a number of areas, from distributed storage to complexity theory.

Two constructions of codes with locality are *lifted codes* [6] and *multiplicity codes* [15]; in fact, both of these constructions were among the first known high-rate Locally Correctable Codes. In this work, consider a combination of the two ideas in *lifted multiplicity codes,* and we show that these codes exhibit locality beyond what's known for either lifted codes or for multiplicity codes.

More precisely, we study a particular notion of locality called the *t-disjoint-repair-group property* (t-DRGP). Informally, we say that $\mathcal{C}$ has the t-DRGP if any symbol c_i of $c \in \mathcal{C}$ can be obtained in t different ways, each of which involves a disjoint set of coordinates of c. Formally, we have the following definition.

▶ **Definition 1.** *A code $\mathcal{C} \subseteq \Sigma^N$ has the* t-disjoint repair property *if for every $i \in [N]$, there is a collection of t disjoint subsets $S_1, \ldots, S_t \subseteq [N] \setminus \{i\}$, and functions $f_1, \ldots, f_t$ so that for all $c \in \mathcal{C}$ and for all $j \in [t]$, $f_j(c|_{S_j}) = c_i$. The sets $S_1, \ldots, S_t$ are called* repair groups.

As discussed more in Section 1.1 below, the t-DRGP naturally interpolates between many different notions of locality. The t-DRGP is well-studied both when $t = O(1)$ is small (where it is related to Locally Repairable Codes and nearly equivalently to Private Information Retrieval Codes) and $t = \Omega(N)$ is large (where it is equivalent to Locally Correctable Codes). For this reason, it is natural to study the t-DRGP when t is intermediate; for example, when $t = N^a$ for $a \in (0, 1)$. In this case, it is possible for the size of the code $|\mathcal{C}|$ to be quite large: more precisely, it is possible for the *rate* $R = \frac{\log_{|\Sigma|} |\mathcal{C}|}{N}$ to approach 1 (notice that we always have $|\mathcal{C}| \leq |\Sigma|^N$, hence we always have $R \leq 1$). Thus, the goal is to understand exactly how quickly the rate can approach 1. That is, given t, how small can the *redundancy* $N - RN$ be?

Several works have tackled this question, and we illustrate previous results in Figure 1. Our main result is that lifted multiplicity codes improve on the best-known trade-offs for all super-constant $t \leq \sqrt{N}$.

Contributions

We summarize the main contributions of this work below.

1. For $t \leq \sqrt{N}$, we construct codes with the t-DRGP and redundancy at most
$$O\left(t^{\log_2(3)-1}\sqrt{N}\right) \approx O\left(t^{0.585}\sqrt{N}\right).$$
This gives the best known construction for all t so that $t = \omega(1)$ and $t \leq \sqrt{N}$; the only previous result that held non-trivially for a range of t was redundancy $O(t\sqrt{N})$ [4, 2, 1] and our result also surpasses the specialized bound for $t = N^{1/4}$ of [5]. Moreover, both our argument and our construction are quite clean.
2. We give a new analysis of bivariate lifts of multiplicity codes. Both multiplicity codes and lifted codes have been studied before (even in the context of the t-DRGP), but to the best of our knowledge the only work to consider lifted multiplicity codes is [26]. That work studies m-variate lifts of multiplicity codes, where m is large; its goal is to obtain new constructions of high-rate locally correctable codes. In the context of our discussion, this corresponds to the t-DRGP when $t = N^{0.99}$. In contrast, for bivariate lifts, we are able to obtain more refined bounds which lead to improved results for the t-DRGP when $t \leq \sqrt{N}$.

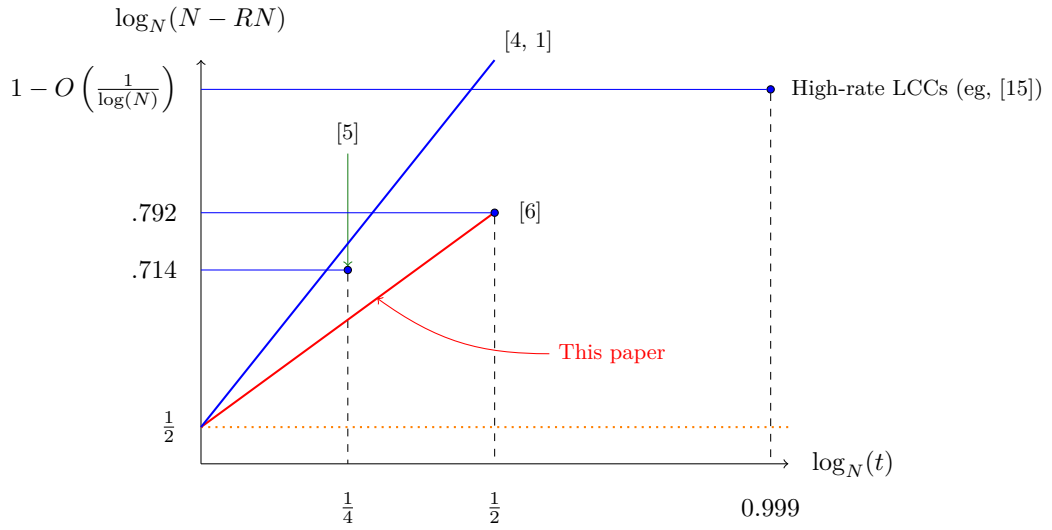


Figure 1 The best trade-offs known between the number t of disjoint repair groups and the redundancy $N - RN$. Blue points and lines indicate upper bounds (possibility results), and the red line indicates our upper bound. The best lower bound (impossibility result) available is that we must have $\log_N((1 - R)N) \geq 1/2$ for any $t \geq 2$, and this is shown as the dotted orange line.

Organization

In the remainder of the introduction, we survey related work and give an overview of our approach. In Section 2, we give the formal definitions about polynomials and derivatives that we need. In Section 3, we formally define lifted multiplicity codes. In Section 4, we prove that lifted multiplicity codes have high rate, and in Section 5, we prove that they have the t-DRGP, which gives rise to our main theorem, Theorem 2.

1.1 Background and Related Work

1.1.1 Disjoint Repair Groups

The t-DRGP and related notions have been studied both implicitly and explicitly across several communities. When $t = O(1)$ is small, several notions related to the t-DRGP have been studied, motivated primarily by distributed storage. These include codes for Private Information Retrieval (PIR) [4, 2, 1], Locally Repairable Codes (LRCs) with availability [23, 18, 21, 22], and batch codes [10, 19, 1]. In more detail, PIR codes are basically equivalent to codes with the DRGP, with the slight difference that PIR codes generally require that every message symbol should be recoverable by many disjoint repair groups, rather than every codeword symbol. LRCs with availability are a slightly stronger notion where the disjoint repair groups should additionally be small. Batch codes are also a slightly stronger notion, where one should be able to access any t-tuple of symbols (possibly with repetition) in t disjoint ways. We refer the reader to [20] for a survey of these notions.

To see why the t-DRGP might be relevant for distributed storage, consider a setting where some data is encoded as $c \in \mathcal{C}$, and then each c_i is sent to a separate server. If server i is later unavailable, we might want to reconstruct c_i without contacting too many other servers. This can be done if each symbol has one small repair group; this is the defining

property of LRCs. Now suppose that several (say, $t-1$) servers are unavailable. If $\mathcal{C}$ has the t-DRGP then all $t-1$ unavailable symbols can be locally reconstructed: each node has at least t disjoint repair groups and at most $t-1$ of them have been compromised.

On the other hand, when $t = \Omega(N)$ is large, the t-DRGP has been studied in the context of Locally Decodable Codes and Locally Correctable Codes (LDCs/LCCs). In fact, the $\Omega(N)$-DRGP is equivalent to a constant-query LCC, and the notion has been used to prove impossibility results for such codes [11, 24].

Because of these motivations, there are several constructions of t-DRGP codes for a wide range of t; we illustrate the relevant ones in Figure 1. In the context of coded PIR, [4, 2, 1] give constructions of t-DRGP codes with redundancy $O(t\sqrt{N})$. This is known to be tight for $t = 2$ [17, 25], but no better lower bound is known.[1] When $t = \Omega(N)$ is very large, constructing codes with the t-DRGP is equivalent to constructing constant-query LCCs, and it is known that the rate of the code must tend to zero [24]. On the other hand, for any $\epsilon > 0$, when $t = O(N^{1-\epsilon})$ is just slightly smaller, then work on high-rate LCCs [15, 6, 8, 14] (see also [1]) imply that there are codes with rate 0.99 (or any constant less than 1) with the t-DRGP.[2]

When $t = \sqrt{N}$, there are a few constructions known that beat the $O(t\sqrt{N})$ bound mentioned above, including difference-set codes (see, e.g., [16]) and, relevant for us, lifted parity-check codes [6]. These constructions achieve redundancy $N^{\log_4(3)} \approx N^{0.79}$ when $t = \sqrt{N}$. In Appendix B, we include a new proof of the fact that the lifted codes of [6] have this redundancy using a dual view of lifted codes.

When $t < \sqrt{N}$, there is only one construction known which beats the $O(t\sqrt{N})$ bound, due to [5]. For the special case of $t = N^{1/4}$, they give a construction based on "partially lifted codes" which has redundancy $O(N^{0.72}) = O(t^{0.88}\sqrt{N})$.

1.1.2 Lifting and multiplicity codes

Lifted multiplicity codes are based on lifted codes and multiplicity codes, both of which have a long history in the study of locality in error correcting codes.

1.1.2.1 Lifted Codes

Lifting was introduced by Guo, Kopparty and Sudan in [6]. The basic idea can be illustrated by Reed-Solomon (RS) codes. An RS code of degree d over $\mathbb{F}_q$ is the code

$$\mathsf{RS}_{d,q} = \{(f(x_1), \ldots, f(x_q)) : f \in \mathbb{F}_q[X], \deg(f) < d\},$$

where $x_1, \ldots, x_q$ are the elements of $\mathbb{F}_q$. There is a natural multi-variate version of RS codes, known as Reed-Muller codes:

$$\mathsf{RM}_{d,q,m} = \{(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_{q^m})) : f \in \mathbb{F}_q[X_1, \ldots, X_m], \deg(f) < d\},$$

where $\mathbf{x}_1, \ldots, \mathbf{x}_{q^m}$ are the elements of $\mathbb{F}_q^m$. Reed-Muller codes have a very nice locality property, which is that the restriction of a RM codeword to a line in $\mathbb{F}_q^m$ yields an RS codeword. This fact has been taken advantage of extensively in applications like local decoding, local list-decoding and property testing. However, RM codes have a downside,

[1] When the size s of the repair groups is bounded, it is known that the redundancy must be at least $\Omega(N \ln(t)/s)$ [22].

[2] In fact we may even take ϵ slightly sub-constant using the construction of [14].

which is that if $d < q$ (required for the above property to kick in), they have very low rate. With this inspiration, we could ask for the set $\mathcal{C}$ which contains evaluations of *all* m-variate polynomials which restrict to low-degree univariate polynomials on every line. Surprisingly, [6] showed that this set $\mathcal{C}$ can be much larger than the corresponding RM code! This code $\mathcal{C}$ is called a *lifted* Reed-Solomon code, and the main structural result of [6] is that $\mathcal{C}$ is the span of the monomials whose restrictions to lines are low-degree. This property is key when analyzing the rate of these codes. Moreover [6] showed that this is the case when we begin with *any* affine-invariant code, not just RS codes.

The original motivation for lifted codes was to construct LCCs, but [6] actually also give a code with the $\sqrt{N}$-DRGP, mentioned above; we give an alternate proof that this construction has the $\sqrt{N}$-DRGP in Appendix B. A variant of lifting was also used in [5] to construct $N^{1/4}$-DRGP codes; however, the analysis of this construction is quite brittle and seems difficult to extend to non-trivial constructions for $t \neq N^{1/4}$.

1.1.2.2 Multiplicity Codes

Multiplicity codes were introduced by Kopparty, Saraf and Yekhanin [15] with the goal of constructing high-rate LCCs. The basic idea of multiplicity codes is to get around the low rate of RM codes discussed above in a different way, by appending derivative information to allow for higher-degree polynomials. That is, it is not useful to have an RS code with degree $d > q$, since $x^q = x$ for any $x \in \mathbb{F}_q$. However, if we replace the single evaluation $f(x)$ with a vector of evaluations $(f(x), f^{(1)}(x), \ldots, f^{(r-1)}(x))$, where $f^{(i)}$ denotes the i'th derivative, then it does make sense to take $d > q$. The m-variate multiplicity code $\mathsf{Mult}_{d,q,m,r}$ of degree d and order r over $\mathbb{F}_q$ is then defined similarly to $\mathsf{RM}_{d,q,m}$:

$$\mathsf{Mult}_{d,q,m,r} = \left\{ (f^{(<r)}(\mathbf{x}_1), \ldots, f^{(<r)}(\mathbf{x}_{q^m})) \,:\, f \in \mathbb{F}_q[X_1, \ldots, X_m], \deg(f) < d \right\},$$

where $f^{(<r)}(\mathbf{x}) \in \mathbb{F}_q^{\binom{m+r-1}{m}}$ is a vector containing all of the partial derivatives of f of order less than r, evaluated at $\mathbf{x}$. Since their introduction, multiplicity codes have found several uses beyond LCCs, including list-decoding [12, 7], and have even been used to explicitly construct codes with the t-DRGP [1].

1.1.2.3 Lifted Multiplicity Codes

To the best of our knowledge, the only work to study lifted multiplicity codes is the work of Wu [26]. The goal of that work is to obtain versions of multiplicity codes which are still high-rate LCCs but which require lower-order derivatives than the construction of [15]. The main result is that lifted multiplicity codes of rate $1 - \alpha$ are LCCs with locality N^ϵ (this corresponds roughly to having the t-DRGP with $t = O(N^{1-\epsilon})$). However, since the number of variables in the lift is large, it is hard to get a very precise handle on the codimension, and in particular the codimension of the code in that work is not shown to be $o(N)$.

In contrast, we study bivariate lifts of multiplicity codes. By focusing only on bivariate lifts, we are able to get a more precise handle on the codimension of lifted multiplicity codes, which gives results for the t-DRGP for $t \leq \sqrt{N}$.

We note that the construction in [26] is similar to the construction presented here. Since this construction is somewhat non-trivial (for reasons discussed below), we include the details.

1.2 Our approach

We study lifted multiplicity codes to obtain improved constructions of codes with the t-DRGP. We focus on bivariate lifts in this paper in order to obtain codes with t-DRGP for $t \leq \sqrt{N}$. We expect that lifted multiplicity codes in more than two variables also give better codes for the t-DRGP when $t > \sqrt{N}$.

1.2.1 Definition of lifted multiplicity codes

It is not immediately obvious how to apply lifting (and in particular, the nice characterization of it developed in [6] as the span of "good" monomials) to univariate multiplicity codes. We first note that the univariate multiplicity code $\mathsf{Mult}_{d,q,1,r} \subseteq \left(\mathbb{F}_q^r\right)^q$ does not fit the affine-invariant framework of [6], so their results do not immediately apply. Instead, we might try to define the bivariate lift of $\mathsf{Mult}_{d,q,1,r}$ as the set of vectors $(f^{(<r)}(\mathbf{x}_1), \ldots, f^{(<r)}(\mathbf{x}_{q^2}))$ for all polynomials f so that every restriction of f to a line agrees with some polynomial of degree less than d on its first $r-1$ derivatives; that is, the restriction of f is *equivalent up to order* r to a polynomial of degree less than d. This works, but there are two non-trivial things to deal with.

1. First, in order to get a handle on the rate of the code, as in [6] we show that the set of valid polynomials f includes the span of a large set of "good" monomials. In contrast to [6], the good monomials in this work do not span the entire code. However, lower bounding the number of good monomials, which in turns gives a lower bound on the rate of the code, turns out to be enough for our results.
2. Second, we need to take some care about what monomials we allow. With lifted RS codes, one only allows monomials X^aY^b with individual degrees $a, b < q$; otherwise, we could have multiple monomials which correspond to the same codeword which leads to problems if we are counting monomials in order to understand the dimension of the code. As we show in Lemma 14, it turns out that with multiplicity codes, we should only allow monomials X^aY^b with $\lfloor a/q \rfloor + \lfloor b/q \rfloor < r$; otherwise, we would have multiple monomials the correspond to the same codeword and this would create similar problems.

Dealing with these issues leads us to the final code and rate analysis, where we define the lifted multiplicity code to be all polynomials spanned by monomials X^aY^b with $\lfloor a/q \rfloor + \lfloor b/q \rfloor < r$, such that the restriction of the polynomial to a line is equivalent up to order r to some univariate polynomial of degree less than d. We then lower bound the number of evaluations of monomials in this code, giving a lower bound on the rate. We note that the work [26] considers a similar construction.

1.2.2 Lifted multiplicity codes have the t-DRGP

In Corollary 21 we give a lower bound on the number of (q, r, d)-good monomials, and this leads to a lower bound on the dimension of the lifted multiplicity code; crucially, this can be quite a bit bigger than the dimension of the corresponding multivariate multiplicity code.

Finally, we observe that lifted multiplicity codes have the t-DRGP for a range of values of t. Similarly to previous constructions based on multivariate polynomial codes, the disjoint repair groups to recover the symbol $f^{(<r)}(\mathbf{x})$ are given by disjoint collections of lines through $\mathbf{x}$. More precisely, the values $f^{(<r)}(\mathbf{y})$ for the set of $\mathbf{y}$ that lie on r distinct lines through $\mathbf{x}$ can be used to recover $f^{(<r)}(\mathbf{x})$. Thus, the number of disjoint repair groups is $q/r = \sqrt{N}/r$. By adjusting r, we obtain the trade-off shown in Figure 1. Our main theorem is as follows.

▶ **Theorem 2.** *For $q = 2^\ell$ and $r = 2^{\ell'}$ with $1 \le \ell' \le \ell$, there exists a code $\mathcal{C}$ over $\mathbb{F}_q^{\binom{r+1}{2}}$ with the following properties.*

- *The length of the code is q^2.*
- *The rate of the code is at least*

$$1 - \frac{3r^{\log_2(8/3)}q^{\log_2(3)}}{\binom{r+1}{2}q^2},$$

so that the redundancy is at most

$$\frac{3r^{\log_2(8/3)}q^{\log_2(3)}}{\binom{r+1}{2}}.$$

- *The code has the q/r-disjoint repair group property.*

As a remark, our techniques can also recover any symbol from any one of its repair groups in polynomial time. For any $\gamma \in [0, 1]$, choosing $q = 2^\ell$ and $r = 2^{\ell'}$ with $\gamma \approx \ell'/\ell$ gives a code with length $N = q^2$ and redundancy at most

$$6N^{\log_4(3)-\gamma(1-\log_4(8/3))}$$

with the $N^{(1-\gamma)/2}$-DRGP. This is made formal in the following corollary.

▶ **Corollary 3.** *For any $\epsilon > 0$, there are infinitely many N so that, for $t = \lfloor N^\epsilon \rfloor$, there exists a code of length N which has the t-DRGP and redundancy at most $6t^{\log_2(3)-1}\sqrt{N}$.*

We note that Theorem 2 also yields results for constant t, not just for $t = N^\epsilon$ as presented in Corollary 3. For example, by setting $r = q/2$ we obtain a code with the 2-DRGP and redundancy at most $9\sqrt{N}$. The constant 9 is not optimal here (the optimal constant for $t = 2$ is known to be $\sqrt{2}$ [17]), but to the best of our knowledge, Theorem 2 does yield the best known bounds for any super-constant t.

2 Preliminaries

In this section, we introduce the background we need on polynomials and derivatives over finite fields. Throughout this paper, we assume that q is a power of 2. Let $\mathbb{F}_q$ denote the finite field of order q, and let $\mathbb{F}_q^*$ denote its multiplicative subgroup.

If a and b are nonnegative integers with binary representations $a = \overline{a_{\ell-1}\cdots a_0}$ and $b = \overline{b_{\ell-1}\cdots b_0}$, then we write $a \le_2 b$ if $a_i \le b_i$ for $i = 0, \ldots, \ell - 1$. If a is an integer, let $(a \mod c)$ denote the element of $\{0, \ldots, c-1\}$ congruent to a mod c. We write $a \le_2^\ell b$ if $(a \mod 2^\ell) \le_2 (b \mod 2^\ell)$.

As in [6], we use Lucas's theorem.

▶ **Proposition 4** (Lucas's theorem). *Let p be a prime and $a = \overline{a_{\ell-1}\cdots a_0}, b = \overline{b_{\ell-1}\cdots b_0}$ be written in base p. Then*

$$\binom{a}{b} \equiv \prod_{i=0}^{\ell-1} \binom{a_i}{b_i} \mod p \tag{1}$$

In particular, if $p = 2$, then $\binom{a}{b} \equiv 1 \mod p$ if and only if $a \le_2 b$.

2.1 Polynomials and derivatives

For a vector $\mathbf{i} = (i_1, \ldots, i_m)$ of nonnegative integers, its *weight*, denoted $\mathrm{wt}(\mathbf{i})$, equals $\sum_{k=1}^{m} i_k$. For a field $\mathbb{F}$, let $\mathbb{F}[X_1, \ldots, X_m] = \mathbb{F}[\mathbf{X}]$ be the ring of polynomials in the variables $X_1, \ldots, X_m$ with coefficients in $\mathbb{F}$. For a vector of nonnegative integers $\mathbf{i} = (i_1, \ldots, i_m)$ and a vector $\mathbf{X} = (X_1, \ldots, X_m)$ of variables, let $\mathbf{X}^{\mathbf{i}}$ denote the monomial $\prod_{j=1}^{m} X_j^{i_j} \in \mathbb{F}[\mathbf{X}]$, and for a vector $\mathbf{a} = (\alpha_1, \ldots, \alpha_m) \in \mathbb{F}^m$, let $\mathbf{a}^{\mathbf{i}}$ denote the value $\prod_{j=1}^{m} \alpha_j^{i_j}$, where $0^0 \stackrel{\text{def}}{=} 1$. For nonnegative vectors $\mathbf{i} = (i_1, \ldots, i_m)$ and $\mathbf{j} = (j_1, \ldots, j_m)$, we write $\mathbf{i} \leq \mathbf{j}$ if $i_k \leq j_k$ for all k. We also write $\binom{\mathbf{i}+\mathbf{j}}{\mathbf{i}}$ to denote $\prod_{k=1}^{m} \binom{i_k+j_k}{i_k}$. For nonnegative vector $\mathbf{i}$, we let $[\mathbf{X}^{\mathbf{i}}]P(\mathbf{X})$ denote the coefficient of $\mathbf{X}^{\mathbf{i}}$ in the polynomial $P(\mathbf{X})$.

We will use Hasse derivatives, a notion of derivatives over finite fields:

▶ **Definition 5** (Hasse derivatives). *For $P(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]$ and a nonnegative vector $\mathbf{i}$, the $\mathbf{i}$-th (Hasse) derivative of P, denoted $P^{(\mathbf{i})}(\mathbf{X})$ or $D^{(\mathbf{i})}P(\mathbf{X})$, is the coefficient of $\mathbf{Z}^{\mathbf{i}}$ in the polynomial $\tilde{P}(\mathbf{X}, \mathbf{Z}) \stackrel{\text{def}}{=} P(\mathbf{X} + \mathbf{Z}) \in \mathbb{F}[\mathbf{X}, \mathbf{Z}]$. Thus,*

$$P(\mathbf{X} + \mathbf{Z}) = \sum_{\mathbf{i}} P^{(\mathbf{i})}(\mathbf{X})\mathbf{Z}^{i}. \tag{2}$$

For $\mathbf{x} \in \mathbb{F}_q^m$ and $P(X) \in \mathbb{F}_q[\mathbf{X}]$, we use the notation $P^{(<r)}(\mathbf{x}) \in \mathbb{F}_q^{\binom{m+r-1}{m}}$ to denote the vector containing $P^{(\mathbf{i})}(\mathbf{x})$ for all $\mathbf{i}$ so that $\mathrm{wt}(\mathbf{i}) < r$. We record a few useful (well-known) properties of Hasse derivatives below (see [9]).

▶ **Proposition 6** (Properties of Hasse derivatives). *Let $P(\mathbf{X}), Q(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]^m$ and let $\mathbf{i}, \mathbf{j}$ be vectors of nonnegative integers. Then*

1. $P^{(\mathbf{i})}(\mathbf{X}) + Q^{(\mathbf{i})}(\mathbf{X}) = (P + Q)^{(\mathbf{i})}(\mathbf{X})$.
2. $(P \cdot Q)^{(\mathbf{i})}(\mathbf{X}) = \sum_{0 \leq \mathbf{e} \leq \mathbf{i}} P^{(\mathbf{e})}(\mathbf{X}) \cdot Q^{(\mathbf{i}-\mathbf{e})}(\mathbf{X})$.
3. $(P^{(\mathbf{i})})^{(\mathbf{j})}(\mathbf{X}) = \binom{\mathbf{i}+\mathbf{j}}{\mathbf{i}} P^{(\mathbf{i}+\mathbf{j})}$.

Using the above, we obtain the following useful derivative computation, and we provide a proof in Appendix A for completeness.

▶ **Proposition 7.** *Let $1 \leq r < q$ with q a power of 2, and let $P(X) = (X^q - X)^r$. Then,*

$$P^{(i)}(X) = \begin{cases} \binom{r}{i}(X^q - X)^{r-i} & 0 \leq i \leq r \\ 0 & i > r \end{cases} \tag{3}$$

2.2 Polynomial local recovery

A key property exploited by earlier work on multiplicity codes [15, 13] is that $f^{(<r)}(\mathbf{x})$ can be recovered from $f^{(<q)}(\mathbf{y})$ for $\mathbf{y}$ that lie on a collection of lines through $\mathbf{x}$. More precisely, let $\mathcal{L}_m$ be the set of lines $L(T)$ of the form $\mathbf{a}T + \mathbf{b}$ with $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^m$. Given a multivariate polynomial $P(\mathbf{X}) \in \mathbb{F}_q[X_1, \ldots, X_m]$, if L is the line $\mathbf{a}T + \mathbf{b}$, let $P_L(T) \in \mathbb{F}_q[T]$ denote the univariate polynomial $P(\mathbf{a}T + \mathbf{b})$. Let $\mathcal{L}$ be the set of lines in $\mathbb{F}_q^2$ of the form $L(T) = (T, \alpha T + \beta)$ for $\alpha, \beta \in \mathbb{F}_q$.

For simplicity – and because it is enough for our application to the t-DRGP – we will consider only bivariate polynomials in this paper, although (see for example [13]) the same basic idea works for any m. We will further specialize to lines in $\mathcal{L}$ – that is, lines of the form $L(T) = (T, \alpha T + \beta)$ – because it will simplify some computations later in the paper. With these restrictions, we can specialize Equation (4) of [13] to obtain the following relationship between the derivatives of $P_L(T)$ and the derivatives of $P(X, Y)$.

▶ **Lemma 8** (Follows from, e.g., [15, 13]). *Suppose that $L_1, \ldots, L_r$ are r lines in $\mathcal{L}$ all passing through a point (γ, δ), with L_k being the line $(T, \alpha_k T + \beta_k)$. Then, for all polynomials $P(X, Y) \in \mathbb{F}_q[X, Y]$, the following matrix equality holds for all $i = 0, \ldots, r-1$.*

$$\begin{bmatrix} P^{(i)}_{L_1}(\gamma) \\ P^{(i)}_{L_2}(\gamma) \\ \vdots \\ P^{(i)}_{L_{i+1}}(\gamma) \end{bmatrix} = \begin{bmatrix} \alpha_1^0 & \alpha_1^1 & \cdots & \alpha_1^i \\ \alpha_2^0 & \alpha_2^1 & \cdots & \alpha_2^i \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{i+1}^0 & \alpha_{i+1}^1 & \cdots & \alpha_{i+1}^i \end{bmatrix} \begin{bmatrix} P^{(i,0)}(\gamma, \delta) \\ P^{(i-1,1)}(\gamma, \delta) \\ \vdots \\ P^{(0,i)}(\gamma, \delta) \end{bmatrix}. \tag{4}$$

When lines $L_1, \ldots, L_k$ are distinct, the middle matrix in (4) is a Vandermonde matrix, and Vandermonde matrices are invertible in polynomial time. Hence, we immediately have the following corollary.

▶ **Corollary 9.** *Suppose that $L_1, \ldots, L_r$ are r distinct lines of the form $L_k(T) = (T, \alpha_k T + \beta_k)$ all passing through a point $(\gamma, \delta) \in \mathbb{F}_q^2$. For a polynomial $P(X, Y) \in \mathbb{F}_q[X, Y]$, given the polynomials $P_{L_1}(T), \ldots, P_{L_k}(T)$, the derivatives $P^{(\mathbf{i})}(\gamma, \delta)$ are uniquely determined and computable efficiently for all $\mathbf{i}$ such that $\mathrm{wt}(\mathbf{i}) < r$.*

3 Lifted multiplicity codes

In this section, we define lifted multiplicity codes. As noted in the introduction, we restrict our attention to bivariate codes because this is enough for our application to the t-DRGP. However, everything in this section extends to general m-variate codes. We define bivariate lifted multiplicity codes as the vectors $(f^{(<r)}(\mathbf{x}))_{\mathbf{x} \in \mathbb{F}_q^2}$ for polynomials $f(X)$ that live in the span of "good" monomials. In order to define these "good" monomials, we need a few more definitions.

3.1 Polynomial equivalence

We first define a notion of polynomial equivalence.

▶ **Definition 10.** *We say that two univariate polynomials $A(X), B(X) \in \mathbb{F}_q[X]$ are* equivalent up to order r, *written $A \equiv_r B$, if $A^{(i)}(\gamma) = B^{(i)}(\gamma)$ for all $i = 0, \ldots, r-1$ and $\gamma \in \mathbb{F}_q$.*

It is easy to see that the above definition does in fact give an equivalence relation. There is a simple way to characterize this equivalence.

▶ **Lemma 11.** *For $A(X), B(X) \in \mathbb{F}_q[X]$ we have $A(X) \equiv_r B(X)$ if and only if $(X^q - X)^r | A(X) - B(X)$.*

Proof. By considering the polynomial $A(X) - B(X)$, it suffices to prove $A(X)$ is equivalent to the zero polynomial up to order r if and only if $(X^q - X)^r | A(X)$. If $A(X) = (X^q - X)^r C(X)$ for some polynomial $C(X) \in \mathbb{F}_q[X]$, then, by part 2 of Proposition 6 and Proposition 7, for $0 \le i < r$, we have $X^q - X | A^{(i)}(X)$, so $A^{(i)}(\gamma) = 0$ for all $0 \le i < r$ and all $\gamma \in \mathbb{F}_q$, so $A(X) \equiv_r 0$.

Conversely, suppose that $A(X) \equiv_r 0$. By the definition of Hasse derivatives, we have $A(X) = A(\gamma + (X - \gamma)) = \sum_i A^{(i)}(\gamma)(X - \gamma)^i$. Since $A^{(i)}(\gamma) = 0$ for $i = 0, \ldots, r-1$, we have $(X - \gamma)^r | A(X)$. Thus is true for all γ, so $\prod_\gamma (X - \gamma)^r | A(X)$, so $(X^q - X)^r | A(X)$. ◀

Lemma 11 gives the following corollary.

▶ **Lemma 12.** *Let q be a power of 2 and $r \geq 1$. For every univariate polynomial $A(X)$, there exists a unique degree-at-most $rq-1$ polynomial $B(X)$ such that $A(X) \equiv_r B(X)$. Furthermore, if r is a power of 2, then for all a such that $\deg A - (qr-r) < a < qr$, we have $[X^a]A(X) = [X^a]B(X)$.*

Proof. For existence of $B(X)$, note that, by Lemma 11, we can subtract an appropriate multiple of $(X^q - X)^r$ from $A(X)$ to obtain the desired $B(X)$. For uniqueness of $B(X)$, suppose that $B_1(X)$ and $B_2(X)$ are equivalent to $A(X)$ up to order r and are of degree at most $rq-1$. By Lemma 11, we have $(X^q-X)^r | B_1(X) - B_2(X)$. Additionally, $B_1(X) - B_2(X)$ has degree at most $rq-1$, so $B_1(X) - B_2(X) = 0$.

Now suppose r is a power of 2. Then $(X^q - X)^r = X^{rq} + X^r$. Above, to obtain $B(X)$ from $A(X)$, we need only to subtract terms of the form $X^{qr}+X^r, X^{qr+1}+X^{r+1}, \ldots, X^{\deg A}+X^{\deg A - qr + r}$. Thus, for a such that $\deg A - qr + r < a < qr$, the coefficients of X^a in $A(X)$ and $B(X)$ are equal. ◀

3.2 Type-r polynomials

Define the order-r evaluation map $\mathrm{eval}_{q,r} : \mathbb{F}_q[X,Y] \to \left(\mathbb{F}_q^{\binom{r+1}{2}}\right)^{q^2}$ by

$$\mathrm{eval}_{q,r}(P) := (P^{(<r)}(\mathbf{x}))_{\mathbf{x}\in\mathbb{F}_q^2}, \tag{5}$$

We will want to restrict our attention to a subset of monomials $M(X,Y) = X^aY^b$ whose order-r evaluations $\mathrm{eval}_{q,r}(M)$ form a basis for the space $\{\mathrm{eval}_{q,r}(P) : P \in \mathbb{F}_q[X,Y]\}$. To that end, we introduce the following definition.

▶ **Definition 13** (Type-r monomials)**.** *Call a monomial X^aY^b type-r if $\lfloor a/q \rfloor + \lfloor b/q \rfloor \leq r-1$. Let $\mathcal{F}_{q,r}$ be the family of polynomials $P \in \mathbb{F}_q[X,Y]$ that are spanned by type-r monomials.*

It is easy to see that $\mathcal{F}_{q,r}$ is a dimension $\binom{r+1}{2}q^2$ vector space over $\mathbb{F}_q$. We now show that the type-r polynomials form a basis for bivariate polynomials, up to order r equivalence.

▶ **Lemma 14.** *The evaluation map $\mathrm{eval}_{q,r} : \mathcal{F}_{q,r} \to \left(\mathbb{F}_q^{\binom{r+1}{2}}\right)^{q^2}$ is a bijection.*

Proof of Lemma 14. Since $\mathrm{eval}_{q,r}$ is a linear map and $\mathcal{F}_{q,r}$ and $\mathbb{F}_q^{\binom{r+1}{2}q^2}$ have the same $\mathbb{F}_q$ dimension, it suffices to prove the map has trivial kernel. We prove by induction.

Base Case: $r = 1$. Suppose $P \in \mathcal{F}_{q,1}$ and $\mathrm{eval}_1(P)$ is the 0-vector. Then $P(X,Y) = 0$ for all X, Y. For any $\delta \in \mathbb{F}_q$, the polynomial $P(X,\delta) \in \mathbb{F}_q[X]$ has degree at most $q-1$ but has q roots, so the polynomial must be 0. Hence, $(Y-\delta) | P(X,Y)$ for all δ, so $Y^q - Y | P(X,Y)$, which implies $P = 0$. This proves that eval_1 has trivial kernel.

Inductive step. Assume $r \geq 1$ and $\mathrm{eval}_{q,r}$ has trivial kernel. We prove that eval_{r+1} has trivial kernel.

Assume $P(X,Y)$ is a polynomial spanned by type-$(r+1)$ monomials with all $\mathbf{i}$th derivatives equal to 0 for $\mathrm{wt}(\mathbf{i}) < r+1$. Let $\delta \in \mathbb{F}_q$ and $B_\delta(X) \stackrel{\text{def}}{=} P(X,\delta)$. Then, for $0 \leq i < r$, we have $B_\delta^{(i)}(\gamma) = B^{(i,0)}(\gamma,\delta) = 0$ for all $\gamma \in \mathbb{F}_q$. Hence, for all $\gamma \in \mathbb{F}_q$, we have $(X-\gamma)^r | B_\delta(X)$. Hence, $(X^q - X)^r | B_\delta(X)$. Since $\deg B_\delta(X) \leq \deg_X P(X,Y) < qr$ for all δ, we have $B_\delta(X) = 0$. Thus, $P(X,\delta)$ is the 0 polynomial for all δ, so $Y - \delta | P(X,Y)$ for all δ, so $Y^q - Y | P(X,Y)$. Hence, we may write $P(X,Y) = (Y^q - Y)Q(X,Y)$ for some polynomial $Q(X,Y) \in \mathbb{F}_q[X,Y]$.

As polynomial P is type-$(r+1)$, polynomial Q is type-r: if Q had a nonzero coefficient for X^aY^b with $\lfloor a/q \rfloor + \lfloor b/q \rfloor > r-1$, then the coefficient X^aY^{b+q} is nonzero in P, which is a contradiction. For all i, j with $i \geq 0, j \geq 1$ and $i + j \leq r$, we have

$$P^{(i,j)}(X,Y) \;=\; (Y^q - Y)Q^{(i,j)}(X,Y) - Q^{(i,j-1)}(X,Y). \tag{6}$$

Here we applied part 2 of Proposition 6 and the $r = 1$ case of Proposition 7. At every X and Y, the left side is 0 by assumption on P and the right side $Q^{(i,j-1)}(X,Y)$. We conclude that $Q^{(i',j')}$ evaluates to 0 everywhere for every nonnegative i' and j' satisfying $i' + j' \leq r-1$. Since Q is type-r, we have $Q = 0$ by the induction hypothesis, so $P = 0$. This completes the induction, completing the proof. ◀

3.3 Definition of lifted multiplicity codes

Finally we are ready to define lifted multiplicity codes, which we define as the set of evaluations $\mathrm{eval}_{q,r}(P)$ of polynomials whose restrictions to lines[3] are equivalent, up to order r, to a low degree polynomial:

▶ **Definition 15** (Lifted multiplicity codes, first definition). *The* (q, r, d) (bivariate) lifted multiplicity code *is a code* $\mathcal{C}$ *over alphabet* $\Sigma = \mathbb{F}_q^{\binom{r+1}{2}}$ *of length* q^2 *given by*

$$\mathcal{C} = \left\{ \mathrm{eval}_{q,r}(P) : \begin{array}{c} P \in \mathbb{F}_q[X,Y] \text{ and, for any } L(T) \in \mathcal{L}, \\ P(L(T)) \equiv_r Q(T) \text{ for some } Q \in \mathbb{F}_q[T] \text{ of degree at} \\ \text{most } d. \end{array} \right\}$$

Definition 15 is natural but difficult to get a handle on directly. Following the approach of previous work [6, 5], we observe that lifted multiplicity code contains the set of vectors $\mathrm{eval}_{q,r}(P)$ for P that lie in the span of a set of "good" monomials, which makes it easier to bound the rate. Informally, a monomial is (q, r, d)-good if its restriction along every line is equivalent, up to order r, to a polynomial of degree at most d.

▶ **Definition 16** ((q, r, d)-good monomials). *Call a monomial* $M_{a,b}(X,Y) = X^aY^b \in \mathbb{F}_q[X,Y]$ (q, r, d)-good *(or simply good, when* r *and* d *are understood) if it is type-*r *and for every line* $(T, \alpha T + \beta) \in \mathcal{L}$*, the univariate polynomial* $M_{a,b}(T, \alpha T + \beta)$ *is equivalent, up to order* r*, to polynomial of degree less than* d*, and call it* (q, r, d)-bad *otherwise.*

By definition all good monomials lie in our lifted multiplicity code, so to lower bound the rate of the code it suffices to lower bound the number of good monomials.

▶ **Lemma 17.** *Let* $\mathcal{C}$ *be the bivariate* (q, r, d) *lifted multiplicity code. Then, for every* (q, r, d)*-good monomial* $M(X,Y)$, $\mathrm{eval}_{q,r}(M) \in \mathcal{C}$*, and the rate of* $\mathcal{C}$ *is at least* $\frac{\#(q,r,d)\text{-good monomials}}{\binom{r+1}{2}q^2}$.

Proof. The first part follows from the definition of good monomial. For the second part, $\mathcal{C}$ is linear and the $\mathbb{F}_q$-span of all good monomials have pairwise distinct evaluations by Lemma 14, so $|\mathcal{C}| \geq q^{(\#(q,r,d)\text{-good monomials})}$. As $\mathcal{C}$ is a length q^2 code over an alphabet of size $|\Sigma| = q^{\binom{r+1}{2}}$, the rate is at least $\frac{\log|\mathcal{C}|}{q^2 \log|\Sigma|} = \frac{\#(q,r,d)\text{-good monomials}}{\binom{r+1}{2}q^2}$. ◀

[3] To simplify calculations, we consider restrictions to lines of the form $L(T) = (T, \alpha T + \beta)$. That is, we do not include lines of the form $L(T) = (\alpha, T)$.

▶ Remark 18. A previous version of this paper incorrectly asserted that every codeword of the lifted multiplicity code is spanned by good monomials. As observed by Nikita Polianskii, this is in fact not true. For example, when $r = 2$ and $d = 2q-1$, the monomials $X^{2q-2}Y$ and $X^{q-1}Y^q$ are not (q, r, d)-good as verified by the line (T, T), but their sum $X^{2q-2}Y + X^{q-1}Y^q$ is in the (q, r, d)-lifted multiplicity code: the restriction of the sum to a line $(T, \alpha T + \beta) \in \mathcal{L}$ has a T^{2q-1} coefficient of $\alpha + \alpha^q = 0$ and hence has degree strictly less than $d = 2q - 1$.

4 The rate of lifted multiplicity codes

In this section, we bound the rate (and hence, the redundancy) of lifted multiplicity codes. Our final result on the rate is Corollary 21 below, which implies that for r, q and d of an appropriate form, the lifted multiplicity code over order r and degree d over $\mathbb{F}_q$ has rate at least

$$1 - \frac{6}{r}\left(r - \frac{d}{q}\right)^{\log_2(4/3)}.$$

In the next section, we will choose $d = qr - r$, which will yield a code of rate $1 - \frac{6}{r}\left(\frac{r}{q}\right)^{\log_2(4/3)}$ and will give us Theorem 2. We begin with a lemma that will be useful.

▶ **Lemma 19.** *Let $s = 2^{\ell_s}$ and $q = 2^\ell$ with $\ell_s \le \ell$. The number of $a_1, b_1 \in \{0, 1, \ldots, q-1\}$ such that at least one of the following is true*

$$\begin{aligned} q - 1 - a_1 &\le_2^\ell b_1 \\ q - 2 - a_1 &\le_2^\ell b_1 \\ &\vdots \\ q - s - a_1 &\le_2^\ell b_1 \end{aligned} \tag{7}$$

is at most $2 \cdot 3^\ell \cdot (4/3)^{\ell_s} = 2 \cdot 3^\ell \cdot s^{\log_2(4/3)}$.

Proof. Suppose we write the numbers $(q-1-a_1 \mod q), (q-2-a_1 \mod q), \ldots, (q-s-a_1 \mod q)$ in binary with ℓ digits (possibly with leading zeros). As these number span 2^{ℓ_s} consecutive integers mod q, when written in this binary form, their most significant $\ell - \ell_s$ coordinates take on at most 2 values. Let $a_2 = \lfloor \frac{(q-1-a_1 \mod q)}{2^{\ell_s}} \rfloor$ and $b_2 = \lfloor \frac{b_1}{2^{\ell_s}} \rfloor$ so that $a_2, b_2 \in \{0, \ldots, 2^{\ell-\ell_s} - 1\}$, and a_2 and b_2 are the most significant $\ell - \ell_s$ coordinates of $(q - 1 - a_1 \mod q)$ and b_1, respectively, when written in ℓ-digit binary. Then if one of the equations of (7) is true, then we must have either $a_2 \le_2 b_2$ or $a_2 - 1 \le_2 b_2$. This gives at most $2 \cdot 3^{\ell-\ell_s}$ choices for the pair (a_2, b_2). Given a_2 and b_2, there are 2^{ℓ_s} choices for each of a_1 and b_1, for a total of at most $2 \cdot 3^{\ell-\ell_s} \cdot 4^{\ell_s}$ solutions to (7). ◀

▶ **Lemma 20.** *Let $r = 2^{\ell_r}$, $s = 2^{\ell_s}$ and $q = 2^\ell$ with $\ell_r, \ell_s \in \{1, \ldots, \ell - 1\}$. The number of $(q, r, rq - s)$-good monomials is at least $\binom{r+1}{2}4^\ell - 3rs^{\log_2(4/3)} \cdot 3^\ell$.*

Proof. The number of type-r monomials is $\binom{r+1}{2}q^2 = \binom{r+1}{2}4^\ell$. A monomial $M_{a,b}$ is $(q, r, rq - s)$-good if, for every $\alpha, \beta \in \mathbb{F}_q$, we have

$$M_{a,b,\alpha,\beta}(T) \stackrel{\text{def}}{=} T^a(\alpha T + \beta)^b = \sum_{i=0}^{b} \alpha^i \beta^{b-i} T^{a+i} \binom{b}{i}. \tag{8}$$

can be represented as a polynomial of degree less than $rq - s$. Next, we apply Lemma 12, which says that there is a unique polynomial $B(T)$ so that $\deg(B) \le rq - 1$ so that $B(T) \equiv_r$

$M_{a,b,\alpha,\beta}(T)$, and further that all of the coefficients $[T^c]B(T)$ for $\deg(M_{a,b,\alpha,\beta}) - (qr - r) < c < qr$ are equal to the corresponding coefficient of $B(T)$. The degree of the polynomial $M_{a,b,\alpha,\beta}$ is at most $(r+1)q - 2$, and

$$((r+1)q - 2) - (qr - r) = r + q - 2 < qr - s$$

for any allowed choice of q, r, s, so $[T^c]B(T) = [T^c]M_{a,b,\alpha,\beta}(T)$ for all c so that

$$qr - s \leq c \leq qr.$$

Thus, to show that $B(T)$ has degree less than $qr - s$, it suffices to show that the coefficients of $T^{qr-s}, T^{qr-s+1}, \ldots, T^{qr-1}$ in $M_{a,b,\alpha,\beta}$ are all zero.

Write $a = a_0 q + a_1$ and $b = b_0 q + b_1$ where $a_0 + b_0 \leq r - 1$ and $0 \leq a_1, b_1 \leq q - 1$. Note that if $a_0 + b_0 < r - 1$, then for $s' = 1, \ldots, s$ coefficient $[T^{rq-s'}]M_{a,b,\alpha,\beta}$ is always zero except possibly when $a_0 + b_0 = r - 2$ and $a_1 + b_1 \geq 2q - s$. This can happen for at most $\frac{rs^2}{2}$ pairs (a, b). Hence, for $a_0 + b_0 < r - 1$, there are $\leq \frac{rs^2}{2}$ bad monomials (a, b).

Now assume $a_0 + b_0 = r - 1$. For $s' = 1, \ldots, s$, the coefficient of $T^{rq-s'}$ in $T^a(\alpha T + \beta)^b$ is 0 if $rq - s' < a$ or $a + b < rq - s'$. Otherwise, the coefficient is

$$\alpha^{rq-s'-a}\beta^{b-rq+s'+a}\binom{b}{rq-s'-a} = \alpha^{rq-s'-a}\beta^{b-rq+s'+a}\binom{b_0q+b_1}{b_0q+q-s'-a_1}. \tag{9}$$

By Proposition 4, the binomial coefficient is nonzero (mod 2) if and only if $b_0 q + q - s' - a_1 \leq_2 b_0 q + b_1$, which, as q is a power of 2, happens only if $q - s' - a_1 \leq_2^\ell b_1$. Hence, if $a_0 + b_0 = r - 1$, the monomial $M_{a,b}$ is $(r, rq - s)$-bad only if some $s' = 1, \ldots, s$ satisfies $q - s' - a_1 \leq_2^\ell b_1$. Hence, by Lemma 19, for a fixed a_0, b_0 with $a_0 + b_0 = r - 1$, there are at most $2s^{\log_2(4/3)}3^\ell$ bad monomials $M_{a,b}$, so there are at most $r \cdot s^{\log_2(4/3)}3^\ell$ bad monomials $M_{a,b}$ over all a_0, b_0 with $a_0 + b_0 = r - 1$. As we showed, there are at most $\frac{rs^2}{2}$ bad monomials when $a_0 + b_0 < r - 1$. Hence, there are at least $\binom{r+1}{2}4^\ell - 2rs^{\log_2(4/3)}3^\ell - \frac{rs^2}{2} \geq \binom{r+1}{2}q^2 - 3rs^{\log_2(4/3)}q^{\log_2(3)}$ good monomials, as desired. ◀

Lemma 20 and Lemma 17 together imply Corollary 21, which in turn implies the informal result stated at the beginning of the section.

▶ **Corollary 21.** *Let $r = 2^{\ell_r}$, $s = 2^{\ell_s}$ and $q = 2^\ell$ with $\ell_r, \ell_s \in \{1, \ldots, \ell - 1\}$. A $(q, r, rq - s)$ lifted multiplicity code has rate at least $1 - 6r^{-1}s^{\log_2(4/3)}q^{\log_2(3/4)}$.*

▶ Remark 22. We apply Corollary 21 for $r = s \leq q$, giving that a lifted multiplicity code of rate at least $1 - 6r^{\log_2(2/3)}q^{\log_2(3/4)}$. By comparison [15], a 2-variate multiplicity code of order r evaluations of degree at most $rq - r$ polynomials over $\mathbb{F}_q$ has rate $\frac{\binom{rq-r+2}{2}}{\binom{r+1}{2}q^2} \leq 1 - \Omega(\frac{1}{r})$, which is smaller than the rate of lifted multiplicity codes for $r \ll q$.

5 Disjoint repair groups of lifted multiplicity codes

Finally, we prove Theorem 2, which we repeat below.

▶ **Theorem** (Theorem 2, restated). *Let $r = 2^{\ell_r}$ and $q = 2^\ell$ with $\ell_r < \ell$ and $\mathcal{C}$ be the $(q, r, rq - r)$ lifted multiplicity code.*

- *The length of the code is q^2.*
- *The rate of the code is at least $1 - 6r^{\log_2(2/3)}q^{\log_2(3/4)}$.*
- *The code has the q/r-disjoint repair group property.*

Proof. The first item follows from the definition of $\mathcal{C}$, and the second item is by Corollary 21. To see the third item, we show that, given a point $(\gamma, \delta) \in \mathbb{F}_q^2$, lines $L_1, \ldots, L_r$ passing through (γ, δ), and $P^{(<r)}(\mathbf{y})$ at all points $\mathbf{y}$ on the lines $L_1, \ldots, L_r$ except (γ, δ) itself, we can (efficiently) recover $P^{(<r)}(\gamma, \delta)$. This guarantees the q/r-disjoint repair group property, because we can group the q lines of $\mathcal{L}$ of the form $L(T) = (T, \alpha T + \beta)$ passing through (γ, δ) arbitrarily into groups of r, giving q/r disjoint repair groups. For any line L_k, the polynomial $P_{L_k}(T)$ has degree at most $rq-r-1$, as P is $(q, r, qr-r)$-good. By taking linear combinations of directional derivatives (Lemma 8), we can efficiently compute $P_{L_k}^{(i)}(\gamma')$ for every $i = 0, \ldots, r-1$, every $k = 1, \ldots, r$, and every $\gamma' \neq \gamma$. We can compute $P_{L_k}(T)$ using a generalization of polynomial interpolation. This can be done in $O(D \log D)$ time, where $D < rq$ is the degree of the polynomial (see e.g. [3]) Hence, by Corollary 9, from $P_{L_1}(T), \ldots, P_{L_k}(T)$, we can efficiently compute $P^{(i,j)}(\gamma, \delta)$ for all i, j with $0 \leq i + j \leq r - 1$. ◀

6 Conclusion

We conclude with some open questions.

1. We have shown that lifted multiplicity codes with redundancy $O(t^{0.585}\sqrt{N})$ have the t-DRGP for a range of $t \leq \sqrt{N}$. However, we do not know of any general lower bounds beyond the lower bound for $t = 2$ which implies that the redundancy must be at least $\Omega(\sqrt{N})$ for any t. Thus, it is an open question whether or not our bound is tight or whether one can do better.
2. Lifted multiplicity codes display better locality for the t-DRGP problem for $t \leq \sqrt{N}$; it is a natural question to ask whether they can be used for larger t, and in particular whether they could lead to improved constructions of locally correctable codes.

References

1 Hilal Asi and Eitan Yaakobi. Nearly optimal constructions of PIR and batch codes. *IEEE Transactions on Information Theory*, 65(2):947–964, 2019.

2 Simon R. Blackburn and Tuvi Etzion. PIR Array Codes with Optimal PIR Rate. *ArXiv e-prints*, July 2016. `arXiv:1607.00235`.

3 Francis Y Chin. A generalized asymptotic upper bound on fast polynomial evaluation and interpolation. *SIAM Journal on Computing*, 5(4):682–690, 1976.

4 Arman Fazeli, Alexander Vardy, and Eitan Yaakobi. Codes for distributed PIR with low storage overhead. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2852–2856. IEEE, 2015.

5 S. Luna Frank-Fischer, Venkatesan Guruswami, and Mary Wootters. Locality via Partially Lifted Codes. *CoRR*, abs/1704.08627, 2017. `arXiv:1704.08627`.

6 Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 529–540, 2013. `doi:10.1145/2422436.2422494`.

7 Venkatesan Guruswami and Carol Wang. Linear-algebraic list decoding for variants of Reed–Solomon codes. *IEEE Transactions on Information Theory*, 59(6):3257–3268, 2013.

8 Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Local correctability of expander codes. *Information and Computation*, 243:178–190, 2015.

9 James W. P. Hirschfeld, Gábor Korchmáros, and Fernando Torres. *Algebraic Curves over a Finite Field*. Princeton Series in Applied Mathematics. Princeton University Press, 2008.

10 Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Batch codes and their applications. In *Proceedings of the thirty-sixth annual ACM Symposium on the Theory of Computing*, STOC 2004, pages 262–271. ACM, 2004.

11 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the 32nd symposium on Theory of Computing*, STOC 2000, pages 80–86, 2000. doi:10.1145/335305.335315.

12 Swastik Kopparty. List-decoding multiplicity codes. *Theory of Computing*, 11(1):149–182, 2015.

13 Swastik Kopparty. Some remarks on multiplicity codes. *CoRR*, abs/1505.07547, 2015. arXiv:1505.07547.

14 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-correctable and locally-testable codes with sub-polynomial query complexity. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2016, pages 202–215. ACM, 2016.

15 Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *Journal of the ACM (JACM)*, 61(5):28, 2014.

16 Shu Lin and Daniel J Costello. *Error control coding*. Pearson Education India, 2001.

17 Sankeerth Rao and Alexander Vardy. Lower Bound on the Redundancy of PIR Codes. *arXiv preprint arXiv:1605.01869*, 2016. arXiv:1605.01869.

18 Ankit Singh Rawat, Dimitris S Papailiopoulos, Alexandros G Dimakis, and Sriram Vishwanath. Locality and availability in distributed storage. In *2014 IEEE International Symposium on Information Theory*, pages 681–685. IEEE, 2014.

19 Ankit Singh Rawat, Zhao Song, Alexandros G Dimakis, and Anna Gál. Batch codes through dense graphs without short cycles. *IEEE Transactions on Information Theory*, 62(4):1592–1604, 2016.

20 Vitaly Skachek. Batch and PIR codes and their connections to locally repairable codes. In *Network Coding and Subspace Designs*, pages 427–442. Springer, 2018.

21 Itzhak Tamo and Alexander Barg. Bounds on locally recoverable codes with multiple recovering sets. In *2014 IEEE International Symposium on Information Theory*, pages 691–695. IEEE, 2014.

22 Itzhak Tamo, Alexander Barg, and Alexey Frolov. Bounds on the parameters of locally recoverable codes. *IEEE Transactions on Information Theory*, 62(6):3070–3083, 2016.

23 Anyu Wang and Zhifang Zhang. Repair locality with multiple erasure tolerance. *IEEE Transactions on Information Theory*, 60(11):6979–6987, 2014.

24 David P. Woodruff. *A Quadratic Lower Bound for Three-Query Linear Locally Decodable Codes over Any Field*, pages 766–779. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. doi:10.1007/978-3-642-15369-3_57.

25 Mary Wootters. Linear codes with disjoint repair groups. Not intended for publication, available at https://sites.google.com/site/marywootters/disjoint_repair_groups.pdf, 2016.

26 Liyasi Wu. Revisiting the multiplicity codes: A new class of high-rate locally correctable codes. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 509–513. IEEE, 2015.

A Proofs of polynomial facts

Proof of Proposition 7. By part 2 of Proposition 6,

$$P^{(i)}(X) = \sum_{j_1+\cdots+j_r=i} \prod_{k=1}^{r} D^{(j_k)}(X^q - X). \tag{10}$$

We have $D^{(1)}(X^q - X) = 1$ (the field has characteristic 2). For $2 \le i < q$, the ith derivative of $X^q - X$ is $\binom{q}{i} X^{q-i}$, which is 0, as $\binom{q}{i}$ is even by Proposition 4. The summand above is nonzero if and only if $j_1, j_2, \ldots, j_r \le 1$. When $i \le r$, this happens when i of the j_k's are 1 and $r-i$ are 0, which happens for $\binom{r}{i}$ choices of $(j_1, \ldots, j_r)$. This gives $P^{(i)}(X) = \binom{r}{i}(X^q - X)^{r-i}$ for $0 \le i \le r$. When $i > r$, some j_k is at least 2, in which case $P^{(r)}(X) = 0$ for $r < i < q$. ◀

Proof of Lemma 8. Let $\mathbf{a}_k$ denote the vector $(1, \alpha_k)$, and let $\mathbf{b}_k$ denote the vector $(0, \beta_k)$. By assumption, we have that $\mathbf{a}_k\gamma + \mathbf{b}_k = (\gamma, \delta)$. By the definition of Hasse derivatives, we have, for all $k = 1, \ldots, r$

$$\begin{aligned} P_{L_k}(T+Z) &= P(\mathbf{a}_k T + \mathbf{b}_k + \mathbf{a}_k Z) \\ &= \sum_{\mathbf{i}\in\mathbb{N}^2} P^{(\mathbf{i})}(\mathbf{a}_k T + \mathbf{b}_k) \cdot (\mathbf{a}_k Z)^{\mathbf{i}} \\ &= \sum_{\mathbf{i}\in\mathbb{N}^2} P^{(\mathbf{i})}(\mathbf{a}_k T + \mathbf{b}_k) \cdot \mathbf{a}_k^{\mathbf{i}} Z^{\mathrm{wt}(\mathbf{i})} \\ P_{L_k}(T+Z) &= \sum_{i\geq 0} P_{L_k}^{(i)}(T) Z^i \end{aligned} \tag{11}$$

Hence, for all $i \geq 0$ and $k = 1, \ldots, r$, we have

$$P_{L_k}^{(i)}(T) = \sum_{\mathbf{i}:\mathrm{wt}(\mathbf{i})=i} P^{(\mathbf{i})}(\mathbf{a}_k T + \mathbf{b}_k)\mathbf{a}_k^{\mathbf{i}} \tag{12}$$

By plugging in $T = \gamma$, we have for all $i \geq 0$ and $k = 1, \ldots, r$,

$$P_{L_k}^{(i)}(\gamma) = \sum_{\mathbf{i}:\mathrm{wt}(\mathbf{i})=i} P^{(\mathbf{i})}(\gamma, \delta)\mathbf{a}_k^{\mathbf{i}}. \tag{13}$$

Rewriting this in matrix form gives the desired result. ◀

B Lifted codes via dual codes

It was shown in [6] that bivariate lifted parity-check codes over $\mathbb{F}_q$, where $q = 2^\ell$, have co-dimension 3^ℓ. Here, we give an alternative proof using dual codes. The techniques in this proof are not directly related to the techniques that we used in the main body of the paper, but we found this alternative proof illuminating so we include it.

Let $q = 2^\ell$. Recall $\mathcal{L}$ is the set of lines expressible as $L(T) = (T, \alpha T + \beta)$ where $\alpha, \beta \in \mathbb{F}_q$. One way to think about codes with locality is by considering their dual code. If the code is a subset of $\mathbb{F}_q^{q\times q}$, then the dual code corresponds to lines of repair groups. Given a line $L(T)$ in $\mathcal{L}$, define the corresponding dual codeword:

$$(c_L^\perp)_{ij} \stackrel{\text{def}}{=} \begin{cases} 1 & (i,j) = L(t) \text{ for some } t \in \mathbb{F}_q \\ 0 & \text{o/w} \end{cases} \tag{14}$$

Let

$$V_{\mathcal{L}} \stackrel{\text{def}}{=} \mathrm{span}\left\{c_L^\perp : L \in \mathcal{L}\right\}. \tag{15}$$

Note that $V_{\mathcal{L}}$ is spanned by 4^ℓ elements, so the trivial bound on the dimension is 4^ℓ. We give the following improved bound, matching the analysis of [6].

▶ **Lemma 23.** *The subspace $V_{\mathcal{L}}$ has dimension at most 3^ℓ.*

Proof. A codeword $c_L^\perp$ is the evaluation of the following polynomial on $\mathbb{F}_q^{q\times q}$:

$$P_L(X, Y) \stackrel{\text{def}}{=} \prod_{\beta\neq\beta_L} (\alpha_L X + \beta - Y). \tag{16}$$

If $(X, Y) \notin L$, then the polynomial evaluates to 0 as $Y - \alpha_L X \neq \beta_L$, and otherwise it evaluates to

$$\prod_{\beta \neq \beta_L} (\beta - \beta_L) = \prod_{\beta \in \mathbb{F}_q^*} \beta = 1. \tag{17}$$

For $a + b \geq q$, the coefficient of $X^a Y^b$ in $P_L(X, Y)$ is 0. For $a + b \leq q$, the coefficient of $X^a Y^b$ in $P_L(X, Y)$ is

$$\binom{a+b}{a} \alpha_L^a (-1)^b \sum_{\substack{\beta_1, \ldots, \beta_{q-1-a-b} \in \mathbb{F}_q \\ \text{distinct}, \neq \beta_L}} \prod_{j=1}^{q-1-a-b} \beta_j. \tag{18}$$

This is because we first chose $a+b$ terms that contain X or Y, then choose which terms are X and which terms are Y, and this gives us a many α_L's and b many -1's, and we sum over the choices of the β terms that we choose. Hence, the only a, b such that $[X^a Y^b] P_L(X, Y) \neq 0$ for any L are the pairs (a, b) such that $a + b \leq q - 1$ and $\binom{a+b}{a} \equiv 1 \mod 2$. There are at most 3^ℓ pairs by Proposition 4. It follows that the polynomials $P_L(X, Y)$ are spanned by 3^ℓ monomials $X^a Y^b$ with $\binom{a+b}{a} \equiv 1 \mod 2$. Hence, the vector space $V_{\mathcal{L}}$ is spanned by 3^ℓ dual codewords in $\mathbb{F}_q^{q \times q}$ and thus has dimension at most 3^ℓ. ◀

Revision Notice

This is a revised version of the eponymous paper that appeared in the proceedings of APPROX/RANDOM 2019 (LIPIcs, volume 145, http://www.dagstuhl.de/dagpub/978-3-95977-125-2, published in September, 2019), in which an incorrect proposition (formerly Proposition 18) and the corresponding proof (formerly Appendix B) was deleted and in which the exposition was adjusted accordingly. Previously it was claimed that the lifted code is exactly the span of all good monomials. In fact the span of good monomial forms only a subset of the lifted code.

Dagstuhl Publishing – May 4, 2020.

Think Globally, Act Locally: On the Optimal Seeding for Nonsubmodular Influence Maximization

Grant Schoenebeck
University of Michigan, Ann Arbor, USA
http://web.eecs.umich.edu/~schoeneb/
schoeneb@umich.edu

Biaoshuai Tao
University of Michigan, Ann Arbor, USA
http://www-personal.umich.edu/~bstao/
bstao@umich.edu

Fang-Yi Yu
University of Michigan, Ann Arbor, USA
http://www-personal.umich.edu/~fayu/
fayu@umich.edu

Abstract

We study the r-complex contagion influence maximization problem. In the influence maximization problem, one chooses a fixed number of initial seeds in a social network to maximize the spread of their influence. In the r-complex contagion model, each uninfected vertex in the network becomes infected if it has at least r infected neighbors.

In this paper, we focus on a random graph model named the *stochastic hierarchical blockmodel*, which is a special case of the well-studied *stochastic blockmodel*. When the graph is not exceptionally sparse, in particular, when each edge appears with probability $\omega\left(n^{-(1+1/r)}\right)$, under certain mild assumptions, we prove that the optimal seeding strategy is to put all the seeds in a single community. This matches the intuition that in a nonsubmodular cascade model placing seeds near each other creates synergy. However, it sharply contrasts with the intuition for submodular cascade models (e.g., the independent cascade model and the linear threshold model) in which nearby seeds tend to erode each others' effects.

Finally, we show that this observation yields a polynomial time dynamic programming algorithm which outputs optimal seeds if each edge appears with a probability either in $\omega\left(n^{-(1+1/r)}\right)$ or in $o\left(n^{-2}\right)$.

2012 ACM Subject Classification Theory of computation → Social networks; Mathematics of computing → Random graphs

Keywords and phrases Nonsubmodular Influence Maximization, Bootstrap Percolation, Stochastic Blockmodel

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.39

Category RANDOM

Funding *Grant Schoenebeck*: National Science Foundation AitF #1535912 and CAREER #1452915
Biaoshuai Tao: National Science Foundation CAREER #1452915
Fang-Yi Yu: National Science Foundation AitF #1535912



© Grant Schoenebeck, Biaoshuai Tao, and Fang-Yi Yu;
licensed under Creative Commons License CC-BY



1 Introduction

A *cascade*, or a *contagion*[1], is a fundamental process on social networks: starting with some seed agents, the infection then spreads to their neighbors. A natural question known as influence maximization [4, 6, 18, 28] asks how to place a fixed number of initial seeds to maximize the spread of the resulting cascade. For example, which students can most effectively be enrolled in an intervention to decrease student conflict at a school [30]?

Influence maximization is extensively studied when the contagion process is submodular (a node's marginal probability of becoming infected after a new neighbor is infected decreases when the number of previously infected neighbors increases [22]). However, many examples of nonsubmodular contagions have been reported, including pricey technology innovations, the change of social behaviors, the decision to participate in a migration, etc [14, 27, 31, 2, 24]. In this case, a node's marginal influence may increase in the presence of other nodes – creating a kind of synergy.

Network structure and seed placement

We address this lack of understanding for nonsubmodular influence maximization by characterizing the optimal seed placement for certain settings which we will remark on shortly. In these settings, the optimal seeding strategy is to put all the seeds near each other. This is significantly different than in the submodular setting, where the optimal solutions tend to spread out the seeds, lest they erode each others' influence. We demonstrate this in the appendix (Sect. A) by presenting an example of submodular influence maximization where the optimal seeding strategy is to spread out the seeds.

This formally captures the intuition, as presented by Angell and Schoenebeck [1], that it is better to target one market to saturation first (act locally) and then to allow the success in this initial market to drive broader success (think globally) rather than to initially attempt a scattershot approach (act globally). It is also underscores the need to understand the particular nature of a contagion before blindly applying influence maximization tools.

We consider a well-known nonsubmodular cascade model which is also the most extreme one (in terms of nonsubmodularity), the r-complex contagion [19, 7, 8, 16] (a node is infected if and only if at least r of its neighbors are infected, also known as *bootstrap percolation*) when $r \geq 2$.

We consider networks formed by the *stochastic hierarchical blockmodel* [32, 33] which is a special case of the stochastic blockmodel [15, 20, 36] equipped with a hierarchical structure. Vertices are partitioned into m blocks. The blocks are arranged in a hierarchical structure which represents blocks merging to form larger and larger blocks (communities). The probability of an edge's presence between two vertices is based solely on smallest block to which both the vertices belong. This model captures the intuitive hierarchical structure which is also observed in many real-world networks [17, 12]. The stochastic hierarchical blockmodel is rather general and captures other well-studied models (e.g. Erdős-Rényi random graphs, and the planted community model) as special cases.

Result 1. We first prove that, for the influence maximization problem on the stochastic hierarchical blockmodel with r-complex contagion, under certain mild technical assumptions, the optimal seeding strategy is to put all the seeds in a single community, if, for each

[1] As is common in the literature, we use these terms interchangeably.

vertex-pair (u, v), the probability that the edge (u, v) is included satisfies $p_{uv} = \omega(n^{-(1+1/r)})$. Notice that the assumption $p_{uv} = \omega(n^{-(1+1/r)})$ captures many real life social networks. In fact, it is well-known that an Erdős-Rényi graph $\mathcal{G}(n, p)$ with $p = o(1/n)$ is globally disconnected: with probability $1 - o(1)$, the graph consists of a union of tiny connected components, each of which has size $O(\log n)$.

The technical heart of this result is a novel coupling argument in Proposition 16. We simultaneously couple four cascade processes to compare two probabilities: 1) the probability of infection spreading throughout an Erdős-Rényi graph after the $(k+1)$-st seed, conditioned on not already being entirely infected after k seeds; 2) the probability of infection spreading throughout the same graph after the $(k + 2)$-nd seed, conditioned on not already being entirely infected after $k+1$ seeds. This shows that the marginal rate of infection always goes up, revealing the "supermodular" nature of the r-complex contagion. The supermodular property revealed by Proposition 16 is a property for cascade behavior on Erdős-Rényi random graphs in general, so it is also interesting on its own.

Our result is in sharp contrast to Balkanski et al.'s observation. Balkanski et al. [3] studies the stochastic blockmodel with a well-studied submodular cascade model, *the independent cascade model*, and remarks that "when an influential node from a certain community is selected to initiate a cascade, the marginal contribution of adding another node from that same community is small, since the nodes in that community were likely already influenced."

Algorithmic aspects

For influence maximization in submodular cascades, a greedy algorithm efficiently finds a seeding set with influence at least a $(1 - 1/e)$ fraction of the optimal [22], and much of the work following Kempe et al. [22], which proposed the greedy algorithm, has attempted to make greedy approaches efficient and scalable [10, 11, 26, 13, 35, 34].

Greedy approaches, unfortunately, can perform poorly in the nonsubmodular setting [1]. Moreover, in contrast to the submodular case which has efficient constant approximation algorithms, for general nonsubmodular cascades, it is NP-hard even to approximate influence maximization to within an $\Omega(n^{1-\epsilon})$ factor of the optimal [23]. This inapproximability result has been extended to several much more restrictive nonsubmodular models [9, 25, 32, 33]. Intuitively, nonsubmodular influence maximization is hard because the potential synergy of multiple seeds makes it necessary to consider groups of seeds rather than just individual seeds. In contrast, with submodular influence maximization, not much is lost by considering seeds one at a time in a myopic way.

Can the $\Omega(n^{1-\epsilon})$ inapproximability results of Kempe et al. [23] be circumvented if we further assume the stochastic hierarchical blockmodel? On the one hand, the stochastic hierarchical structure seems optimized for a dynamic programming approach: perform dynamic programming from the bottom to the root in the tree-like community structure. On the other hand, Schoenebeck and Tao [32, 33] show that the $\Omega(n^{1-\epsilon})$ inapproximability results extend to the setting where the networks are stochastic hierarchical blockmodels.

Result 2. However, Result 1 (when the network is reasonably dense, putting all the seeds in a single community is optimal) can naturally be extended to a dynamic programming algorithm. We show that this algorithm is optimal if the probability p_{uv} that each edge appears does not fall into a narrow regime. Interestingly, a heuristic based on dynamic programming works fairly well in practice [1]. Our second result theoretically justifies the success of this approach, at least in the setting of r-complex contagions.

2 Preliminaries

We study complex contagions on social networks with community structure. This section defines the complex contagion and our model for social networks with community structure.

2.1 r-Complex Contagion

Given a social network modeled as an undirected graph $G = (V, E)$, in a cascade, a subset of nodes $S \subseteq V$ is chosen as the seed set; these seeds, being infected, then spread their influence across the graph according to some specified model.

In this paper, we consider a well-known cascade model named *r-complex contagion*, also known as *bootstrap percolation* and the *fixed threshold model*: a node is infected if and only if at least r of its neighbors are infected. We use $\sigma_{r,G}(S)$ to denote the total number of infected vertices at the end of the cascade, and $\sigma_{r,\mathcal{G}}(S) = \mathbb{E}_{G\sim\mathcal{G}}[\sigma_{r,G}(S)]$ if the graph G is sampled from some distribution $\mathcal{G}$. Notice that the function $\sigma_{r,G}(\cdot)$ is deterministic once the graph G and r are fixed.

Submodularity of a cascade model

Other than the r-complex contagion, most cascade models are stochastic: the total number of infected vertices is not deterministic but rather a *random variable*. $\sigma_G(S)$ usually refers to the *expected* number of infected vertices given the seed set S. A cascade model is *submodular* if, given any graph, subsets of vertices $S \subseteq T \subseteq V$, and any additional vertex $v \in V \setminus T$, we have

$$\sigma_G(S \cup \{v\}) - \sigma_G(S) \geq \sigma_G(T \cup \{v\}) - \sigma_G(T),$$

and it is *nonsubmodular* otherwise. Typical submodular cascade models include *the linear threshold model* and *the independent cascade model* [22], which are studied in an enormous past literature. The r-complex contagion, on the other hand, is a paradigmatic nonsubmodular model.

2.2 Stochastic hierarchical blockmodels

We study the *stochastic hierarchical blockmodel* first introduced in [33]. The stochastic hierarchical blockmodel is a special case of the *stochastic blockmodel* [20]. Intuitively, the stochastic blockmodel is a stochastic graph model generating networks with community structure, and the stochastic hierarchical blockmodel further assumes that the communities form a hierarchical structure. Our definition in this section follows closely to [33].

▶ **Definition 1.** *A* stochastic hierarchical blockmodel *is a distribution $\mathcal{G} = (V, T)$ of unweighted undirected graphs sharing the same vertex set V, where $T = (V_T, E_T, w)$ is a weighted tree called a* hierarchy tree. *The third parameter is the weight function $w : V_T \to [0, 1]$ satisfying $w(t_1) < w(t_2)$ for any $t_1, t_2 \in V_T$ such that t_1 is an ancestor of t_2. Let $L_T \subseteq V_T$ be the set of leaves in T. Each leaf node $t \in L_T$ corresponds to a subset of vertices $V(t) \subseteq V$, where the $V(t)$ sets partition the vertices in V. In general, if $t \notin L_T$, we define $V(t) = \bigcup_{t' \in L_T : t' \text{ is an offspring of } t} V(t')$.*

The graph $G = (V, E)$ is sampled from $\mathcal{G}$ in the following way. The vertex set V is deterministic. For $u, v \in V$, the edge (u, v) appears in G with probability equal to the weight of the least common ancestor of u and v in T. That is $\Pr((u, v) \in E) = \max_{t : u, v \in V(t)} w(t)$.

In the rest of this paper, we use the words *tree node* and *vertex* to refer to the vertices in V_T and V respectively. In Definition 1, the tree node $t \in V_T$ corresponds to community $V(t) \subseteq V$ in the social network. Moreover, if t is not a leaf and $t_1, t_2, \ldots$ are the children of t in V_T, then $V(t_1), V(t_2), \ldots$ partition $V(t)$ into sub-communities. Thus, our assumption that for any $t_1, t_2 \in V_T$ where t_1 is an ancestor of t_2 we have $w(t_1) < w(t_2)$ implies that the relation between two vertices is stronger if they are in a same sub-community in a lower level, which is natural.

To capture the scenario where the advertiser has the information on the high-level community structure but lacks the knowledge of the detailed connections inside the communities, when defining the influence maximization problem as an optimization problem, we would like to include T as a part of input, but not G. Rather than choosing which specific vertices are seeds, the seed-picker decides the number of seeds on each leaf and the graph $G \sim \mathcal{G}(n, T)$ is realized after seeds are chosen. Moreover, we are interested in large social networks with $n \to \infty$, so we would like that a single encoding of T is compatible with varying n. To enable this feature, we consider the following variant of the stochastic hierarchical block model.

▶ **Definition 2.** *A* succinct stochastic hierarchical blockmodel *is a distribution $\mathcal{G}(n, T)$ of unweighted undirected graphs sharing the same vertex set V with $|V| = n$, where n is an integer which is assumed to be extremely large. The hierarchy tree $T = (V_T, E_T, w, v)$ is the same as it is in Definition 1, except for the followings.*

1. *Instead of mapping a tree node t to a weight in $[0, 1]$, the weight function $w : V_T \to \mathcal{F}$ maps each tree node to a function $f \in \mathcal{F} = \{f \mid f : \mathbb{Z}^+ \to [0, 1]\}$ which maps an integer (denoting the number of vertices in the network) to a weight in $[0, 1]$. The weight of t is then defined by $(w(t))(n)$. We assume $\mathcal{F}$ is the space of all functions that can be succinctly encoded.*
2. *The fourth parameter $v : V_T \to (0, 1]$ maps each tree node $t \in V_T$ to a fraction of vertices in $V(t)$. That is: $v(t) = |V(t)|/n$. Naturally, we have $\sum_{t \in L_T} v(t) = 1$ and $\sum_{t' : t' \text{ is a child of } t} v(t') = v(t)$.*

We assume throughout that $\mathcal{G}(n, T)$ has the following properties.

Large communities. For tree node $t \in V_T$, because $v(t)$ does not depend on n, $|V(t)| = v(t)n = \Theta(n)$. In particular, $|V(t)|$ goes to infinity as n does.

Proper separation. $w(t_1) = o(w(t_2))$ for any $t_1, t_2 \in V_T$ such that t_1 is an ancestor of t_2. That is, the connection between sub-community t_2 is asymptotically (with respect to n) denser than its super-community t_1.

Our definitions of w and v are designed so that we can fix a hierarchy tree $T = (V_T, E_T, w, v)$ and naturally define $\mathcal{G}(n, T)$ for any n. As we will see in the next subsection, this allows us to take T as input and then allow $n \to \infty$ when considering InfMax (to be defined soon). This enables us to consider graphs having arbitrarily many vertices.

Finally, we define the *density* of a tree node.

▶ **Definition 3.** *Given a hierarchy tree $T = (V_T, E_T, w, v)$ and a tree node $t \in V_T$, the* density *of the tree node is $\rho(t) = w(t) \cdot (v(t)n)^{1/r}$.*

2.3 The InfMax problem

We study the r-complex contagion on the succinct stochastic hierarchical blockmodel. Roughly speaking, given hierarchy tree T and an integer K, we want to choose K seeds which maximize the expected total number of infected vertices, where the expectation is taken over the graph sampling $G \sim \mathcal{G}(n, T)$ as $n \to \infty$.

▶ **Definition 4.** *The* influence maximization problem INFMAX *is an optimization problem which takes as input an integer r, a hierarchy tree $T = (V_T, E_T, w, v)$ as in Definition 2, and an integer K, and outputs $\boldsymbol{k} \in \mathbb{N}_{\geq 0}^{|L_T|}$ – an allocation of K seeds into the leaves L_T with $\sum_{t \in L_T} k_t = K$ that maximizes*

$$\Sigma_{r,T}(\boldsymbol{k}) := \lim_{n \to \infty} \frac{\mathbb{E}_{G \sim \mathcal{G}(n,T)} \left[\sigma_{r,G}(S_{\boldsymbol{k}})\right]}{n},$$ [2]

the expected fraction of infected vertices in $\mathcal{G}(n,T)$ with the seeding strategy defined by $\boldsymbol{k}$, where $S_{\boldsymbol{k}}$ denotes the seed set in G generated according to $\boldsymbol{k}$.

Before we move on, the following remark is very important throughout the paper.

▶ Remark 5. In Definition 4, n is not part of the inputs to the INFMAX instance. Instead, the tree T is given as an input to the instance, and we take $n \to \infty$ to compute $\Sigma_{r,T}(\boldsymbol{k})$ *after* the seed allocation is determined. Therefore, asymptotically, all the input parameters to the instance, including K, r and the encoding size of T, are *constants* with respect to n. Thus, there are two different asymptotic scopes in this paper: *the asymptotic scope with respect to the input size* and *the asymptotic scope with respect to n*. Naturally, when we are analyzing the running time of an INFMAX algorithm, we should use the asymptotic scope with respect to the input size, not of n. On the other hand, when we are analyzing the number of infected vertices after the cascade, we should use the asymptotic scope with respect to n.

In this paper, we use $O_I(\cdot), \Omega_I(\cdot), \Theta_I(\cdot), o_I(\cdot), \omega_I(\cdot)$ to refer to the asymptotic scope with respect to the input size, and we use $O(\cdot), \Omega(\cdot), \Theta(\cdot), o(\cdot), \omega(\cdot)$ to refer to the asymptotic scope with respect to n. For example, with respect to n we always have $r = \Theta(1)$, $K = \Theta(1)$ and $|V_T| = \Theta(1)$.

Lastly, we have assumed that $r \geq 2$, so that the contagion is nonsubmodular. When $r = 1$, the cascade model becomes a special case of the *independent cascade model* [22], which is a submodular cascade model. As mentioned, for submodular INFMAX, a simple greedy algorithm is known to achieve a $(1 - 1/e)$-approximation to the optimal influence [22, 23, 29].

2.4 r-Complex Contagion on Erdős-Rényi graphs

In this section, we consider the r-complex contagion on the Erdős-Rényi random graph $\mathcal{G}(n,p)$. We review some results from [21] which are used in our paper.

▶ **Definition 6.** *The* Erdős-Rényi random graph $\mathcal{G}(n,p)$ *is a distribution of graphs with the same vertex set V with $|V| = n$. For each pair of vertices u, v, the edge (u,v) in included in E independently with probability p.*

The INFMAX problem in Definition 4 on $\mathcal{G}(n,p)$ is trivial, as there is only one possible allocation of the K seeds: allocate all the seeds to the single leaf node of T, which is the root. Therefore, $\sigma_{r,T}(\cdot)$ in Definition 4 depends only on the *number* of seeds $K = |\boldsymbol{k}|$, not on the seed allocation $\boldsymbol{k}$ itself. In this section, we slightly abuse the notation σ such that it is a function mapping an *integer* to $\mathbb{R}_{\geq 0}$ (rather than mapping *an allocation of K seeds* to $\mathbb{R}_{\geq 0}$ as it is in Definition 4). Let $\sigma_{r,\mathcal{G}(n,p)}(k)$ denote the expected number of infected vertices after the cascade given k seeds. Correspondingly, let $\sigma_{r,G}(k)$ denote the actual number of infected vertices after the graph G is sampled from $\mathcal{G}(n,p)$.

▶ **Theorem 7** (A special case of Theorem 3.1 in [21])**.** *Suppose $r \geq 2$, $p = o(n^{-1/r})$ and $p = \omega(n^{-1})$. We have*

1. *if k is a constant, then $\sigma_{r,\mathcal{G}(n,p)}(k) \leq 2k$ with probability $1 - o(1)$;*
2. *if $k = \omega\left((1/np^r)^{1/(r-1)}\right)$, then $\sigma_{r,\mathcal{G}(n,p)}(k) = n - o(n)$ with probability $1 - o(1)$.*

▶ **Theorem 8** (Theorem 5.8 in [21]). *If $r \geq 2$, $p = \omega(n^{-1/r})$ and $k \geq r$, then we have* $\Pr_{G\sim\mathcal{G}(n,p)}[\sigma_{r,G}(k) = n] = 1 - o(1)$.

When $p = \Theta(n^{-1/r})$, the probability that k seeds infect all the n vertices is positive, but bounded away from 1. We use $\mathrm{Po}(\lambda)$ to denote the Poisson distribution with mean λ.

▶ **Theorem 9** (Theorem 5.6 and Remark 5.7 in [21]). *If $r \geq 2$, $p = cn^{-1/r} + o(n^{-1/r})$ for some constant $c > 0$, and $k \geq r$ is a constant, then*

$$\lim_{n\to\infty} \Pr\left(\sigma_{r,\mathcal{G}(n,p)}(k) = n\right) = \zeta(k, c),$$

for some $\zeta(k, c) \in (0, 1)$. Furthermore, there exist numbers $\zeta(k, c, \ell) > 0$ for $\ell \geq k$ such that

$$\lim_{n\to\infty} \Pr\left(\sigma_{r,\mathcal{G}(n,p)}(k) = \ell\right) = \zeta(k, c, \ell)$$

for each $\ell \geq k$, and $\zeta(k, c) + \sum_{\ell=k}^{\infty} \zeta(k, c, \ell) = 1$.

Moreover, the numbers $\zeta(k, c, \ell)$'s and $\zeta(k, c)$ can be expressed as the hitting probabilities of the following inhomogeneous random walk. Let $\xi_\ell \sim \mathrm{Po}\left(\binom{\ell-1}{r-1}c^r\right)$, $\ell \geq 1$ be independent, and let $\tilde{S}_\ell := \sum_{j=1}^{\ell}(\xi_j - 1)$ and $\tilde{T} := \min\{\ell : k + \tilde{S}_\ell = 0\} \in \mathbb{N} \cup \{\infty\}$. Then

$$\zeta(k, c) = \Pr\left(\tilde{T} = \infty\right) = \Pr\left(k + \tilde{S}_\ell \geq 1 \text{ for all } \ell \geq 1\right) \tag{1}$$

and $\zeta(k, c, \ell) = \Pr(\tilde{T} = \ell)$.

We have the following corollary for Theorem 9, saying that when $p = \Theta(n^{-1/r})$, if not all vertices are infected, then the number of infected vertices is constant. As a consequence, if the cascade spreads to more than constantly many vertices, then all vertices will be infected.

▶ **Corollary 10** (Lemma 11.4 in [21]). *If $r \geq 2$, $p = cn^{-1/r} + o(n^{-1/r})$ for some constant $c > 0$, and $k \geq r$, then*

$$\lim_{n\to\infty} \Pr\left(\phi(n) \leq \sigma_{r,\mathcal{G}(n,p)}(k) < n\right) = 0$$

for any function $\phi : \mathbb{Z}^+ \to \mathbb{R}^+$ such that $\lim_{n\to\infty} \phi(n) = \infty$.

3 Our main result

Our main result is the following theorem, which states that the optimal seeding strategy is to put all the seeds in a community with the highest density, when the root has a weight in $\omega(1/n^{1+1/r})$.

▶ **Theorem 11.** *Consider the* InfMax *problem with $r \geq 2$, $T = (V_T, E_T, w, v)$, $K > 0$ and the weight of the root node satisfying $w(root) = \omega(1/n^{1+1/r})$. Let $t^* \in \operatorname{argmax}_{t\in L_T} \rho(t)$ and $\boldsymbol{k}^*$ be the seeding strategy that puts all the K seeds on t^*. Then $\boldsymbol{k}^* \in \operatorname{argmax}_{\boldsymbol{k}} \Sigma_{r,T}(\boldsymbol{k})$.*

Notice that the assumption $w(\mathrm{root}) = \omega(1/n^{1+1/r})$ captures many real life social networks. In fact, it is well-known that an Erdős-Rényi graph $\mathcal{G}(n, p)$ with $p = o(1/n)$ is globally disconnected: with probability $1 - o(1)$, the graph consists of a union of tiny connected components, each of which has size $O(\log n)$.

The remaining part of this section is dedicated to proving Theorem 11. We assume $w(\mathrm{root}) = \omega(1/n^{1+1/r})$ in this section from now on. It is worth noting that, in many parts of this proof, and also in the proof of Theorem 23, we have used the fact that an infection of $o(n)$ vertices contributes 0 to the objective $\Sigma_{r,T}(\boldsymbol{k})$, as we have taken the limit $n \to \infty$ and divided the expected number of infections by n in Definition 4.

▶ **Definition 12.** *Given $T = (V_T, E_T, w, v)$, a tree node $t \in V_T$ is* supercritical *if $w(t) = \omega(1/n^{1/r})$, is* critical *if $w(t) = \Theta(1/n^{1/r})$, and is* subcritical *if $w(t) = o(1/n^{1/r})$.*

From the results in Sect. 2.4, if we allocate $k \geq r$ seeds on a supercritical leaf $t \in L_T$, then with probability $1 - o(1)$ all vertices in $V(t)$ will be infected; if we allocate k seeds on a subcritical leaf $t \in L_T$, at most a negligible number of vertices, $2k = \Theta(1)$, will be infected; if we allocate $k \geq r$ seeds on a critical leaf $t \in L_T$, the number of infected vertices in $V(t)$ follows Theorem 9.

We say a tree node $t \in V_T$ is *activated* in a cascade process if the number of infected vertices in $V(t)$ is $v(t)n - o(n)$, i.e., almost all vertices in $V(t)$ are infected. Given a seeding strategy $\boldsymbol{k}$, let $P_{\boldsymbol{k}}$ be the probability that at least one tree node is activated when $n \to \infty$. Notice that this is equivalent to at least one leaf being activated. The proof of Theorem 11 consists of two parts. We will first show that, $P_{\boldsymbol{k}}$ completely determines $\Sigma_{r,T}(\boldsymbol{k})$ (Lemma 13). Secondly, we show that placing all the seeds on a single leaf with the maximum density will maximize $P_{\boldsymbol{k}}$ (Lemma 14).

▶ **Lemma 13.** *Given any two seeding strategies $\boldsymbol{k}_1, \boldsymbol{k}_2$, if $P_{\boldsymbol{k}_1} \leq P_{\boldsymbol{k}_2}$, then $\Sigma_{r,T}(\boldsymbol{k}_1) \leq \Sigma_{r,T}(\boldsymbol{k}_2)$.*

▶ **Lemma 14.** *Let $\boldsymbol{k}$ be the seeding strategy that allocates all the K seeds on a leaf $t^* \in \operatorname{argmax}_{t \in L_T}(\rho(t))$. Then $\boldsymbol{k}$ maximizes $P_{\boldsymbol{k}}$.*

Lemma 13 and Lemma 14 imply Theorem 11. The proof of Lemma 13 is available in the full version. We prove Lemma 14 in the next section.

3.1 Proof of Lemma 14

We first handle some corner cases. If $K < r$, then the cascade will not even start, and any seeding strategy is considered optimal. If T contains a supercritical leaf, the leaf with the highest density is also supercritical. Putting all the $K \geq r$ seeds in this leaf, by Theorem 8, will activate the leaf with probability $1 - o(1)$. Therefore, this strategy makes $P_{\boldsymbol{k}} = 1$, which is clearly optimal. In the remaining part of this subsection, we shall only consider the case $K \geq r$ and all the leaves are either critical or subcritical. Notice that, by the proper separation assumption, all internal tree nodes of T are subcritical.

We split the cascade process into two stages. In Stage I, we restrict the cascade within the leaf blocks ($V(t)$ where $t \in L_T$), and temporarily assume there are no edges between two different leaf blocks (similar to if $w(t) = 0$ for all $t \notin L_T$). After Stage I, Stage II consists of the remaining cascade process.

Proposition 15 shows that maximizing $P_{\boldsymbol{k}}$ is equivalent to maximizing the probability that a leaf is activated in Stage I. Therefore, we can treat T such that all the leaves, each of which corresponds to a $\mathcal{G}(n, p)$ random graph, are isolated.

▶ **Proposition 15.** *If no leaf is activated after Stage I, then with probability $1 - o(1)$ no vertex will be infected in Stage II, i.e., the cascade will end after Stage I.*

We defer the proof of Proposition 15 to Appendix C. Notice that Proposition 15 is the only part where we have used the proper separation assumption.

Since Theorem 7 suggests that any constant number of seeds will not activate a subcritical leaf, we should only consider putting seeds in critical leaves. In Proposition 16, we show that in a critical leaf t, the probability that the $(i+1)$-th seed will activate t conditioning on the first i seeds failing to do so is increasing as i increases. Intuitively, Proposition 16 reveals a

super-modular nature of the r-complex contagion on a critical leaf, making it beneficial to put all seeds together so that the synergy is maximized, which intuitively implies Lemma 14. The proof of Proposition 16 is the most technical result of this paper, we will present it in Sect. 4.

▶ **Proposition 16** (log-concavity of $\lim_{n\to\infty}\Pr(E_k^n)$)**.** *Consider an Erdős-Rényi random graph $\mathcal{G}(n,p)$ with $p = cn^{-1/r} + o(n^{-1/r})$, and assume an arbitrary order on the n vertices. Let E_k^n be the event that seeding the first k vertices does not make all the n vertices infected. We have $\lim_{n\to\infty}\Pr(E_{k+2}^n \mid E_{k+1}^n) < \lim_{n\to\infty}\Pr(E_{k+1}^n \mid E_k^n)$ for any $k \geq r-1$.*

Equipped with Proposition 16, to show Lemma 14, we show that the seeding strategy that allocates $K_1 > 0$ seeds on a critical leaf t_1 and $K_2 > 0$ seeds on a critical leaf t_2 cannot be optimal. Firstly, it is obvious that both K_1 and K_2 should be at least r, for otherwise those K_1 (K_2) seeds on t_1 (t_2) are simply wasted.

Let E_k^n be the event that the first k seeds on t_1 fail to activate t_1 and F_k^n be the event that the first k seeds on t_2 fail to activate t_2. By Proposition 16, we have $\lim_{n\to\infty}\Pr(E_{K_1+1}^n \mid E_{K_1}^n) < \lim_{n\to\infty}\Pr(E_{K_1}^n \mid E_{K_1-1}^n)$ and $\lim_{n\to\infty}\Pr(F_{K_2+1}^n \mid F_{K_2}^n) < \lim_{n\to\infty}\Pr(F_{K_2}^n \mid F_{K_2-1}^n)$, which implies

$$\begin{aligned}&\lim_{n\to\infty}\frac{\Pr(E_{K_1+1}^n)\Pr(F_{K_2-1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}\cdot\frac{\Pr(E_{K_1-1}^n)\Pr(F_{K_2+1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}\\ =&\lim_{n\to\infty}\frac{\Pr(E_{K_1+1}^n \mid E_{K_1}^n)\Pr(F_{K_2+1}^n \mid F_{K_2}^n)}{\Pr(E_{K_1}^n \mid E_{K_1-1}^n)\Pr(F_{K_2}^n \mid F_{K_2-1}^n)} < 1.\end{aligned}$$

Therefore, we have either $\lim_{n\to\infty}\frac{\Pr(E_{K_1+1}^n)\Pr(F_{K_2-1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}$ or $\lim_{n\to\infty}\frac{\Pr(E_{K_1-1}^n)\Pr(F_{K_2+1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}$ is less than 1. This means either the strategy putting K_1+1 seeds on t_1 and K_2-1 seeds on t_2, or the strategy putting K_1-1 seeds on t_1 and K_2+1 seeds on t_2 makes it more likely that at least one of t_1 and t_2 is activated. Therefore, the strategy putting K_1 and K_2 seeds on t_1 and t_2 respectively cannot be optimal. This implies an optimal strategy should not allocate seeds on more than one leaf.

Finally, a critical leaf t with $v(t)n$ vertices and weight $w(t)$ can be viewed as an Erdős-Rényi random graph $\mathcal{G}(m,p)$ with $m = v(t)n$ and $p = w(t) = \rho(t)\cdot(v(t)n)^{-1/r} = \rho(t)m^{-1/r}$, where $\rho(t) = \Theta(1)$ when t is critical. Taking $c = \rho(t)$ in Theorem 9, we can see that ξ_ℓ has a larger Poisson mean if c is larger, making it more likely that the $\mathcal{G}(m,p)$ is fully infected (to see this more naturally, larger c means larger p if we fix m). Thus, given that we should put all the K seeds in a single leaf, we should put them on a leaf with the highest density. This concludes Lemma 14.

4 Proof for Proposition 16

Since the event E_{k+1}^n implies E_k^n, we have $\Pr(E_{k+1}^n|E_k^n) = \Pr(E_{k+1}^n)/\Pr(E_k^n)$. Therefore, the inequality we are proving is equivalent to $\lim_{n\to\infty}\Pr(E_{k+2}^n)/\Pr(E_{k+1}^n) < \lim_{n\to\infty}\Pr(E_{k+1}^n)/\Pr(E_k^n)$, and it suffices to show that

$$\lim_{n\to\infty}\Pr(E_{k+2}^n)\lim_{n\to\infty}\Pr(E_k^n) < \lim_{n\to\infty}\Pr(E_{k+1}^n)\lim_{n\to\infty}\Pr(E_{k+1}^n). \tag{2}$$

Proposition 16 shows that the failure probability, $\lim_{n\to\infty}\Pr(E_k^n)$, is logarithmically concave.

The remaining part of the proof is split into four parts: In Sect. 4.1, we begin by translating Eqn (2) in the language of inhomogeneous random walks. In Sect. 4.2, we present a coupling of two inhomogeneous random walks to prove Eqn. (2). In Sect. 4.3, we prove the validity of the coupling. in Sect. 4.4, we finally show the coupling implies Eqn. (2).

4.1 Inhomogeneous random walk interpretation

We adopt the inhomogeneous random walk interpretation from Theorem 9, and view the event E_k^n as the following: the random walk starts at $x = k$; in the i-th iteration, x moves to the left by 1 unit, and moves to the right by $\alpha(i) \sim \mathrm{Po}\left(\binom{i-1}{r-1}c^r\right)$ units; Let $\mathcal{E}_k$ be the event that the random walk reaches $x = 0$. By Theorem 9, $\Pr(\mathcal{E}_k) = \lim_{n\to\infty} \Pr(E_k^n)$. Thus, $\lim_{n\to\infty} \Pr(E_{k+2}^n) \lim_{n\to\infty} \Pr(E_k^n) = \Pr(\mathcal{E}_{k+2}) \Pr(\mathcal{E}_k)$. In this proof, we let $\lambda(i) = \binom{i-1}{r-1}c^r$, and in particular, $\lambda(0) = \lambda(1) = \cdots = \lambda(r-1) = 0$. Note that as i increases, the expected movement of the walk increases, and make it harder to reach 0. This observation is important for our proof.

To compute $\Pr(\mathcal{E}_{k+2}) \Pr(\mathcal{E}_k)$, we consider the following process. A random walk in $\mathbb{Z}^2$ starts at $(k+2, k)$. In each iteration i, the random walk moves from (x, y) to $(x - 1 + \alpha(i), y - 1 + \beta(i))$ where $\alpha(i)$ and $\beta(i)$ are sampled from $\mathrm{Po}(\lambda(i))$ independently. If the random walk hits the axis $y = 0$ after a certain iteration $\mathcal{T}$, then it is stuck to the axis, i.e., for any $i > \mathcal{T}$, the update in the i-th iteration is from $(x, 0)$ to $(x - 1 + \alpha(i), 0)$; similarly, after reaching the axis $x = 0$, the random walk is stuck to the axis $x = 0$ and updates to $(0, y - 1 + \beta(i))$. Then, $\Pr(\mathcal{E}_{k+2}) \Pr(\mathcal{E}_k)$ is the probability that the random walk starting from $(k+2, k)$ reaches $(0, 0)$.

To prove (2), we consider two random walks in $\mathbb{Z}^2$ defined above. Let A be the random walk starting from $(k+2, k)$, and let B be the random walk starting from $(k+1, k+1)$. Let H_A and H_B be the event that A and B reaches $(0, 0)$ respectively. To prove (2), it is sufficient to show:

$$\Pr(H_A) < \Pr(H_B).$$

To formalize this idea, we define a coupling between A and B such that: 1) whenever A reaches $(0, 0)$, B also reaches $(0, 0)$, and 2) with a positive probability, B reaches $(0, 0)$ but A never does.

In defining the coupling, we use the idea of splitting and merging of Poisson processes [5]. We reinterpret the random walk by breaking down each *iteration* i into $J(i)$ *steps* such that it is symmetric in the x- and y-directions (with respect to the line $y = x$) and the movement in each step is "small".

If at the beginning of iteration i the process is at (x, y) with $x > 0$ and $y > 0$:

- At step 0 of iteration i, we sample $J(i) \sim \mathrm{Po}(2\lambda(i))$, set $(\alpha(i, 0), \beta(i, 0)) = (-1, -1)$, and update $(x, y) \mapsto (x + \alpha(i, 0), y + \beta(i, 0))$;
- At each step j for $j = 1, \ldots, J(i)$, $(\alpha(i, j), \beta(i, j)) = (1, 0)$ with probability 0.5, and $(\alpha(i, j), \beta(i, j)) = (0, 1)$ otherwise. Update $(x, y) \mapsto (x + \alpha(i, j), y + \beta(i, j))$;[3]

On the other hand, if $x = 0$ (or $y = 0$) at the beginning of iteration:

- At step 0 of iteration i, we sample $J(i) \sim \mathrm{Po}(2\lambda(i))$, set $\big(\alpha(i, 0), \beta(i, 0)\big) = (0, -1)$ (or $(-1, 0)$ if $y = 0$), and update $(x, y) \mapsto \big(x + \alpha(i, 0), y + \beta(i, 0)\big)$;
- At each step j for $j = 1, \ldots, J(i)$, with probability 0.5 $\big(\alpha(i, j), \beta(i, j)\big) = (1, 0)$, (or $\big(\alpha(i, j), \beta(i, j)\big) = (0, 1)$) and $(\alpha(i, j), \beta(i, j)) = (0, 0)$, otherwise. Update $(x, y) \mapsto \big(x + \alpha(i, j), y + \beta(i, j)\big)$;

If at the end of iteration i, $(x, y) = (0, 0)$ we stop the process.

[3] Standard results from Poisson process indicate that, $\sum_{j=1}^{J(i)} \alpha(i, j) \sim \mathrm{Po}(\lambda(i))$, and $\sum_{j=1}^{J(i)} \beta(i, j) \sim \mathrm{Po}(\lambda(i))$ which are two independent Poisson random variables.

Notice that we only switch from one type of iteration to the other if $x = 0$ (or $y = 0$) at the *end* of an iteration i. Here way say the random walk is stuck to the axis $x = 0$ (or the axis $y = 0$). If this happens, it will be stuck to this axis forever. Also, notice that in each step we have at most 1 unit movement. Also, in steps $j = 1, \ldots, J(i)$ the walk can only move further away from both axes $y = 0$ and $x = 0$.

Let $\big(x(i,j), y(i,j)\big)$ be the position of the random walk after iteration i step j, and $\big(x(i), y(i)\big)$ be its position at the end of iteration i. Moreover, let $\alpha(i) = \sum_{j=1}^{J(i)} \alpha(i,j)$ be the net movement in x direction during iteration i excluding the movement in Step 0, and let $\bar{\alpha}(i) = \alpha(i) + \alpha(i,0)$ be the net movement including movement at step 0. Similarly define y-directional movements $\beta(i) = \sum_{j=1}^{J(i)} \beta(i,j)$ and $\bar{\beta}(i)$.

4.2 The coupling

We want to show that the probability of A reaching the origin is less that of B. To this end, we create a coupling between the two walks, which we outline here. Fig. 1 and Fig. 2 illustrate most aspects of this coupling. In the description of the coupling, we will let B move "freely", and define how A is "coupled with" B.

Recall that A starts at $(k+2, k)$ and B starts at $(k+1, k+1)$. At the beginning, we set A's movement to be identical to B's. Before one of them hits the origin, either of the following two events must happen: A and B become symmetric to the line $x = y$ at some step, $\mathcal{E}_{\mathsf{symm}}$, or A reaches the axis $y = 0$ at the end of some iteration, $\mathcal{E}_{\mathsf{skew}}$. This is called Phase I and is further discussed in Sect. 4.2.1.

In the first case $\mathcal{E}_{\mathsf{symm}}$, the positions of A and B are symmetric. We set A's movement to mirror B's movement. Therefore, in this case, A and B will both hit the origin, or neither of them will. This is called Phase II Symm and is further discussed in Sect. 4.2.2.

For the latter case $\mathcal{E}_{\mathsf{skew}}$, A reaches the axis $y = 0$ at iteration $\mathcal{T}_{\mathsf{skew}}$. We call the process is in Phase II Skew and further discussed in Sect. 4.2.3. Because B starts one unit above A and one unit to the left of A, at iteration $\mathcal{T}_{\mathsf{skew}}$, B is at the axis $y = 1$ and one unit to the left of A. Next we couple A's movement in the x-direction to be identical to B's, so that B is always one unit to the left of A. This coupling continues unless B hits the axis $x = 0$. Denote this iteration $\mathcal{T}^*$. At time $\mathcal{T}^*$, A is one unit to the right of the axis $x = 0$. Recall that at iteration $\mathcal{T}_{\mathsf{skew}}$ when $\mathcal{E}_{\mathsf{skew}}$ happens, B is one unit above the axis so that $y = 1$. Therefore, we can couple the movement of A in the x-direction after iteration $\mathcal{T}^*$ with B's movement in the y-direction after iteration $\mathcal{T}_{\mathsf{skew}}$. Because $\lambda(i)$ increases with i, we can couple the walks in such a way as to ensure that A moves toward the origin at a strictly slower rate than B does. Therefore, A only reaches the y-axis $x = 0$ if B reaches the x-axis $y = 0$, and we have shown that A is less likely to reach the origin than B does.

Let $\big(x^A(i,j), y^A(i,j)\big)$, and $\big(x^B(i,j), y^B(i,j)\big)$ be the coordinates for A and B respectively after iteration i step j. Similarly, let $J^A(i)$ and $J^B(i)$ be the number of steps for A and B in iteration i. Let $\alpha^A(i,j)$ and $\alpha^B(i,j)$ be the x-direction movements of both walks in iteration i step j, and $\beta^A(i,j)$ and $\beta^B(i,j)$ be the corresponding y-direction movements.

4.2.1 Phase I

Starting with $\big(x^A(0), y^A(0)\big) = (k+2, k)$ and $\big(x^B(0), y^B(0)\big) = (k+1, k+1)$, A moves in exactly the same way as B, i.e., $J^A(i) = J^B(i)$, $\alpha^A(i,j) = \alpha^B(i,j)$ and $\beta^A(i,j) = \beta^B(i,j)$, until one of the following two events happens.

Event $\mathcal{E}_{\mathsf{symm}}$. The current position of A and B are symmetric with respect to the line $y = x$, i.e., $x^A(i,j) - x^B(i,j) = y^B(i,j) - y^A(i,j)$ and $x^A(i,j) + x^B(i,j) = y^A(i,j) + y^B(i,j)$. Notice that $\mathcal{E}_{\mathsf{symm}}$ may happen in some middle step j of an iteration i. When $\mathcal{E}_{\mathsf{symm}}$ happens, we move on to Phase II Symm.

Event $\mathcal{E}_{\mathsf{skew}}$. A hits the axis $y = 0$ *at the end of an iteration*. Notice that this means A is then stuck to the axis $y = 0$ forever. When $\mathcal{E}_{\mathsf{skew}}$ happens, we move on to Phase II Skew. Note that B is one unit away from the axis $y = 0$, $y^B = 1$. We remark that the in the third part we show, if event $\mathcal{E}_{\mathsf{skew}}$ happens, B has a higher chance to reach $(0, 0)$ than A.

The following three claims will be useful.

▷ Claim 17. A is always below the line $y = x$ before $\mathcal{E}_{\mathsf{symm}}$ happens, so A will never hit the axis $x = 0$ in Phase I.

Proof. To see this, A can only have four types of movements in each step: lower-left $(x, y) \mapsto (x - 1, y - 1)$, up $(x, y) \mapsto (x, y + 1)$, and right $(x, y) \mapsto (x + 1, y)$. It is easy to see that, 1) A will never step across the line $y = x$ in one step, and 2) if A ever reaches the line $y = x$ at (w, w) for some w, then A must be at $(w, w - 1)$ in the previous step. However, when A is at $(w, w - 1)$, B should be at $(w - 1, w)$ according to the relative position of A, B. In this case event $\mathcal{E}_{\mathsf{symm}}$ already happens. ◁

▷ Claim 18. $\mathcal{E}_{\mathsf{symm}}$ and $\mathcal{E}_{\mathsf{skew}}$ cannot happen simultaneously.

Proof. Suppose $\mathcal{E}_{\mathsf{symm}}$ and $\mathcal{E}_{\mathsf{skew}}$ happen at the same time, then it must be that A is at $(1, 0)$ and B is at $(0, 1)$, as the relative position of A and B is unchanged in Phase I, and this must be at the end of a certain *iteration*. In the previous iteration, A must be at $(2, 1)$, since $\mathcal{E}_{\mathsf{skew}}$ did not happen yet and A is below the line $y = x$. However, B is at $(1, 2)$ when A is at $(2, 1)$, implying that case $\mathcal{E}_{\mathsf{symm}}$ has already happened in the previous iteration, which is a contradiction. ◁

▷ Claim 19. B cannot reach the axis $x = 0$ before either $\mathcal{E}_{\mathsf{symm}}$ or $\mathcal{E}_{\mathsf{skew}}$ happen.

Proof. If $\mathcal{E}_{\mathsf{symm}}$ happens before $\mathcal{E}_{\mathsf{skew}}$, B cannot reach the axis $x = 0$ before $\mathcal{E}_{\mathsf{symm}}$ as A is always below the line $y = x$ and B is always on the upper-left diagonal of A. If $\mathcal{E}_{\mathsf{skew}}$ happens before $\mathcal{E}_{\mathsf{symm}}$, B cannot reach the axis $x = 0$ before $\mathcal{E}_{\mathsf{skew}}$, or even by the time $\mathcal{E}_{\mathsf{skew}}$ happens: by the time $\mathcal{E}_{\mathsf{skew}}$ happens, A can only at one of $(2, 0), (3, 0), (4, 0), \ldots$ (A cannot be at $(1, 0)$, for otherwise $\mathcal{E}_{\mathsf{symm}}$ and $\mathcal{E}_{\mathsf{skew}}$ happen simultaneously, which is impossible as shown just now), in which case B will not be at the axis $x = 0$. ◁

4.2.2 Phase II Symm

Let A move in a way that is symmetric to B with respect to the line $y = x$: $J^A(i) = J^B(j)$, $\alpha^A(i,j) = \beta^B(i,j)$ and $\beta^A(i,j) = \alpha^B(i,j)$. Notice that, in Phase II Symm, A may cross the line $y = x$, after which A is above the line $y = x$ while B is below.

4.2.3 Phase II Skew

If event $\mathcal{E}_{\mathsf{skew}}$ happens, we need a more complicated coupling. Suppose Phase II Skew starts after iteration $\mathcal{T}_{\mathsf{skew}}$. Here we use $\mathcal{T}_S^A$ (and $\mathcal{T}_S^B$) to denote the hitting time of A (and B) to a set of states S which is the first iteration of the process into the set S. For example $i = \mathcal{T}_{y=1}^B$ is the hitting time of B such that $y^B(i) = 1$. Here we list six relevant hitting times and their relationship.

$$\mathcal{T}_{\mathsf{skew}} = \mathcal{T}_{y=1}^B = \mathcal{T}_{y=0}^A < \mathcal{T}_{y=0}^B, \text{ and } \mathcal{T}_{\mathsf{skew}} < \mathcal{T}_{x=0}^B = \mathcal{T}_{x=1}^A < \mathcal{T}_{x=0}^A.$$

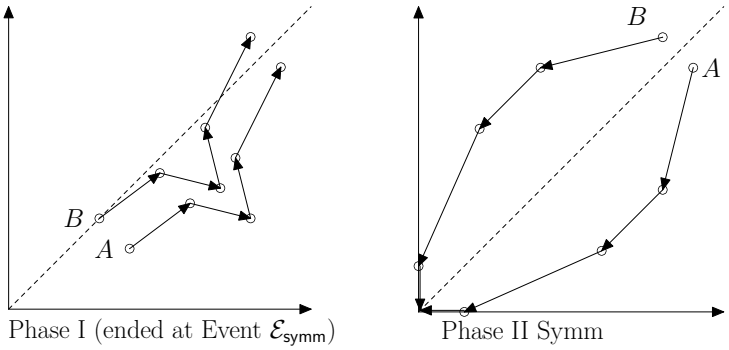


Figure 1 The coupling with Phase I ended at Event $\mathcal{E}_{\mathsf{symm}}$.

Back to the coupling, we first let the x-direction movement of A be the same with that of B. To be specific, in each iteration $\mathcal{T}_{\mathsf{skew}} < i \le \mathcal{T}^B_{x=0}$, set $J^A(i) = J^B(i)$. At step j, we set $\alpha^A(i,j) = \alpha^B(i,j)$ and $\beta^A(i,j) = 0$ ($\beta^A(i,j)$ is always 0 now, as A is stuck to the axis $y = 0$). Till now, the relative position of A and B in x-coordinate is preserved $x^A(i,j) = x^B(i,j) + 1$. Let $\mathcal{E}^*$ be the event that B reaches the axis $x = 0$, and let $\mathcal{E}^*$ happens at the end of iteration $\mathcal{T}^* = \mathcal{T}^B_{x=0}$. We further define $\Delta = \mathcal{T}^* - \mathcal{T}_{\mathsf{skew}}$ to be the additional time before $x^B = 0$ (if both stopping times exist), and $L = \mathcal{T}^B_{y=0} - \mathcal{T}_{\mathsf{skew}}$ to be the additional time before $y^B = 0$ (if both stopping times exist).

At the end of iteration $\mathcal{T}^*$, the positions for A is one unit to the right of the origin. That is $x^A(\mathcal{T}^*) = 1$ while $y^A(\mathcal{T}^*)\big) = 0$. Informally, we want to couple the movement of A from $(1,0)$ at $\mathcal{T}^*$ to the movement of B in the y-direction at $\mathcal{T}_{\mathsf{skew}}$ which is one unit above the axis at $y = 1$. Formally, starting at $(1,0)$, A is a 1-dimensional random walk on the axis $y = 0$, and we couple it to B in the following way.

- For each $t = 1, \dots, L$, we couple A's movement in the x direction at iteration $\mathcal{T}^* + t$ with B's movement Δ steps earlier in the y direction at iteration $\mathcal{T}^* + t - \Delta = \mathcal{T}_{\mathsf{skew}} + t$ such that $\alpha^A(\mathcal{T}^* + t) \sim \mathrm{Po}(\lambda(\mathcal{T}^* + t))$ and $\alpha^A(\mathcal{T}^* + t) \ge \beta^B(\mathcal{T}_{\mathsf{skew}} + t)$. [4]
- We do not couple A to B for future iterations after $\mathcal{T}^* + L$.

A key property of this coupling is that the x-coordinate of A at $\mathcal{T}^* + t$ is always greater or equal to the y-coordinate of B at iteration $\mathcal{T}_{\mathsf{skew}} + t$.

▷ Claim 20. For all $t = 1, \dots, L$, $x^A(\mathcal{T}^* + t) \ge y^B(\mathcal{T}_{\mathsf{skew}} + t)$.

Proof. We use induction. For the base case, we have $1 = x^A(\mathcal{T}^*) = y^B(\mathcal{T}_{\mathsf{skew}})$ from the definitions of $\mathcal{T}_{\mathsf{skew}}$ and $\mathcal{T}^*$. For the inductive case, $\alpha^A(\mathcal{T}^* + t) \ge \beta^B(\mathcal{T}_{\mathsf{skew}} + t)$ due to our coupling. ◁

[4] Here is an example of such a coupling. Consider iteration $i = \mathcal{T}^* + t$ for A, and we want to couple it with B's movement at iteration $\iota = \mathcal{T}_{\mathsf{skew}} + t$. Let $J^B(\iota)$ be the number of steps of B in the iteration ι which is not necessary equal to the number of steps of A after iteration $\mathcal{T}^*$. At step 0, we sample a non-negative integer $d(i) \sim \mathrm{Po}(2(\lambda(i) - \lambda_\iota))$ independent to $J^B(\iota)$, and set the number of steps of A to be $J^A(i) = J^B(\iota) + d(i)$. Then set $\alpha^A(i,0) = -1$ and $\beta(i,0)^A = 0$. At each step $j = 1, \dots, J^B(\iota)$, we set $(\alpha^A(i,j), \beta^A(i,j)) = (\beta^B_{\iota j}, 0)$. At the later steps $j = J^B(\iota) + 1, \dots, J^A(i)$, we set $(\alpha^A(i,j), \beta^A(i,j)) = (1,0)$ with probability 0.5, or $(0,0)$ otherwise.

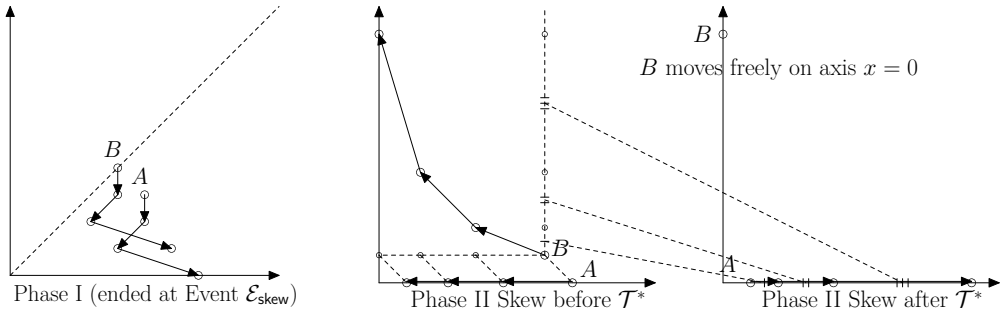


Figure 2 The coupling with Phase I ended at Event $\mathcal{E}_{\mathsf{skew}}$, if $\mathcal{E}^*$ happens.

4.3 Validity of the coupling

The coupling induces the correct marginal random walk process for B, as we have defined the coupling in a way that B is moving "freely" and A is being "coupled" with B. The only non-trivial part is to show that the coupling induces the correct marginal random walk process for A. It is straightforward to check that the marginal probabilities are correct during Phase I, before the event $\mathcal{E}^*$ occurs, or if the event $\mathcal{E}^*$ does not occur. If the process enters Phase II Skew and B reaches the axis $x = 0$, the movement of A in the x direction is coupled with B's movement in y direction $\Delta = \mathcal{T}^* - \mathcal{T}_{\mathsf{skew}}$ iterations ago. We note that B's movements in the x direction and the y direction are independent and A does not contain two iterations that are coupled to a same iteration of B. Therefore, the movements of A in x direction after $\mathcal{T}^*$ are independent to its previous movement, so the marginal distribution is correct. Fig. 3 illustrates the coupling time line.

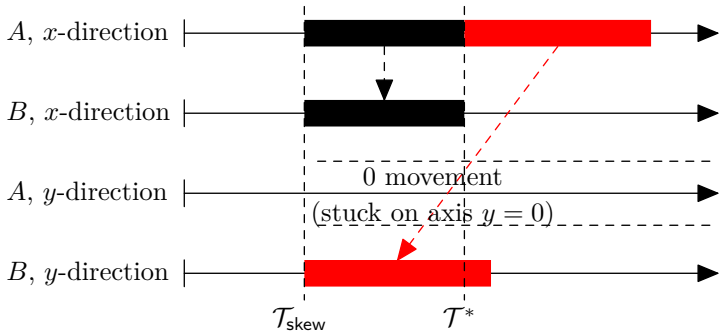


Figure 3 The time line for the coupling after event $\mathcal{E}_{\mathsf{skew}}$ happens.

▶ Remark 21. The coupling of the two random walks A and B in $\mathbb{Z}^2$ in the proof above can be alternatively viewed as a coupling of four independent random walks in $\mathbb{Z}$ (this is why we have said that "we simultaneously couple four cascade processes" in the introduction), as the x-directional and y-directional movements for both A and B correspond to the four terms in inequality (2), which are intrinsically independent.

4.4 Proof of Inequality (2)

It suffices to show that in our coupling $H_A \subseteq H_B$ and $H_B \setminus H_A$ is not empty, because this implies inequality (2): $\Pr(H_A) = \Pr(H_B \cap H_A) < \Pr(H_B \cap H_A) + \Pr(H_B \setminus H_A) = \Pr(H_B)$. We aim to show that:

1. if the coupling never moves to Phase II, neither A nor B reaches $(0,0)$;
2. if the coupling moves to Phase II Symm, A reaches $(0,0)$ if and only if B reaches $(0,0)$;
3. if the coupling moves to Phase II Skew, A reaches $(0,0)$ implies that B also reaches $(0,0)$;
4. with a positive probability, there is an event such that B reaches $(0,0)$ but A does not.

The first, second, and third show $H_A \subseteq H_B$. The last one shows $H_B \setminus H_A$ has a positive probability.

1 is trivial. 2 follows from symmetry.

To see 3, first notice that in Phase II Skew, $\mathcal{E}^*$ must happens if A ever reaches $(0,0)$: because A can move to the left by at most 1 unit in each iteration, A must first reach $(1,0)$, but at this point $x^B = 0$ and event $\mathcal{E}^*$ happens. Now consider the case that B never reaches the origin after event $\mathcal{E}^*$. Then the x movement of A remains coupled to the y-movement of B in such a way that $\bar{\alpha}^A(\mathcal{T}^* + t) \geq \bar{\beta}^B(\mathcal{T}_{\mathsf{skew}} + t)$. Walk A starts at $x^A = 1$, and walk B starts at $y^B = 1$. Therefore, A cannot reach the origin if B does not. In the case walk B meets the origin, the statement is vacuously true.

For 4, to show $\Pr(H_B \setminus H_A) > 0$, we define the following event which consists of four parts. i) For all $i = 1, \ldots, k$, it happens that $\alpha^A(i) = \beta^A(i) = 0$, in which case the event $\mathcal{E}_{\mathsf{skew}}$ happens at $\mathcal{T}_{\mathsf{skew}} = k$ and A reaches $(2,0)$. ii) For $i = k+1$, it happens that $\alpha^A(i) = 0$ and $\beta^B(i) = 1$, in which case A reaches $(1,0)$ and B reaches $(0,1)$, and the process B reaches the axis $x = 0$ at iteration $\mathcal{T}^* = k+1$. iii) In iteration $i = \mathcal{T}^* + 1$, it happens that $\beta^B(i) = 0$, so B reaches $(0,0)$. On the other hand, by the coupling $\alpha^A(\mathcal{T}^* + 1) \geq \beta^B(\mathcal{T}_{\mathsf{skew}} + 1) = 1$, so A does not reach $(0,0)$ at iteration $\mathcal{T}^* + 1 = k+2$. iv) Finally, it happens that $\alpha^A(i) \geq 1$ for all $i > k+2$. It is straightforward the i), ii), and iii) happen with positive probabilities. By direct computations, iv) happens with a positive probability as well.[5] Since the above event consisted of i), ii), iii) and iv) belongs to $H_B \setminus H_A$ and each of the four sub-events happens with a positive probability, 4 is implied.

From 2, 3, and 4, we learn that the probability that B reaches $(0,0)$ is strictly larger than that of A, which implies inequality (2) and concludes the proof.

References

1 Rico Angell and Grant Schoenebeck. Don't be greedy: leveraging community structure to find high quality seed sets for influence maximization. In *International Conference on Web and Internet Economics*, pages 16–29. Springer, 2017.

2 Lars Backstrom, Daniel P. Huttenlocher, Jon M. Kleinberg, and Xiangyang Lan. Group formation in large social networks: membership, growth, and evolution. In *ACM SIGKDD*, 2006.

3 Eric Balkanski, Nicole Immorlica, and Yaron Singer. The Importance of Communities for Learning to Influence. In *Advances in Neural Information Processing Systems*, pages 5862–5871, 2017.

4 Frank M Bass. A new product growth for model consumer durables. *Management science*, 15(5):215–227, 1969.

5 Dimitri P Bertsekas and John N Tsitsiklis. *Introduction to probability*, volume 1. Athena Scientific Belmont, MA, 2002.

6 Jacqueline Johnson Brown and Peter H Reingen. Social ties and word-of-mouth referral behavior. *Journal of Consumer research*, 14(3):350–362, 1987.

[5] The event that $\alpha^A(i) \geq 1$ for all $i > k+2$ happens with probability $\prod_{i>k+2} \Pr(\mathrm{Po}(\lambda(i)) \geq 1) = \prod_{i>k+2}(1 - \exp(-\lambda(i))) \geq \prod_{i \geq r+1}(1 - \exp(-\binom{i-1}{r-1}c^r))$ which is a positive constant depending on r and c.

7 Damon Centola and Michael Macy. Complex contagions and the weakness of long ties. *American journal of Sociology*, 113(3):702–734, 2007.

8 John Chalupa, Paul L Leath, and Gary R Reich. Bootstrap percolation on a Bethe lattice. *Journal of Physics C: Solid State Physics*, 12(1):L31, 1979.

9 Wei Chen, Tian Lin, Zihan Tan, Mingfei Zhao, and Xuren Zhou. Robust influence maximization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 795–804. ACM, 2016.

10 Wei Chen, Yajun Wang, and Siyu Yang. Efficient influence maximization in social networks. In *ACM SIGKDD*, pages 199–208. ACM, 2009.

11 Wei Chen, Yifei Yuan, and Li Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 88–97. IEEE, 2010.

12 Aaron Clauset, Cristopher Moore, and Mark EJ Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98, 2008.

13 Edith Cohen, Daniel Delling, Thomas Pajor, and Renato F Werneck. Sketch-based influence maximization and computation: Scaling up with guarantees. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 629–638. ACM, 2014.

14 James Samuel Coleman, Elihu Katz, and Herbert Menzel. *Medical innovation: A diffusion study*. Bobbs-Merrill Co, 1966.

15 Paul DiMaggio. Structural analysis of organizational fields: A blockmodel approach. *Research in organizational behavior*, 1986.

16 John W Essam. Percolation theory. *Reports on Progress in Physics*, 43(7):833, 1980.

17 Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

18 Jacob Goldenberg, Barak Libai, and Eitan Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 9(3):1–18, 2001.

19 Mark Granovetter. Threshold Models of Collective Behavior. *American Journal of Sociology*, 83(6):1420–1443, 1978. URL: `http://www.journals.uchicago.edu/doi/abs/10.1086/226707`.

20 Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

21 Svante Janson, Tomasz Łuczak, Tatyana Turova, and Thomas Vallier. Bootstrap percolation on the random graph $G_{N,P}$. *The Annals of Applied Probability*, 22(5):1989–2047, 2012.

22 David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

23 David Kempe, Jon Kleinberg, and Éva Tardos. Influential nodes in a diffusion model for social networks. In *International Colloquium on Automata, Languages, and Programming*, pages 1127–1138. Springer, 2005.

24 Jure Leskovec, Lada A. Adamic, and Bernardo A. Huberman. The dynamics of viral marketing. In *EC*, pages 228–237, 2006. `doi:10.1145/1134707.1134732`.

25 Qiang Li, Wei Chen, Xiaoming Sun, and Jialin Zhang. Influence Maximization with ε-Almost Submodular Threshold Functions. In *NIPS*, pages 3804–3814, 2017.

26 Brendan Lucier, Joel Oren, and Yaron Singer. Influence at Scale: Distributed Computation of Complex Contagion in Networks. In *ACM SIGKDD*, pages 735–744. ACM, 2015.

27 John S MacDonald and Leatrice D MacDonald. Chain migration ethnic neighborhood formation and social networks. *The Milbank Memorial Fund Quarterly*, 42(1):82–97, 1964.

28 Vijay Mahajan, Eitan Muller, and Frank M Bass. New product diffusion models in marketing: A review and directions for research. In *Diffusion of technologies and social behavior*, pages 125–177. Springer, 1991.

29 Elchanan Mossel and Sébastien Roch. Submodularity of Influence in Social Networks: From Local to Global. *SIAM J. Comput.*, 39(6):2176–2188, 2010. doi:10.1137/080714452.

30 Elizabeth Levy Paluck, Hana Shepherd, and Peter M Aronow. Changing climates of conflict: A social network experiment in 56 schools. *Proceedings of the National Academy of Sciences*, 113(3):566–571, 2016.

31 Daniel M Romero, Brendan Meeder, and Jon Kleinberg. Differences in the Mechanics of Information Diffusion Across Topics : Idioms , Political Hashtags , and Complex Contagion on Twitter. In *WWW*, pages 695–704. ACM, 2011. URL: http://dl.acm.org/citation.cfm?id=1963503.

32 Grant Schoenebeck and Biaoshuai Tao. Beyond Worst-Case (In)approximability of Nonsubmodular Influence Maximization. In *International Conference on Web and Internet Economics*, pages 368–382. Springer, 2017.

33 Grant Schoenebeck and Biaoshuai Tao. Beyond worst-case (in) approximability of nonsubmodular influence maximization. *ACM Transactions on Computation Theory (TOCT)*, 11(3):12, 2019.

34 Grant Schoenebeck and Biaoshuai Tao. Influence Maximization on Undirected Graphs: Towards Closing the (1-1/e) Gap. In *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, June 24-28, 2019.*, pages 423–453, 2019. doi:10.1145/3328526.3329650.

35 Gordon Tullock. Toward a theory of the rent-seeking society, chapter Efficient rent seeking,(pp. 112), 1980.

36 Harrison C White, Scott A Boorman, and Ronald L Breiger. Social structure from multiple networks. I. Blockmodels of roles and positions. *American journal of sociology*, 81(4):730–780, 1976.

A Optimal seeds in submodular InfMax

We have seen that putting all the K seeds in a single leaf is optimal for r-complex contagion, when the root node has weight $\omega(1/n^{1+1/r})$. To demonstrate the sharp difference between r-complex contagion and a submodular cascade model, we present a submodular InfMax example where the optimal seeding strategy is to put no more than one seed in each leaf. The hierarchy tree T in our example meets all the assumptions we have made in the previous sections, including large communities, proper separation, and $w(\text{root}) = \omega(1/n^{1+1/r})$, where r is now an arbitrarily fixed integer with $r \geq 2$.

We consider a well-known submodular cascade model, *the independent cascade model* [22], where, after seeds are placed, each edge (u, v) in the graph appears with probability p_{uv} and vertices in all the connected components of the resultant graph that contain seeds are infected. In our example, the probability p_{uv} is the same for all edges, and it is $p = 1/n^{1-\frac{1}{4r}}$. The hierarchy tree T contains only two levels: a root and K leaves. The root has weight $1/n^{1+\frac{1}{2r}}$, and each leaf has weight 1. After $G \sim \mathcal{G}(n, T)$ is sampled and each edge in G is sampled with probability p, the probability that an edge appears between two vertices from different leaves is $(1/n^{1-\frac{1}{4r}}) \cdot (1/n^{1+\frac{1}{2r}}) = o(1/n^2)$, and the probability that an edge appears between two vertices from a same leaf is $1 \cdot (1/n^{1-\frac{1}{4r}}) = \omega(\log n/n)$. Therefore, with probability $1 - o(1)$, the resultant graph is a union of K connected components, each of which corresponds to a leaf of T. It is then straightforward to see that the optimal seeding strategy is to put a single seed in each leaf.

B A dynamic programming algorithm

In this section, we present an algorithm which finds an optimal seeding strategy when all $w(t)$'s fall into two regimes: $w(t) = \omega(1/n^{1+1/r})$ and $w(t) = o(1/n^2)$. We will assume this for $w(t)$'s throughout this section. Since a parent tree node always has less weight than its children (see Definition 1), we can decompose T into *the upper part* and *the lower part*, where the lower part consists of many subtrees whose roots have weights in $\omega(1/n^{1+1/r})$, and the upper part is a single tree containing only tree nodes with weights in $o(1/n^2)$ and whose leaves are the parents of those roots of the subtrees in the lower part. We call each subtree in the lower part a *maximal dense subtree* defined formally below.

▶ **Definition 22.** *Given a hierarchy tree $T = (V_T, E_T, w, v)$, a subtree rooted at $t \in V_T$ is a* maximal dense subtree *if $w(t) = \omega(1/n^{1+1/r})$, and either t is the root, or $w(t') = O(1/n^{1+1/r})$ where t' is the parent of t.*

Since we have assumed either $w(t) = \omega(1/n^{1+1/r})$ or $w(t) = o(1/n^2)$, $w(t') = O(1/n^{1+1/r})$ in the definition above implies $w(t') = o(1/n^2)$.

The idea of our algorithm is the following: firstly, after the decomposition of T into the upper and lower parts, we will show that the weights of the tree nodes in the upper part, falling into $w(t) = o(1/n^2)$, are negligible so that we can treat the whole tree T as a forest with only those maximal dense subtrees in the lower part (that is, we can remove the entire upper part from T); secondly, Theorem 11 shows that when we have decided the number of seeds to be allocated for each maximal dense subtree, the optimal seeding strategy is to put all the seeds together in a single leaf that has the highest density, where the density of a leaf $t \in L_T$ is defined in Definition 3; finally, the only problem remaining is how to allocate the K seeds among those maximal dense subtrees, and we decide this allocation by a dynamic programming approach.

Now, we are ready to describe our algorithm, presented in Algorithm 1.

The correctness of Algorithm 1 follows immediately from Theorem 23 (below) and Theorem 11. Recall Theorem 23 shows that we can ignore the upper part of T and treat T as the forest consisting of all the maximal dense subtrees of T when considering the InfMax problem. Theorem 11 shows that for each subtree T_i and given the number of seeds, the optimal seeding strategy is to put all the seeds on the leaf with the highest density.

▶ **Theorem 23.** *Given $T = (V_T, E_T, w, v)$, let $\{T_1, \ldots, T_m\}$ be the set of all T's maximal dense subtrees and let T^- be the forest consisting of $T_1, \ldots, T_m$. For any seeding strategy $\boldsymbol{k}$ and any $r \geq 2$, we have $\Sigma_{r,T}(\boldsymbol{k}) = \Sigma_{r,T^-}(\boldsymbol{k})$.*

Proof. Let $V(T_i)$ be the set of vertices corresponding to the subtree T_i. Since the total number of possible edges between those $V(T_i)$'s is upper bounded by n^2 and each edge appears with probability $o(1/n^2)$, the expected number of edges is $o(1)$. By Markov's inequality the probability there exists edges between those $V(T_i)$'s is $o(1)$. Therefore, we have

$$\frac{\mathop{\mathbb{E}}_{G\sim\mathcal{G}(n,T)}[\sigma_{r,G}(\boldsymbol{k})]}{n} = \frac{o(1)O(n) + (1-o(1))\mathop{\mathbb{E}}_{G\sim\mathcal{G}(n,T')}[\sigma_{r,G}(\boldsymbol{k})]}{n}.$$

Taking $n \to \infty$ concludes the proof. ◀

Finally, it is straightforward to see the time complexity of Algorithm 1, in terms of the number of evaluations of $\Sigma_{r,\mathcal{G}(n,T)}(\cdot)$.

▶ **Theorem 24.** *Algorithm 1 requires $O_I(|V_T|K^2)$ computations of $\Sigma_{r,\mathcal{G}(n,T)}(\cdot)$.*

Algorithm 1 The InfMax algorithm.

Input: $r \in \mathbb{Z}$ with $r \geq 2$, $T = (V_T, E_T, w, v)$, and $K \in \mathbb{Z}^+$
Find all maximal dense subtrees $T_1, \ldots, T_m$, and let $r_1, \ldots, r_m$ be their roots (Definition 22).
For each T_i and each $k = 0, 1, \ldots, K$, let $\boldsymbol{s}_i^*(k)$ be the seeding strategy that puts k seeds in the leaf $t \in L_{T_i}$ with the highest density, and let

$$h(T_i, k) = \lim_{n\to\infty} \frac{\mathbb{E}_{G\sim\mathcal{G}(v(r_i)\cdot n, T_i)}[\sigma_{r,G}(\boldsymbol{s}_i^*(k))]}{n}$$

be the expected number of infected vertices in the subgraph defined by T_i, normalized by the total number of vertices in the whole graph.
Let $S[i,k]$ store a seeding strategy that allocates k seeds in the first i subtrees $T_1, \ldots, T_i$, and let $H[i,k]$ be the expected total number of infected vertices corresponding to $S[i,k]$, divided by n.
for $k = 0, 1, \ldots, K$ **do**
  set $S[1,k] = \boldsymbol{s}_i^*(k)$ and $H[1,k] = h(T_1, k)$.
end for
for each $i = 2, \ldots, m$ **do**
  for $k = 0, 1, \ldots, K$ **do**
    $k_i = \operatorname{argmax}_{k_i \in \{0,1,\ldots,k\}} H[i-1, k-k_i] + h(T_i, k_i)$;
    set $S[i,k]$ be the strategy that allocates $k - k_i$ seeds among $T_1, \ldots, T_{i-1}$ according to $S[i-1, k-k_i]$ and puts the remaining k_i seeds in the leaf of T_i with the highest density;
    set $H[i,k] = H[i-1, k-k_i] + h(T_i, k_i)$;
  end for
end for
Output: the seeding strategy $S[m, K]$.

C Proof of Proposition 15

By Theorem 7 and Corollary 10, if no leaf is activated by the local seeds, then there can be at most constantly many infected vertices. Consider an arbitrary vertex v that is not infected, and let t be the leaf such that $v \in V(t)$. Let K_{in} be the number of infected vertices in $V(t)$ after Stage I and K_{out} be the number of infected vertices outside $V(t)$. By our assumption, $K_{in} = O(1)$ and $K_{out} = O(1)$. We compute an upper bound on the probability that v is infected in the next cascade iteration. Let X_v be the number of v's infected neighbors in $V(t)$ and Y_v be the number of v's infected neighbors outside $V(t)$.

Since the probability that v is connected to each of those K_{out} vertices is $o(n^{-1/r})$, we have

$$\Pr(Y_v \geq r - a) \leq \binom{K_{out}}{r-a} \left(o(n^{-1/r})\right)^{r-a} = o\left(n^{-(r-a)/r}\right)$$

for each $a \in \{0, 1, \ldots, r-1\}$.

Ideally, we would also like to claim that

$$\Pr(X_v \geq a) \leq \binom{K_{in}}{a} w(t)^a = O\left(n^{-a/r}\right), \tag{3}$$

so that putting together we have,

$$\Pr(v \text{ is infected}) \leq \sum_{a=0}^{r-1} \Pr(X_v \geq a) \Pr(Y_v \geq r-a) = r \cdot O\left(n^{-a/r}\right) \cdot o\left(n^{-(r-a)/r}\right) = o\left(\frac{1}{n}\right).$$

and conclude that the expected number of infected vertices in the next iteration is $o(1)$, which implies the proposition by the Markov's inequality.

However, conditioning on the cascade in $V(t)$ stopping after K_{in} infections, there is no guarantee that the probability an edge between v and one of the K_{in} infected vertices is still $w(t)$. Moreover, for any two vertices u_1, u_2 that belong to those K_{in} infected vertices, we do not even know if the probability that v connects to u_1 is still independent of the probability that v connects to u_2. Therefore, (3) does not hold in a straightforward way. The remaining part of this proof is dedicated to proving (3).

Consider a different scenario where we have put K_{in} seeds in $V(t)$ (instead of that the cascade in $V(t)$ ends at K_{in} infections), and let $\bar{X}_v$ be the number of edges between v and those K_{in} seeds (where v is not one of those seeds). Then we know each edge appears with probability $w(t)$ independently, and (3) holds for $\bar{X}_v$:

$$\Pr(\bar{X}_v \geq a) \leq \binom{K_{in}}{a} w(t)^a = O\left(n^{-a/r}\right).$$

Finally, (3) follows because $\bar{X}_v$ stochastically dominates X_v (i.e., $\Pr(\bar{X}_v \geq a) \geq \Pr(X_v \geq a)$ for each $a \in \{0, 1, \ldots, r-1\}$), which is easy to see:

$$\Pr\left(X_v \geq a\right) = \Pr\left(\bar{X}_v \geq a \mid \bar{X}_v \leq r-1\right) = \frac{\Pr(a \leq \bar{X}_v \leq r-1)}{\Pr(\bar{X}_v \leq r-1)}$$

$$= \frac{\Pr(\bar{X}_v \geq a) - \Pr(\bar{X}_v \geq r)}{1 - \Pr(\bar{X}_v \geq r)} \leq \Pr\left(\bar{X}_v \geq a\right),$$

where the first equality holds as $\Pr\left(\bar{X}_v \geq a \mid \bar{X}_v \leq r-1\right)$ exactly describes the probability that v has at least a infected neighbors among K_{in} conditioning on v not yet being infected.

Direct Sum Testing: The General Case

Irit Dinur
The Weizmann Institute of Science, Rehovot, Israel
http://www.wisdom.weizmann.ac.il/~dinuri/
irit.dinur@weizmann.ac.il

Konstantin Golubev
D-MATH, ETH Zurich, Switzerland
https://people.math.ethz.ch/~golubevk/
golubevk@ethz.ch

Abstract

A function $f : [n_1] \times \cdots \times [n_d] \to \mathbb{F}_2$ is a direct sum if it is of the form $f(a_1, \ldots, a_d) = f_1(a_1) \oplus \ldots \oplus f_d(a_d)$, for some d functions $f_i : [n_i] \to \mathbb{F}_2$ for all $i = 1, \ldots, d$, and where $n_1, \ldots, n_d \in \mathbb{N}$. We present a 4-query test which distinguishes between direct sums and functions that are far from them. The test relies on the BLR linearity test (Blum, Luby, Rubinfeld, 1993) and on the direct product test constructed by Dinur & Steurer (2014).

We also present a different test, which queries the function $(d+1)$ times, but is easier to analyze.

In multiplicative ± 1 notation, this reads as follows. A d-dimensional tensor with ± 1 entries is called a tensor product if it is a tensor product of d vectors with ± 1 entries, or equivalently, if it is of rank 1. The presented tests can be read as tests for distinguishing between tensor products and tensors that are far from being tensor products.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases property testing, direct sum, tensor product

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.40

Category RANDOM

Funding *Irit Dinur*: ERC-CoG grant number 772839
Konstantin Golubev: ERC grant number 336283 while at the Weizmann Institute and Bar-Ilan University. Currently, the SNF grant number 200020_169106.

1 Introduction

Let us first fix some notations and definitions. By $[n]$ we mean the set $\{0, 1, 2, \ldots, n\}$. For d positive integers $n_1, \ldots, n_d$, we denote $[\overline{n}; d] = [n_1] \times \cdots \times [n_d]$. For two functions $F, G : X \to Y$, we denote by $\mathrm{dist}(F, G)$ the relative Hamming distance between them, namely $\mathrm{dist}(F, G) = \Pr_{x \in X}[F(x) \neq G(x)]$. We say that $F : X \to Y$ is ε-close to have some Property, if there exists a function $G : X \to Y$ such that g has the Property and $\mathrm{dist}(F, G) \leq \varepsilon$.

Given d functions $f_i : [n_i] \to \mathbb{F}_2$, $i = 1, \ldots, d$, where $n_1, \ldots, n_d \in \mathbb{N}$, their direct sum is the function $f : [\overline{n}; d] \to \mathbb{F}_2$ given by $f(a_1, \ldots, a_d) = f_1(a_1) \oplus f_2(a_2) \oplus \ldots \oplus f_d(a_d)$, where $\oplus$ stands for addition is in the field $\mathbb{F}_2$. We denote $f = f_1 \oplus \cdots \oplus f_d$. We study the testability question: given a function $f : [\overline{n}; d] \to \mathbb{F}_2$ test if it is a direct sum, namely if it belongs to the set

$$DirectSum_{[\overline{n};d]} = \{f_1 \oplus \cdots \oplus f_d \mid f_i : [n_i] \to \mathbb{F}_2,\ i = 1, \ldots, d\}.$$

Direct sum is a natural construction that is often used in complexity for hardness amplification [15, 8, 9, 13, 14]. It is related to the direct product construction: a function $f : [\overline{n}; d] \to \mathbb{F}_2^d$ is the direct product of $f_1, \ldots, f_d$ as above if $f(a_1, \ldots, a_d) = (f_1(a_1), \ldots, f_d(a_d))$ for all $(a_1, \ldots, a_d) \in [\overline{n}; d]$. The testability of direct products has received attention


© Irit Dinur and Konstantin Golubev;
licensed under Creative Commons License CC-BY



[7, 5, 4, 10, 6] as abstraction of certain PCP tests. It was not surprising to find [3] that there is a connection between testing direct products to testing direct sum. However, somewhat unsatisfyingly this connection was confined to testing a certain type of *symmetric* direct sum. A symmetric direct sum is a function $f : [n]^d \to \mathbb{F}_2$ that is a direct product with all components equal; namely such that there is a single $g : [n] \to \mathbb{F}_2$ such that

$$f(a_1, \ldots, a_d) = g(a_1) \oplus g(a_2) \oplus \cdots \oplus g(a_d).$$

In [3], a 3-query test was presented for testing if a given f is a symmetric direct sum, and the analysis carried out relying on the direct product test. It was left as an open question to devise and analyze a test for the property of being a (not necessarily symmetric) direct sum.

We design and analyze a four-query test which we call the "square in a cube" test, and show that it is a strong absolute local test for being a direct sum. That is, the number of queries is an absolute constant (namely, 4), and the distance from a function to the subspace of direct sums is bounded by some absolute constant (independent of n and d) times the probability of the failure of the test on this function. We also describe a simpler $(d+1)$-query test, whose easy analysis we defer to Section 3.

In order to define the test, we need to introduce the following notation. Given two strings $a, b \in [\overline{n}; d]$ and a set $S \subseteq [d]$, denote by $a_S b$ the string in $[\overline{n}; d]$ whose i-th coordinate equals a_i if $i \in S$ and b_i otherwise.

Test 1 Square in a Cube test.

Given a query access to a function $f : [\overline{n}; d] \to \mathbb{F}_2$:

1. Choose $a, b \in [\overline{n}; d]$ uniformly at random.
2. Choose two subsets $S, T \subset [d]$ uniformly at random, and let $U = S \triangle T$ be their symmetric difference.
3. Accept iff

$$f(a) \oplus f(a_S b) \oplus f(a_T b) \oplus f(a_U b) = 0.$$

We prove the following theorem for Test 1.

▶ **Theorem 1.1** (Main). *There exists an absolute constant $c > 0$ s.t. for all $d \in \mathbb{N}$ and $n_1, \ldots, n_d \in \mathbb{N}$, given $f : [\overline{n}; d] \to \mathbb{F}_2$,*

$$\mathrm{dist}(f, DirectSum_{[\overline{n};d]}) \leq c \cdot \Pr_{a,b,S,T}[f(a) \oplus f(a_S b) \oplus f(a_T b) \oplus f(a_{S \triangle T} b) \neq 0]$$

where a, b are chosen independently and uniformly from the domain of f, and S, T are random subsets of $[d]$.

Our proof, similarly to [3], relies on a combination of the BLR linearity testing theorem [2] and the direct product test of [6]. The trick is to find the right combination. We first observe that once we fix a, b, the test is confined to a set of at most 2^d points in the domain, and can be viewed as performing a BLR (affinity rather than linearity) test on this piece of the domain. From the BLR theorem, we deduce an affine linear function on this piece. The next step is to combine the different affine linear functions, one from each piece, into one global direct sum, and this is done by reducing to direct product.

Testing if a tensor has rank 1

An equivalent way to formulate our question is as a test for whether a d-dimensional tensor with ± 1 entries has rank 1. Indeed moving to multiplicative notation and writing $h_i = (-1)^{f_i}$ and $h = (-1)^f$, we are asking whether there are $h_1, \ldots, h_d$ such that

$$h = h_1 \otimes \cdots \otimes h_d.$$

Denoting

$$TensorProduct_{[\overline{n};d]} = \{h_1 \otimes \cdots \otimes h_d \mid h_i : [n_i] \to \{-1, 1\}, , i = 1, \ldots, d\}$$

we have

▶ **Corollary 1.2.** *There exists an absolute constant $c > 0$ s.t. for all $d \in \mathbb{N}$ and $n_1, \ldots, n_d \in \mathbb{N}$, for every $h : [\overline{n}; d] \to \{-1, 1\}$,*

$$\mathrm{dist}(h, TensorProduct_{[\overline{n};d]}) \leq c \cdot \Pr_{a,b,S,T}[h(a) \cdot h(a_S b) \cdot h(a_T b) \cdot h(a_{S \triangle T} b) \neq 1].$$

Structure of the Paper

In Sections 2 and 3 we present two different approaches for testing whether a d-dimensional binary tensor is a tensor product. In Section 4 we discuss possible directions for future research. In Appendix A, we give a proof of the proposition which expands the range of parameters in the direct product test of [6]. This is used in the course of the proof in Section 2.

2 Square in a Cube Test

In this section we present the Square in a Cube Test. Then we introduce the required background: the BLR test for a function being Affine in Subsection 2.1, the direct product test of Dinur & Steurer in Subsection 2.2. Finally, in Subsection 2.3 we prove the main result on the test.

We start by introducing some notation.

Given two vectors $a = (a_1, \ldots, a_d)$, $b = (b_1, \ldots, b_d) \in [\overline{n}; d]$, define

- $\Delta(a, b) = \{i : a_i \neq b_i\} \subseteq [d]$;
- the induced subcube $C_{a,b}$ is the binary cube $\mathbb{F}_2^{\Delta(a,b)}$;
- the projection map $\rho_{a,b} : C_{a,b} \to [\overline{n}; d]$ defined for $x \in C_{a,b}$ as

$$\rho_{a,b}(x)_i = \begin{cases} a_i = b_i, & i \notin \Delta(a, b); \\ b_i, & i \in \Delta(a, b) \text{ and } x_i = 1; \\ a_i, & i \in \Delta(a, b) \text{ and } x_i = 0; \end{cases}$$

The following test is the same as Test 1 in Introduction.

Test 2 Square in a Cube test.

Given a query access to a function $f : [\overline{n}; d] \to \mathbb{F}_2$:

1. Choose $a, b \in [\overline{n}; d]$ uniformly at random.
2. Choose $x, y \in C_{a,b}$ uniformly at random.
3. Query f at $\rho_{a,b}(0), \rho_{a,b}(x), \rho_{a,b}(y)$ and $\rho_{a,b}(x \oplus y)$.
4. Accept iff $f(\rho_{a,b}(0)) \oplus f(\rho_{a,b}(x)) \oplus f(\rho_{a,b}(y)) \oplus f(\rho_{a,b}(x \oplus y)) = 0$.

▶ **Theorem 2.1.** *Suppose a function $f : [\overline{n}; d]^d \to \mathbb{F}_2$ passes Test 2 with probability $1 - \varepsilon$ for some $\varepsilon > 0$, then f is $O(\varepsilon)$-close to a tensor product.*

2.1 The BLR affinity test

The Blum-Luby-Rubinfeld linearity test was introduced in [2], where its remarkable properties were proven. Later a simpler proof via Fourier analysis was presented, e.g. see [1]. Below we give a variation of this test for affine functions, see [12, Chapter 1].

▶ **Definition 2.2.** *A function $g : \mathbb{F}_2^d \to \mathbb{F}_2$ is called affine, if there exists a set $S \subseteq [d]$ and a constant $c \in \mathbb{F}_2$ such that for every vector $x \in \mathbb{F}_2^d$*

$$g(x) = c \oplus \bigoplus_{i \in S} x_i.$$

Note that (see [12, Exercise 1.26]) a function g is affine iff for any two vectors $x, y \in \mathbb{F}_2^d$ it satisfies

$$g(0) \oplus g(x) \oplus g(y) \oplus g(x \oplus y) = 0. \tag{1}$$

The BLR test implies that if a function $g : \mathbb{F}_2^d \to \mathbb{F}_2$ satisfies (1) with high probability, then it is close to an affine function.

Test 3 The BLR affinity test.

Given a query access to a function $f : \mathbb{F}_2^d \to \mathbb{F}_2$:
1. Choose $x \sim \mathbb{F}_2^d$ and $y \sim \mathbb{F}_2^d$ independently and uniformly at random.
2. Query g at $0, x, y$ and $x \oplus y$.
3. Accept if $g(0) \oplus g(x) \oplus g(y) \oplus g(x \oplus y) = 0$.

▶ **Theorem 2.3** ([2])**.** *Suppose $g : \mathbb{F}_2^d \to \mathbb{F}_2$ passes the affinity test with probability $1 - \varepsilon$ for some $\varepsilon > 0$. Then g is ε-close to being affine.*

2.2 Direct Product Test

▶ **Definition 2.4.** *For $k, M, N \in \mathbb{N}$, and k functions $g_1, \ldots, g_k : [N] \to [M]$, their direct product is the function $g : [N]^k \to [M]^k$ denoted $g = g_1 \times \cdots \times g_k$ and defined as $g((x_1, \ldots, x_k)) = (g_1(x_1), \ldots, g_k(x_k))$. A function $g : [N]^k \to [M]^k$, is called a direct product if there exist k functions $g_1, \ldots, g_k : [N] \to [M]$ such that $g = g_1 \times \cdots \times g_k$ for all $(x_1, \ldots, x_k) \in [N]^k$.*

Dinur & Steurer [6] presented a 2-query test, Test 4, that, with constant probability, distinguishes between direct products and functions that are far from direct product.

Test 4 Two-query test $\mathcal{T}(t)$.

Given a query access to a function $g : [N]^k \to [M]^k$:
1. Choose $x \sim \mathbb{F}_2^d$ and $y \sim \mathbb{F}_2^d$ independently and uniformly at random.
2. Query g at $0, x, y$ and $x \oplus y$.
3. Accept if $g(0) \oplus g(x) \oplus g(y) \oplus g(x \oplus y) = 0$.

▶ **Theorem 2.5** ([6, Theorem 1.1])**.** *Let k, M, N be positive integers, let $t \leq \alpha k$, where $0 < \alpha < 1$, and let $\varepsilon > 0$. Let $g : [N]^k \to [M]^k$ be given such that*

$$\Pr_{A,x,y} \left(g(x)_A = g(y)_A \right) \geq 1 - \varepsilon,$$

where A, x, y are chosen w.r.t. the test distribution $\mathcal{T}(t)$. Then there exists a direct product function g' such that $\mathbb{E}_x \left[\operatorname{dist}(g(x), g'(x)) \right] = O(\varepsilon k / t)$.

▶ Remark 2.6. The above formulation of Theorem 2.5 is slightly more general than the original statement in [6], as there it is proved for $0 < \alpha < 1/2$. In order to show that the Theorem holds for $0 < \alpha < 1$, we prove the following reduction statement:

> If a function g passes Test $\mathcal{T}(t)$ with probability at least $1 - \varepsilon$ for $t = \alpha k$ with $1/2 \leq \alpha < 1$, then g passes Test $\mathcal{T}(t')$ with probability at least $1 - \varepsilon'$ for $t' = \alpha' k$, where $0 < \alpha' < 1/2$, $\varepsilon' = r\varepsilon/\alpha$ and r is a positive integer.

This reduction shows that Theorem 2.5 is true as it is stated for $t = \alpha k$ for all $0 < \alpha < 1$, as the reduction affects only the constant in the $O(\cdot)$ notation.

For a more detailed explanation, see Appendix A.

2.3 Proof of Theorem 2.1

For a positive integer D, we denote by $\mu_{2/3}(\mathbb{F}_2^D)$ the distribution on $\mathbb{F}_2^D$, where each coordinate, independently, is equal to 0 with probability 1/3 and to 1 with probability 2/3.

We use the following proposition in the course of the proof.

▶ **Proposition 2.7.** *Let $S \subseteq [D]$ be a set and $\chi_S : \mathbb{F}_2^D \to \mathbb{F}_2$ be the corresponding linear function, i.e., $\chi_S(x) = \bigoplus_{i \in S} x_i$. Suppose*

$$\Pr_{x \sim \mu_{2/3}(\mathbb{F}_2^D)} (\chi_S(x) = 0) > \frac{2}{3},$$

then $S = \emptyset$.

Proof. Consider $(-1)^{\chi_S}$. Then

$$\Pr_{x \sim \mu_{2/3}(\mathbb{F}_2^D)} (\chi_S(x) = 0) = \Pr_{x \sim \mu_{2/3}(\mathbb{F}_2^D)} \left((-1)^{\chi_S(x)} = 1 \right).$$

Also the following holds

$$\frac{1}{3} < \left| 2 \Pr_{x \sim \mu_{2/3}(\mathbb{F}_2^D)} \left((-1)^{\chi_S(x)} = 1 \right) - 1 \right| = \left| \mathbb{E}_{x \sim \mu_{2/3}(\mathbb{F}_2^D)} (-1)^{\chi_S(x)} \right| =$$

$$\left| \prod_{i \in [D]} \mathbb{E}_{x_i \sim \mu_{2/3}(\mathbb{F}_2)} (-1)^{x_i} \right| = \left| \left(-\frac{1}{3} \right)^{|S|} \right| = \left(\frac{1}{3} \right)^{|S|},$$

and the statement follows. ◀

Proof of Theorem 2.1. Assume Test 2 rejects a function $f : [\overline{n}; d] \to \mathbb{F}_2$ with probability less than ε, i.e.,

$$\Pr_{\substack{a,b \sim [\overline{n};d] \\ x,y \sim C_{a,b}}} (f_{a,b}(0) \oplus f_{a,b}(x) \oplus f_{a,b}(y) \oplus f_{a,b}(x \oplus y) = 0) > 1 - \varepsilon,$$

where all distributions are uniform, and $f_{a,b}$ is a shorthand for $f \circ \rho_{a,b}$. Then there exists $a \in [\overline{n}; d]$ such that

$$\Pr_{\substack{b \sim [\overline{n};d] \\ x,y \sim C_{a,b}}} (f_{a,b}(0) \oplus f_{a,b}(x) \oplus f_{a,b}(y) \oplus f_{a,b}(x \oplus y) = 0) > 1 - \varepsilon.$$

Note that the operations re-indexing the domain $[\overline{n}; d]$[1], as well as *flipping* a function, i.e., adding the constant one function to it element-wise, preserve the distance between functions. Hence, w.l.o.g. we can assume for convenience that $a = (0, \ldots, 0)$ and that $f(a) = 0$.

We write C_b for $C_{a,b}$ and f_b for $f_{a,b}$. Then for every $b \in [\overline{n}; d]$,

$$\Pr_{x,y \sim C_b} \left(f_b(0) \oplus f_b(x) \oplus f_b(y) \oplus f_b(x \oplus y) = 0\right) = 1 - \varepsilon_b.$$

The BLR theorem (Theorem 2.3) implies that for each $b \in [\overline{n}; d]$ there exists a subset $S(b) \subseteq \Delta(a, b)$, such that

$$\Pr_{x \sim C_b} \left(f_b(x) = \chi_{S(b)}(x)\right) = 1 - \varepsilon_b.$$

▶ Remark 2.8. By the BLR theorem, there should be the 'greater or equal to' sign instead of the equality. We assume equality for convenience.

Let $F : [\overline{n}; d] \to \mathbb{F}_2^d$ be a function defined as follows. For each $b \in [\overline{n}; d]$, the set $S(b) \subseteq \Delta(a, b)$ can be viewed as a subset of $[d]$, since $\Delta(a, b) \subseteq [d]$. Then $F(b)$ is defined as the element of $\mathbb{F}_2^d$ corresponding to the set $S(b)$.

We now show that F passes Test 4 with high probability and hence is close to a direct product.

Let $b \in [\overline{n}; d]$ be chosen uniformly at random, and let $b' \in [\overline{n}; d]$ be chosen with respect to the following distribution $D(b)$. For each $i \in [d]$,

$$b'_i = \begin{cases} b_i, & \text{w.p. } 3/4; \\ \text{chosen uniformly at random from } [n] \setminus \{b_i\}, & \text{w.p. } 1/4. \end{cases}$$

Note that the distribution on pairs (b, b'), where b is chosen uniformly from $[\overline{n}; d]$ and b' w.r.t. $D(b)$, is equivalent to the following: for each $i \in [d]$,

$$\begin{cases} b_i = b'_i \text{ chosen uniformly from } [n], & \text{w.p. } 3/4; \\ b_i \neq b'_i \text{ both chosen uniformly from } [n] & \text{w.p. } 1/4. \end{cases} \tag{2}$$

In particular, it is symmetric in the sense that choosing $b' \sim [\overline{n}; d]$ uniformly at random first, and then $b \sim D(b')$, leads to the same distribution on pairs (b, b') as the one described above.

For such a pair (b, b') define distribution $\mathcal{D}_{b,b'}$ on $[\overline{n}; d]$ as follows. For a vector $x \sim \mathcal{D}_{b,b'}$,

$$x_i = \begin{cases} 0, & \text{if } i \in \Delta(b, b'); \\ 0, \quad \text{w.p. } 1/3; & \\ b_i = b'_i \quad \text{w.p. } 2/3. & \text{if } i \notin \Delta(b, b'). \end{cases}$$

Note that the distribution $\mathcal{D}_{b,b'}$ is supported on a binary cube of dimension $d - |\Delta(b, b')|$ inside $[\overline{n}; d]$. Denote

$$\varepsilon_{b,b'} = \Pr_{x \sim \mathcal{D}_{b,b'}} \left(f(x) \neq \chi_{F(b)(x)}\right).$$

We claim that the following holds

$$\varepsilon_b = \Pr_{x \sim C_b} \left(f(x) \neq \chi_{F(b)(x)}\right) = \mathop{\mathbb{E}}_{b' \sim D(b)} \varepsilon_{b,b'}. \tag{3}$$

[1] By this we mean selecting permutations π_i on $[n_i]$ for $i = 1, \ldots, d$, and setting $f^{\pi_1, \ldots, \pi_d}(x_1, \ldots, x_d) = f(\pi_1(x_1), \ldots, \pi_d(x_d))$

To see (3) note that since b is chosen uniformly, b' is chosen w.r.t. $D(b)$, and $x \sim \mathcal{D}_{b,b'}$, the resulting distribution for x is

$$x_i = \begin{cases} 0, & \text{w.p. } 1/2; \\ b_i & \text{w.p. } 1/2, \end{cases}$$

which is exactly the uniform distribution on C_b.

We now show that

$$\Pr_{\substack{b\sim[\overline{n};d] \\ b'\sim D(b)}} \left(\varepsilon_{b,b'} + \varepsilon_{b',b} > \frac{1}{3} \right) < 6\varepsilon \tag{4}$$

First note that it follows from the definitions that

$$\mathop{\mathbb{E}}_{b\sim[\overline{n};d]} \mathop{\mathbb{E}}_{b'\sim D(b)} \varepsilon_{b,b'} = \mathop{\mathbb{E}}_{b\sim[\overline{n};d]} \varepsilon_b = \varepsilon.$$

And by the symmetry of the distribution on pairs (b, b'),

$$\mathop{\mathbb{E}}_{b\sim[\overline{n};d]} \mathop{\mathbb{E}}_{b'\sim D(b)} \varepsilon_{b',b} = \mathop{\mathbb{E}}_{b'\sim D(b)} \mathop{\mathbb{E}}_{b\sim[\overline{n};d]} \varepsilon_{b',b} = \varepsilon.$$

Combined together, the previous two equations imply that

$$\mathop{\mathbb{E}}_{b\sim[\overline{n};d]} \mathop{\mathbb{E}}_{b'\sim D(b)} (\varepsilon_{b,b'} + \varepsilon_{b',b}) = 2\varepsilon,$$

and by the Markov inequality, Inequality 4 follows. By the definition of $\varepsilon_{b,b'}$,

$$\Pr_{x\sim\mathcal{D}_{b,b'}} \left(\chi_{F(b)}(x) = \chi_{F(b')}(x) \right) > 1 - (\varepsilon_{b,b'} + \varepsilon_{b',b}) .$$

which is equivalent to

$$\Pr_{x\sim\mathcal{D}_{b,b'}} \left(\chi_{F(b)\Delta F(b')}(x) = 1 \right) > 1 - (\varepsilon_{b,b'} + \varepsilon_{b',b}) .$$

Proposition 2.7 implies that if $1 - (\varepsilon_{b,b'} + \varepsilon_{b',b}) > \frac{2}{3}$, then

$$F(b)_{C_b \cap C_{b'}} = F(b')_{C_b \cap C_{b'}} .$$

By Theorem 2.5, the function $F : [\overline{n}; d] \to \mathbb{F}_2^d$ is close to a direct product, i.e., there exist d functions $F_1, \ldots, F_d : [n] \to \mathbb{F}_2$ such that

$$\Pr_{b\sim[\overline{n};d]} (F(b) = (F_1(b_1), \ldots, F_d(b_d))) \geq 1 - O(\varepsilon).$$

Therefore,

$$\Pr_{b\sim[\overline{n};d]} \left(f(b) = \bigoplus_{i=1}^{d} F_i(b_i) \right) \geq 1 - O(\varepsilon). \qquad \blacktriangleleft$$

3 The Shapka Test

In this section we present a different test for whether a tensor is a tensor product. It queries the tensor at $(d+2)$ places at most, but the proof is simpler than for the previous test.

In [11], Kaufman and Lubotzky showed an interesting connection between the theory of high-dimensional expanders and property testing. Namely, they showed that $\mathbb{F}_2$-coboundary expansion of a 2-dimensional complete simplicial complex implies testability of whether a symmetric $\mathbb{F}_2$-matrix is a tensor square of a vector. The following test is inspired by their work and in a way generalizes it. However, since the description below does not employ neither terminology nor machinery of high-dimensional expanders, we refer to [11] for the connection between this theory and property testing.

Given two strings $a, b \in [\overline{n}; d]$, for $i \in [d]$ denote by $a_b^i \in [\overline{n}; d]$ the vector which coincides with a in every coordinate except for the i-th one, where it coincides with b, i.e.,

$$(a_b^i)_j = \begin{cases} a_j, & \text{if } j \neq i; \\ b_i, & \text{if } j = i. \end{cases}$$

For a string $a \in [\overline{n}; d]$, and a number $x \in [n_i]$, we write a_x^i for the string which is equal to a in every coordinate except for the i-th one, where it is equal to x, i.e.,

$$a_x^i = (a_1, \ldots, a_{i-1}, x, a_{i+1}, \ldots, a_d).$$

Test 5 The Shapka Test.

Given a query access to a function $f : [\overline{n}; d] \to \mathbb{F}_2$:

1. Choose $a, b \in [\overline{n}; d]$ uniformly at random.
2. Define the query set $Q_{a,b} \subseteq [\overline{n}; d]$ to consist of a, a_b^j for all $j \in [d]$, and also b if d is even.
3. Query f at the elements of $Q_{a,b}$.
4. Accept iff $\bigoplus_{q \in Q_{a,b}} f(q) = 0$.

▶ Remark 3.1. Shapka is the Russian word for a winter hat (derived from Old French *chape* for a *cap*). The name *the Shapka test* comes from the fact that the set $Q_{a,b}$ consists of the two top layers of the induced binary cube $C_{a,b}$ (and also the bottom layer if d is even).

▶ **Theorem 3.2.** *Suppose a function $f : [\overline{n}; d] \to \mathbb{F}_2$ passes Test 5 with probability $1 - \varepsilon$ for some $\varepsilon > 0$, then f is ε-close to a tensor product.*

Proof. Let δ be the relative Hamming distance from f to the subspace of direct sums, i.e., for every direct sum $g : [\overline{n}; d] \to \mathbb{F}_2$ it holds that

$$Pr_{x \sim [\overline{n}; d]} \left(f(x) \neq g(x) \right) \geq \delta.$$

For a vector $a \in [\overline{n}; d]$, let us define the local view of f from a, that is d functions $f_1^a, \ldots, f_d^a$, where $f_i^d : [n_i] \to \mathbb{F}_2$, $i = 1, \ldots, d$, that are defined as follows. For $1 \leq i \leq d-1$, and $x \in [n_i]$,

$$f_i^a(x) = f(a_x^i).$$

For $i = d$, the definition of $f_d^a : [n_d] \to \mathbb{F}_2$ depends on the parity of d and goes as follows

$$\begin{cases} f_d^a(x) = f(a_x^d), & \text{if } d \text{ is odd}, \\ f_d^a(x) = f(a_x^d) \oplus f(a), & \text{if } d \text{ is even}. \end{cases}$$

Given a collection of d functions, $g_i : [n_i] \to \mathbb{F}_2$, $i = 1, \ldots, d$, recall that their direct sum is the function $g_1 \oplus \cdots \oplus g_d$ such that for a vector $x \in [\overline{n}; d]$ the following holds

$$g_1 \oplus \cdots \oplus g_d = \bigoplus_{i \in [d]} g_i(x_i).$$

The following holds for any $[\overline{n}; d]$,

$$\left(f - f_1^a \oplus \cdots \oplus f_d^a)\right)(b_1, \ldots, b_d) = \bigoplus_{q \in Q_{a,b}} f(q). \tag{5}$$

As $f_1^a \oplus \cdots \oplus f_d^a$ is a direct sum, it is at least δ-far from f, and hence for any $a \in [\overline{n}; d]$,

$$\Pr_{b \sim [\overline{n}; d]} \left(\left(f - f_1^a \oplus \cdots \oplus f_d^a\right)(b) = 1\right) \geq \delta. \tag{6}$$

Assume now that f fails Test 5 with probability ε, i.e.,

$$\varepsilon = \Pr_{a,b \sim [\overline{n}; d]} \left(\bigoplus_{q \in Q_{a,b}} f(q) = 1 \right).$$

Combining this equality with (5) and (6), we get the following

$$\varepsilon = \mathop{\mathbb{E}}_{a \sim [\overline{n}; d]} \Pr_{b \sim [\overline{n}; d]} \left(\left(f - f_a^1 \oplus \cdots \oplus f_a^d\right)(b_1, \ldots, b_d) = 1\right) \geq \left(\mathop{\mathbb{E}}_{a \sim [\overline{n}; d]} \delta \right) = \delta,$$

which completes the proof. ◀

4 Further Directions

Below we present possible directions for future research.

1. Can the original function $f : [\overline{n}; d] \to \mathbb{F}_2$ be reconstructed by a voting scheme using the Shapka Test 5?
2. It is plausible that the Square in the Cube test 2 can be analyzed by the Fourier transform approach similarly to the analysis of the BLR test.
3. Another test in the spirit of the Shapka Test is the following.

Test 6 The Shapka Test.

Given a query access to a function $f : [\overline{n}; d] \to \mathbb{F}_2$:
- **a.** Choose $a, b \in [\overline{n}; d]$ uniformly at random.
- **b.** Choose $x \in C_{a,b}$ uniformly at random.
- **c.** Query f at $\rho_{a,b}(0), \rho_{a,b}(x), \rho_{a,b}(1)$ and $\rho_{a,b}(x \oplus 1)$.
- **d.** Accept iff $f(\rho_{a,b}(0)) \oplus f(\rho_{a,b}(x)) \oplus f(\rho_{a,b}(1)) \oplus f(\rho_{a,b}(x \oplus 1)) = 0$.

We conjecture that this test is also good, i.e., if a function passes the test with high probability then it is close to a tensor product.

References

1 Mihir Bellare, Don Coppersmith, JOHAN Hastad, Marcos Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996.

2 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of computer and system sciences*, 47(3):549–595, 1993.

3 Roee David, Irit Dinur, Elazar Goldenberg, Guy Kindler, and Igor Shinkar. Direct sum testing. *SIAM Journal on Computing*, 46(4):1336–1369, 2017.

4 Irit Dinur and Elazar Goldenberg. Locally testing direct product in the low error range. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 613–622. IEEE, 2008.

5 Irit Dinur and Omer Reingold. Assignment testers: Towards a combinatorial proof of the PCP theorem. *SIAM Journal on Computing*, 36(4):975–1024, 2006.

6 Irit Dinur and David Steurer. Direct product testing. In *2014 IEEE 29th Conference on Computational Complexity (CCC)*, pages 188–196. IEEE, 2014.

7 Oded Goldreich and Shmuel Safra. A combinatorial consistency lemma with application to proving the PCP theorem. *SIAM Journal on Computing*, 29(4):1132–1154, 2000.

8 Russell Impagliazzo, Ragesh Jaiswal, and Valentine Kabanets. Approximate list-decoding of direct product codes and uniform hardness amplification. *SIAM Journal on Computing*, 39(2):564–605, 2009.

9 Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform direct product theorems: simplified, optimized, and derandomized. *SIAM Journal on Computing*, 39(4):1637–1665, 2010.

10 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. New direct-product testers and 2-query PCPs. *SIAM Journal on Computing*, 41(6):1722–1768, 2012.

11 Tali Kaufman and Alexander Lubotzky. High dimensional expanders and property testing. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 501–506. ACM, 2014.

12 Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

13 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

14 Luca Trevisan. List-decoding using the XOR lemma. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 126–135. IEEE, 2003.

15 Andrew C Yao. Theory and application of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 80–91. IEEE, 1982.

A Appendix: Proof of Remark 2.6

In this section we show that Theorem 2.5 holds for a wider range of parameters than in its original formulation in [6]. This was used in the course of the proof of 2.1.

In [6], Dinur and Steurer proved Theorem 2.5 for $0 < \alpha < 1/2$. The following reduction shows that the theorem is true for all $0 < \alpha < 1$ by a reduction from $1/2 \leq \alpha < 1$ to some $0 < \alpha' < 1/2$. Recall that Test 4 makes two queries according to the distribution $\mathcal{T}(t)$, which is the following distribution: (1) Choose a set $A \subset [k]$ of size t uniformly at random. (2) Choose $x, y \in [N]^k$ uniformly at random, conditioned $x_A = y_A$.

▶ **Proposition A.1.** *Let $agr(g, \alpha)$ denote the probability that a function g passes Test 4 with respect to distribution $\mathcal{T}(\alpha k)$. If $agr(g, \alpha) \geq 1 - \varepsilon$ for some $1/2 \leq \alpha \leq 1$, then $agr(g, \alpha') \geq 1 - r\varepsilon$ for $0 < \alpha' \leq 1/2$, where $r = \left\lceil \frac{1}{2(1-\alpha)} \right\rceil$ and $\alpha' = 1 - (1-\alpha)r$.*

In addition, is $agr(g, 1/2) \geq 1 - \varepsilon$, then also $agr(g, \alpha - 1/k) \geq 1 - 2\varepsilon$.

Proof. Fix a function $g : [N]^k \to [M]^k$, and suppose $agr(g, \alpha) \geq 1 - \varepsilon$ for some $1/2 \leq \alpha < 1$, i.e.,

$$\Pr_{A,x,y\sim\mathcal{T}(\alpha k)} (g(x)_A = g(y)_A) \geq 1 - \varepsilon.$$

We will show that $agr(g, \alpha') > 1 - r\varepsilon$ where $r = \left\lceil \frac{1}{2(1-\alpha)} \right\rceil$ and $\alpha' = 1 - (1-\alpha)r$. Note that α' satisfies $0 < \alpha' \leq 1/2$.

Given a pair of random vectors x_0, x_r and a set A distributed according to $\mathcal{T}(\alpha' k)$, we construct a sequence of vectors $x_1, \ldots, x_{r-1}$ such that for all $1 \leq i \leq r$, the pair x_{i-1}, x_i is distributed according to $\mathcal{T}(\alpha k)$.

The complement of A has size $(1-\alpha)rk$. Partition it randomly into r parts of equal size $(1-\alpha)k$, $[k] \setminus A = B_1 \cup \cdots \cup B_r$. Denote $C_i = [k] \setminus B_i$ for all $1 \leq i \leq r$.

For each $1 \leq i \leq r-1$, construct x_i such that it agrees with x_0 on the coordinates in $[k] \setminus \bigcup_{j=1}^{i} B_j$ and with x_r on the rest of the coordinates $\bigcup_{j=i}^{i} B_j$. Then for each $1 \leq i \leq r$, x_i agrees with x_{i-1} on the set C_i of the size αk. Therefore,

$$\Pr\left(g(x_{i-1})_{A_i} = g(x_i)_{A_i}\right) \geq 1 - \varepsilon.$$

Hence,

$$1 - r \cdot \varepsilon \leq \Pr\left(\forall\, 1 \leq i \leq r :\ g(x_{i-1})_{A_i} = g(x_i)_{A_i}\right) \leq \Pr_{A_r,x,y\sim\mathcal{T}(\alpha' k)} \left(g(x_0)_{A_r} = g(x_r)_{A_r}\right).$$

The case of $\alpha' = 1/2$ has to be treated separately. In this case there is a reduction to $\alpha'' = 1/2 - 1/k$ as follows. Given two vectors x_0, x_2 distributed w.r.t. $\mathcal{T}(k/2-1)$ construct an intermediate random vector x_1 which agrees on exactly half of the coordinates with both x_0 and x_2. ◀

Fast Algorithms at Low Temperatures via Markov Chains

Zongchen Chen
School of Computer Science, Georgia Institute of Technology, Atlanta, USA
chenzongchen@gatech.edu

Andreas Galanis
Department of Computer Science, University of Oxford, Oxford, UK
andreas.galanis@cs.ox.ac.uk

Leslie Ann Goldberg
Department of Computer Science, University of Oxford, Oxford, UK
leslie.goldberg@cs.ox.ac.uk

Will Perkins
Department of Mathematics, Statistics, and Computer Science,
University of Illinois at Chicago, Chicago, USA
william.perkins@gmail.com

James Stewart
Department of Computer Science, University of Oxford, Oxford, UK
james.stewart@cs.ox.ac.uk

Eric Vigoda
School of Computer Science, Georgia Institute of Technology, Atlanta, USA
ericvigoda@gmail.com

Abstract

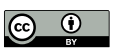
For spin systems, such as the hard-core model on independent sets weighted by fugacity $\lambda > 0$, efficient algorithms for the associated approximate counting/sampling problems typically apply in the high-temperature region, corresponding to low fugacity. Recent work of Jenssen, Keevash and Perkins (2019) yields an FPTAS for approximating the partition function (and an efficient sampling algorithm) on bounded-degree (bipartite) expander graphs for the hard-core model at sufficiently high fugacity, and also the ferromagnetic Potts model at sufficiently low temperatures. Their method is based on using the cluster expansion to obtain a complex zero-free region for the partition function of a polymer model, and then approximating this partition function using the polynomial interpolation method of Barvinok. We present a simple discrete-time Markov chain for abstract polymer models, and present an elementary proof of rapid mixing of this new chain under sufficient decay of the polymer weights. Applying these general polymer results to the hard-core and ferromagnetic Potts models on bounded-degree (bipartite) expander graphs yields fast algorithms with running time $O(n \log n)$ for the Potts model and $O(n^2 \log n)$ for the hard-core model, in contrast to typical running times of $n^{O(\log \Delta)}$ for algorithms based on Barvinok's polynomial interpolation method on graphs of maximum degree Δ. In addition, our approach via our polymer model Markov chain is conceptually simpler as it circumvents the zero-free analysis and the generalization to complex parameters. Finally, we combine our results for the hard-core and ferromagnetic Potts models with standard Markov chain comparison tools to obtain polynomial mixing time for the usual spin system Glauber dynamics restricted to even and odd or "red" dominant portions of the respective state spaces.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains; Theory of computation → Design and analysis of algorithms

Keywords and phrases Markov chains, approximate counting, Potts model, hard-core model, expander graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.41

Category RANDOM



© Zongchen Chen, Andreas Galanis, Leslie Ann Goldberg, Will Perkins, James Stewart, and Eric Vigoda;
licensed under Creative Commons License CC-BY



Related Version A full version of the paper is available at `https://arxiv.org/abs/1901.06653`, and the theorem numbering here matches that of the full version.

Funding *Zongchen Chen*: Research supported in part by NSF grants CCF-1617306 and CCF-1563838.
Andreas Galanis: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828. The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.
Leslie Ann Goldberg: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828. The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.
Will Perkins: Part of this work was done while WP was visiting the Simons Institute for the Theory of Computing.
James Stewart: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828. The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.
Eric Vigoda: Research supported in part by NSF grants CCF-1617306 and CCF-1563838.

1 Introduction

The hard-core model from statistical physics is defined on the set of independent sets of a graph G, where the independent sets are weighted by a fugacity $\lambda > 0$. The associated Gibbs distribution $\mu_{G,\lambda}$ is defined as follows, for an independent set I:

$$\mu_{G,\lambda}(I) = \frac{\lambda^{|I|}}{Z_{G,\lambda}} \tag{1}$$

where $Z_{G,\lambda} = \sum_{I \in \mathcal{I}(G)} \lambda^{|I|}$ is the hard-core partition function (also called the independence polynomial), $\mathcal{I}(G)$ is the set of independent sets of G, and $\lambda > 0$ is the *fugacity*.

In applications, there are two important computational tasks associated to a spin system such as the hard-core model. Given an error parameter $\varepsilon \in (0,1)$, an ε-approximate counting algorithm outputs a number $\hat{Z}$ so that $e^{-\varepsilon} Z_{G,\lambda} \leq \hat{Z} \leq e^{\varepsilon} Z_{G,\lambda}$, and an ε-approximate sampling algorithm outputs a random sample I with distribution $\hat{\mu}$ so that the total variation distance satisfies $\|\mu_\lambda - \hat{\mu}\|_{TV} < \varepsilon$.

While classical statistical physics is most interested in studying the hard-core model on the integer lattice $\mathbb{Z}^d$, the perspective of computer science is to consider wider families of graphs, such as the set of all graphs, all graphs of maximum degree Δ, or all bipartite graphs of maximum degree Δ.

Almost all proven efficient algorithms for approximate counting and sampling from the hard-core model work for low fugacities (*high temperatures* in the language of statistical physics). In the high temperature regime there are at least three distinct algorithmic approaches to approximate counting and sampling: Markov chains, correlation decay, and polynomial interpolation. One striking advantage of the Markov chain approach is that the algorithms are much faster and simpler than the algorithms from the other approaches. In

particular, it is common for a Markov chain sampling algorithm to run in time $O(n \log n)$, e.g., see [8, 10], while typical running times for algorithms based on correlation decay [26, 21] and polynomial interpolation [1] are $n^{O(\log \Delta)}$ where Δ is the maximum degree of the graph.

In general there are no known efficient algorithms at low temperatures (high fugacities), but recently efficient algorithms have been developed for some special classes of graphs including subsets of $\mathbb{Z}^d$ [14], random regular bipartite graphs, and bipartite expander graphs in general [16, 20]. What these bipartite graphs have in common is that for large enough λ, typical independent sets drawn from $\mu_{G,\lambda}$ align closely with one side or the other of the bipartition (the two ground states). This phenomenon is related to the phase transition phenomenon in infinite graphs, and implies the exponentially slow mixing time of local Markov chains [4, 12, 22]. The algorithms introduced in [14] exploit this phenomenon by expressing the partition function $Z_{G,\lambda}$ in terms of deviations from the two ground states, and then using a truncation of a convergent series expansion (the Taylor series or the cluster expansion) to approximate the log partition function. In statistical physics this is called a *perturbative* approach, and while in general it does not work in the largest possible range of parameter space, when it does work it gives a very detailed probabilistic understanding of the model [24, 6, 7].

To apply the perturbative approach at low temperatures, one rewrites the original spin model as a new model in which single spin interactions are replaced by the interaction of connected components representing deviations from a chosen ground state. Such models are called *abstract polymer models*, as detailed below, and have long been used in statistical physics to understand phase transitions. In this paper, we show that once a low temperature spin model has been transformed into an abstract polymer model, Markov chains once again become an effective algorithmic tool. Using this approach we obtain nearly linear and quadratic time sampling algorithms for low temperature models on expander graphs in cases where only $n^{O(\log \Delta)}$-time algorithms were previously known.

1.1 Abstract polymer models

Abstract polymer models, as defined by Gruber and Kunz in 1971 [13], (or "animal models" in Dobruishin's terminology [7]) are an important tool in studying the equilibrium phases of statistical physics models on lattices (e.g. [19, 6] among many others; see [3] for a brief history of their use in statistical physics and combinatorics). Recently they have been used to develop efficient algorithms for sampling and approximating the partition functions of statistical physics models on lattices [14] and expander graphs [16, 20] at low temperatures, the regime in which Markov chains like the Glauber dynamics are known to mix slowly.

We will study the following polymer models. We start with a host graph G and a set $[q] = \{0, \ldots, q-1\}$ of spins. For each vertex v, there is a ground-state spin g_v. A polymer γ consists of a connected set of vertices together with an assignment σ_γ of spins from $\{0, \ldots, q-1\} \setminus g_v$ to each vertex $v \in \gamma$ (we abuse notation and use γ to denote both the polymer and the associated set of vertices). The size of a polymer, $|\gamma|$, is the number of vertices in γ. The set of all polymers is $\mathcal{P}(G)$.

A polymer model on G consists of a set $\mathcal{C}(G) \subseteq \mathcal{P}(G)$ of "allowed" polymers, and a non-negative weight w_γ for each polymer $\gamma \in \mathcal{C}(G)$. We denote this model by $(\mathcal{C}(G), w)$. Two polymers γ and γ' are "compatible" (written $\gamma \sim \gamma'$) if their distance in the host graph is at least 2; otherwise they are incompatible (written $\gamma \nsim \gamma'$). The state space of allowable configurations is $\Omega = \{\Gamma \subseteq \mathcal{C}(G) \mid \forall \gamma, \gamma' \in \Gamma, \gamma \sim \gamma'\}$.

The partition function of the polymer model is $Z(G) = \sum_{\Gamma \in \Omega} \prod_{\gamma \in \Gamma} w_\gamma$, where the empty set of polymers contributes 1 to the partition function. The Gibbs measure μ_G is the probability distribution on Ω given by $\mu_G(\Gamma) = \frac{\prod_{\gamma \in \Gamma} w_\gamma}{Z(G)}$.

The polymer model is in fact a hard-core model on the "incompatibility graph" of $\mathcal{C}(G)$ (two polymers joined by an edge if they are incompatible), with non-uniform fugacities given by the weights w_γ. However, the geometry inherited from the host graph G and the sizes of the polymers adds additional structure to the model.

▶ **Example 1.** One instance of a polymer model is the hard-core model itself: polymers are single vertices of the graph G, labeled with "1" (for occupied) against a ground state "0" (for unoccupied). Each polymer (vertex) v comes with the weight function $w_v = \lambda$. Then the set of allowable polymer configurations is exactly the set of independent sets of G, and so the polymer model partition function is exactly the partition function of the hard-core model on G.

▶ **Example 2.** A second instance of a polymer model is related to the ferromagnetic q-color Potts model on a graph G (see Definition 8 below). Fix a color $g \in [q]$ to be the ground state color, and define polymers to be connected subgraphs of G of size at most M, with vertices labeled by the remaining colors $[q] \setminus \{g\}$. A polymer γ has weight function $w_\gamma = e^{-\beta B(\gamma)}$ where $B(\gamma)$ is the number of bichromatic edges in γ plus the size of the edge boundary of γ in G. A configuration of compatible polymers maps to a Potts configuration σ in which all connected components of non-g-colored vertices have size at most M, and the weight of σ in the Potts model is exactly the product of the weight functions of the polymers. The polymer model partition function $Z(G)$, with an appropriate choice of M, represents the contribution to the Potts model partition function of colorings with dominant color g.

As with the hard-core model, there are two main computational problems associated to a polymer model: approximate sampling from μ_G and approximate counting of $Z(G)$. We will approach them both via Markov chain algorithms. In general we will be interested in families of polymer models defined on classes of graphs. We denote such a family $(\mathcal{C}(\cdot), w, \mathcal{G})$, where for each graph $G \in \mathcal{G}$, $(\mathcal{C}(G), w)$ is a polymer model. We will always use n to denote the number of vertices of a graph G.

We consider two conditions on the weight functions w_γ and give their algorithmic consequences.

▶ **Definition 1.** *A polymer model $(\mathcal{C}(\cdot), w, \mathcal{G})$ satisfies the polymer mixing condition if there exists $\theta \in (0, 1)$ such that*

$$\sum_{\gamma' \not\sim \gamma} |\gamma'| w_{\gamma'} \leq \theta |\gamma| \tag{2}$$

for all $G \in \mathcal{G}$ and all $\gamma \in \mathcal{C}(G)$.

We postpone the formal definition of mixing time to Section 2 and state our first main result here.

▶ **Theorem 2.** *Suppose that a polymer model $(\mathcal{C}(\cdot), w, \mathcal{G})$ satisfies the polymer mixing condition (2). Then for each $G \in \mathcal{G}$ there is a Markov chain making single polymer updates with stationary distribution μ_G and mixing time $T_{\text{mix}}(\varepsilon) = O(n \log(n/\varepsilon))$.*

Theorem 2 on its own does not guarantee an efficient algorithm for sampling from μ_G because the Markov chain only yields an efficient sampling algorithm if we can implement each step efficiently. We will show that under a stronger condition we can do this.

▶ **Definition 3.** *A polymer model $(\mathcal{C}(\cdot), w, \mathcal{G})$ is said to be* computationally feasible *if, for each $G \in \mathcal{G}$ and each $\gamma \in \mathcal{P}(G)$, we can determine, in time polynomial in $|\gamma|$, whether $\gamma \in \mathcal{C}(G)$, and compute w_γ if it is.*

▶ **Definition 4.** *A computationally feasible polymer model $(\mathcal{C}(\cdot), w, \mathcal{G})$ with q spins on a class $\mathcal{G}$ of graphs of maximum degree Δ satisfies the polymer sampling condition with constant $\tau \geq 5 + 3\log((q-1)\Delta)$ if*

$$w_\gamma \leq e^{-\tau|\gamma|} \tag{3}$$

for all $G \in \mathcal{G}$ and all $\gamma \in \mathcal{C}(G)$.

We have the following theorem.

▶ **Theorem 5.** *If a computationally feasible polymer model $(\mathcal{C}(\cdot), w, \mathcal{G})$ satisfies the polymer sampling condition (3) then for all $G \in \mathcal{G}$ there is an ε-approximate sampling algorithm for μ_G with running time $O(n\log(n/\varepsilon))$.*

Finally, we can use the sampling algorithm and simulated annealing to give a fast approximate counting algorithm.

▶ **Theorem 6.** *If a computationally feasible polymer model $(\mathcal{C}(\cdot), w, \mathcal{G})$ satisfies the polymer sampling condition (3) then for all $G \in \mathcal{G}$ there is a randomized ε-approximate counting algorithm for $Z(G)$ with running time $O((n/\varepsilon)^2\log^2(n/\varepsilon))$ and success probability at least $3/4$.*

Fernández, Ferrari, and Garcia [11] introduced a condition very similar to the polymer mixing condition in the setting of polymer models on $\mathbb{Z}^d$. Their objective was to derive probabilistic properties of polymer models directly, without going through the combinatorics and complex analysis inherent in the cluster expansion for the log partition function. They introduced a continuous time stochastic process whose stationary distribution was the infinite volume Gibbs measure of their polymer model and their version of condition (2) implied an exponentially fast rate of convergence of this process. They remarked that such an approach had the potential to be an efficient computational tool.

Here we take an algorithmic point of view, and use the polymer mixing and sampling conditions to show that a simple discrete time Markov chain mixes rapidly and can be used to design efficient sampling and approximation algorithms. Our approach differs from that of [11] in that while they are interested primarily in the probabilistic properties of spin models on $\mathbb{Z}^d$, we are interested in algorithmic problems involving spin models on general families of graphs. Our setting of discrete time processes on finite graphs is also more suitable to studying algorithmic questions. Our work confirms the central point of [11]: that complex analysis and absolute convergence of the cluster expansion is not necessary to derive many important properties of a polymer model.

1.2 Applications

We apply our results for abstract polymer models to two specific examples: the ferromagnetic Potts model and the hard-core model on expander graphs. To state these results we need some definitions.

▶ **Definition 7.** *Let $\alpha > 0$. A graph G is an α-expander graph if for all $S \subset V(G)$ with $|S| \leq |V(G)|/2$, we have $e(S, S^c) \geq \alpha|S|$, where $S^c = V(G) \setminus S$ and $e(S, S^c)$ is the number of edges exiting the set S.*

▶ **Definition 8.** *The q-color ferromagnetic Potts model with parameter $\beta > 0$ is a random assignment of q colors to the vertices of a graph defined by*

$$\mu_{G,\beta}(\sigma) = \frac{e^{-\beta m(G,\sigma)}}{Z_{G,\beta}}$$

where $m(G,\sigma)$ is the number of bichromatic edges of G under the coloring σ and $Z_{G,\beta} = \sum_\sigma e^{-\beta m(G,\sigma)}$ is the Potts model partition function. The parameter β is known as the inverse temperature.

Jenssen, Keevash, and Perkins [16] gave an FPTAS and polynomial-time sampling algorithm for the Potts model on expander graphs, with an algorithm based on the cluster expansion and Barvinok's method of polynomial interpolation. Under essentially the same conditions on the parameters we give a Markov chain based sampling algorithm with near linear running time.

▶ **Theorem 9.** *Suppose $q \geq 2$, $\Delta \geq 3$ are integers and $\alpha > 0$ is a real. Then for $\beta \geq \frac{5+3\log((q-1)\Delta)}{\alpha}$ and any $qe^{-n} \leq \varepsilon < 1$, there is an ε-approximate sampling algorithm for the q-state ferromagnetic Potts model with parameter β on all n-vertex α-expander graphs of maximum degree Δ with running time $O(n\log(n/\varepsilon))$. There is also an ε-approximate counting algorithm with running time $O((n/\varepsilon)^2\log^2(n/\varepsilon))$ and success probability at least 3/4.*

▶ **Definition 10.** *Let $\alpha \in (0,1)$. A bipartite graph $G = (V,E)$ with bipartition $V = V^0 \cup V^1$ is a bipartite α-expander if, for $i \in \{0,1\}$ and all $S \subseteq V^i$ where $|S| \leq |V^i|/2$, we have $N_G(S) \geq (1+\alpha)|S|$ where $N_G(S)$ denotes the set of vertices that are adjacent to some vertex in S.*

Again we give a fast Markov chain based algorithm for sampling from the hard-core model for essentially the same range of parameters for which an FPTAS is given in [16].

▶ **Theorem 11.** *Suppose $\Delta \geq 3$ is an integer and $\alpha \in (0,1)$ is a real. Then for any $\lambda \geq (3\Delta)^{6/\alpha}$ and $4e^{-n} \leq \varepsilon < 1$, there is an ε-approximate sampling algorithm for the hard-core model with parameter λ on all n-vertex bipartite α-expander graphs of maximum degree Δ. There is also an ε-approximate counting algorithm for the hard-core model with success probability at least $1-\varepsilon$. Both algorithms run in time $O((n/\varepsilon)^2\log^3(n/\varepsilon))$.*

The extra factor of n in the running time of the sampling algorithm for the hard-core model as compared to the Potts model is due to the fact that the hard-core model on a bipartite graph does not in general exhibit exact symmetry between the ground states, and so we must approximate the partition functions of the even and odd dominant independent sets to sample.

We can extend these algorithms to obtain fast sampling algorithms in most situations in which a counting problem can be put in the framework of abstract polymer models. For instance, we can use Theorems 5 and 6 to improve the running times of the algorithms given by [17, 20] for sampling and counting proper q-colorings in Δ-regular bipartite graphs (for large Δ). Section 5 of [17] gives a polymer model for proper q-colorings on Δ-regular bipartite graphs. The polymer model is computationally feasible. They prove in Section 5.1 that it satisfies the Kotecký-Preiss condition – in fact, their proof establishes the polymer sampling condition (3). Thus, we get the following corollary of Theorem 5 and 6.

▶ **Corollary 12.** *There is an absolute constant $C > 0$ so that for all even $q \geq 3$, all $\Delta \geq Cq^2 \log^2 q$ and all $\varepsilon > e^{-n/(8q)}$, there is an ε-approximate sampling algorithm to sample a uniformly random proper q-coloring from a random Δ-regular bipartite graph running in time $O(n \log(n/\varepsilon))$. Furthermore, there is a randomized ε-approximation algorithm for the number of proper q-colorings with running time $O((n/\varepsilon)^2 \log^2(n/\varepsilon))$ and success probability at least $3/4$. For odd q, there are ε-approximate counting and sampling algorithms that both run in time $O((n/\varepsilon)^2 \log^3(n/\varepsilon))$.*

As with independent sets, the extra factor of n in the running time for odd q comes from the fact that the ground states (colorings in which one side of the bipartition is assigned $\lceil q/2 \rceil$ colors and the other side $\lfloor q/2 \rfloor$ colors) are exactly symmetric only if q is even.

Finally, we remark that the approximate counting algorithms for these applications based on truncating the cluster expansion can run faster than $n^{O(\log \Delta)}$ if the parameters (expansion, fugacity, inverse temperature) are high enough (see [17, Theorem 8]), but the sampling algorithms derived from this approach will not match the $\tilde{O}(n)$ or $\tilde{O}(n^2)$ sampling algorithms we obtain here.

1.3 Comparison to spin Glauber dynamics

A very natural idea to sample at low temperatures (large β for the Potts model, large λ for the hard-core model) is to use a single-spin update Markov chain like the Glauber dynamics, but to start in one of the ground states of the model chosen at random. For example, pick one of the q-colors with equal probability then start the Potts model Glauber dynamics in the monochromatic configuration with that color. The intuition is that the Glauber dynamics will mix well within the portion of the state space close to the chosen ground state, and the randomness in the choice of ground state will ensure that an accurate sample from the full measure is obtained. Analyzing this algorithm was suggested in [14] and [16].

While we are not yet able to show that this algorithm succeeds, we make partial progress. We show that Glauber dynamics, restricted to remain in a portion of the state space, mixes rapidly (in polynomial time). It is easiest to state our result for the ferromagnetic Potts model.

For a ground state color $g \in [q]$ and an integer M, let $\Omega^g_M(G)$ be the set of q-colorings of the vertices of G so that every connected component of G colored with the palette of colors $[q] \setminus g$ is of size at most M. The set $\Omega^g_M(G)$ consists of colorings that come from the valid polymer configurations from Example 2 above. In [16] it is shown that for an appropriate choice of M, the set $\{\Omega^g_M(G)\}_{g \in [q]}$ forms an "almost partition" of the set of all colorings, in that the weight of both the overlap of the almost partition and the set of colorings uncovered by the almost partition is at most ε under the conditions of Theorem 9. In particular, an ε-approximate sample from the Potts model restricted to $\Omega^g_M(G)$ for $M = O(\log(n/\varepsilon))$ is enough (by symmetry) to obtain a $(q\varepsilon)$-approximate sample from the Potts distribution $\mu_{G,\beta}$ (cf. Lemma 28 of the full version). Using Markov chain comparison, we show in Section 5.3.1 of the full version that an efficient sampler can be obtained using the usual spin Glauber dynamics restricted to remain in $\Omega^g_M(G)$.

▶ **Theorem 13.** *Under the conditions of Theorem 9, and with $M = O(\log(n/\varepsilon))$, the Glauber dynamics restricted to $\Omega^g_M(G)$ has mixing time $T_{\text{mix}}(\varepsilon)$ polynomial in n and $1/\varepsilon$.*

Theorem 13 shows that, despite the exponentially slow mixing of the Glauber dynamics on the full state space, it can still be used by restricting the state space to obtain a polynomial-time approximate sampling algorithm.

In Section 5 of the full version, we give a result (Theorem 23) which is similar to Theorem 13 but applies much more generally – to polymer models which satisfy the polymer mixing condition and other mild conditions. We also obtain a similar theorem (Theorem 29) specifically for the hard-core model.

We leave for future work two important extensions that would complete the picture: 1) showing that *unrestricted* Glauber dynamics starting from a well chosen configuration works, and 2) reducing the running time to $O(n \log n)$ from the large polynomial that we obtain in the theorem.

2 Polymer models and Markov chains

In the full version, we show that the polymer sampling condition (3) implies the well-known Kotecký–Preiss [18] condition $\sum_{\gamma' \nsim \gamma} e^{|\gamma'|} w_{\gamma'} \leq |\gamma|$. The Kotecký–Preiss condition, in turn, implies the polymer mixing condition (2), which is weaker than the Kotecký–Preiss [18] condition.

We next introduce the polymer Markov chain. For each $v \in V(G)$, let $\mathcal{A}(v) = \{\gamma \in \mathcal{C}(G) : v \in \gamma\}$ denote the collection of all polymers containing v and let $a(v) = \sum_{\gamma \in \mathcal{A}(v)} w_\gamma$. By applying (2) to the smallest γ' containing v we have $a(v) \leq \theta < 1$ for all $v \in V(G)$. Define the probability distribution ν_v on $\mathcal{A}(v) \cup \{\emptyset\}$ by $\nu_v(\gamma) = w_\gamma$ for $\gamma \in \mathcal{A}(v)$ and $\nu_v(\emptyset) = 1 - a(v)$.

The polymer dynamics on Ω are defined by the following transition rule from a configuration Γ_t to a configuration Γ_{t+1}:

Polymer Dynamics

1. Choose $v \in V(G)$ uniformly at random. Let $\gamma_v \in \Gamma_t \cap \mathcal{A}(v)$ if $\Gamma_t \cap \mathcal{A}(v) \neq \emptyset$ and let $\gamma_v = \emptyset$ otherwise. Note that γ_v is well defined since Γ_t can have at most one polymer containing v.
2. Mutually exclusively do the following:
   - With probability $\frac{1}{2}$, let $\Gamma_{t+1} = \Gamma_t \setminus \gamma_v$.
   - With probability $\frac{1}{2}$, sample $\boldsymbol{\gamma}$ from ν_v, set $\Gamma_{t+1} = \Gamma_t \cup \boldsymbol{\gamma}$ if this is in Ω and set $\Gamma_{t+1} = \Gamma_t$ otherwise.

In the full version, we verify that the stationary distribution of the polymer dynamics is μ_G by checking detailed balance. Recall that if $\mathcal{M}$ is an ergodic Markov chain with transition matrix P and stationary distribution ν then the mixing time of $\mathcal{M}$ from a state x is given by

$$T_x(\varepsilon) = \min\{t > 0 \mid \text{ for all } t' \geq t, \|P^{t'}(x, \cdot) - \nu(\cdot)\|_{TV} \leq \varepsilon\},$$

where $\|\nu' - \nu\|_{TV}$ denotes the total variation distance between distributions ν and ν'. The mixing time of $\mathcal{M}$ is given by $T_{\text{mix}}(\varepsilon) = \max_x T_x(\varepsilon)$.

Proof of Theorem 2. We will show that under condition 2 the mixing time of the polymer dynamics is $O(n \log(n/\varepsilon))$ by applying the path coupling technique. We define a metric $D(\cdot, \cdot)$ on Ω by setting $D(\Gamma, \Gamma') = 1$ if $\Gamma' = \Gamma \cup \{\gamma\}$ or $\Gamma = \Gamma' \cup \{\gamma\}$ for a polymer γ and extending this as a shortest path metric; i.e., $D(\Gamma, \Gamma') = |\Gamma \triangle \Gamma'|$ for any $\Gamma, \Gamma' \in \Omega$ where $\triangle$ denotes the symmetric difference of two sets. The diameter W of Ω under $D(\cdot, \cdot)$ is no more than $2n$.

Now suppose we couple two chains X_t and Y_t by attempting the same updates in both chains at each step. Suppose that $X_t = Y_t \cup \{\gamma\}$ for some polymer γ. With probability $\frac{|\gamma|}{n} \cdot \frac{1}{2}$ we pick $v \in \gamma$ and remove γ_v which yields $X_{t+1} = Y_{t+1} = X_t$. On the other hand, we may

attempt to add a polymer $\gamma' \nsim \gamma$ so that $Y_t \cup \{\gamma'\} \in \Omega$. That is, $X_{t+1} = X_t = Y_t \cup \{\gamma\}$ and $Y_{t+1} = Y_t \cup \{\gamma'\}$. This occurs with probability $\frac{|\gamma'|}{n} \cdot \frac{1}{2} \cdot w_{\gamma'}$ and in this case $D(X_{t+1}, Y_{t+1}) \leq 2$. Putting these together we can bound

$$\mathbb{E}[D(X_{t+1}, Y_{t+1})] \leq 1 + \frac{1}{2n}\left[-|\gamma| + \sum_{\gamma' \nsim \gamma} |\gamma'| w_{\gamma'}\right].$$

Using (2) we have $\sum_{\gamma' \nsim \gamma} |\gamma'| w_{\gamma'} \leq \theta|\gamma|$, and so $\mathbb{E}[D(X_{t+1}, Y_{t+1})] \leq 1 - |\gamma|\frac{1-\theta}{2n} \leq 1 - \frac{1-\theta}{2n}$. By the path coupling lemma (see [9, Section 6]), the mixing time is at most $\log(W/\varepsilon)2n/(1-\theta) = O(n \log(n/\varepsilon))$. ◀

To prove Theorem 5 we will show that a single update of the polymer dynamics can be computed in constant expected time.

Assume the polymer sampling condition (3) holds with constant $\tau \geq 5 + 3\log((q-1)\Delta)$. We will use the following algorithm. Let $r = \tau - 2 - \log((q-1)\Delta) \geq 3 + 2\log((q-1)\Delta)$ and let $\mathcal{A}_k(v) = \{\gamma \in \mathcal{A}(v) : |\gamma| \leq k\}$.

Single polymer sampler

1. Choose $\mathbf{k}$ according to the following geometric distribution: for k a non-negative integer, $\Pr[\mathbf{k} = k] = (1 - e^{-r})e^{-rk}$. This gives $\Pr[\mathbf{k} \geq k] = e^{-rk}$.
2. Enumerate all polymers in $\mathcal{A}_{\mathbf{k}}(v)$ and compute their weight functions.
3. Mutually exclusively output $\gamma \in \mathcal{A}_{\mathbf{k}}(v)$ with probability $w_\gamma e^{r|\gamma|}$, and with all remaining probability output $\emptyset$. In particular if $\mathbf{k} = 0$, then output $\emptyset$ with probability 1.

We now proceed to prove the following lemma.

▶ **Lemma 16.** *Under the polymer sampling condition* (3) *the output distribution of the* ***single polymer sampler*** *is ν_v and its expected running time is constant.*

Proof. We first show that the probabilities $w_\gamma e^{r|\gamma|}$ sum to less than 1, which shows the last step of the sampling algorithm is well defined. Since $\tau - r = 2 + \log((q-1)\Delta)$,

$$\sum_{\gamma \in A(v)} w_\gamma e^{r|\gamma|} \leq \frac{1}{2}\sum_{k \geq 1} (e\Delta)^{k-1}(q-1)^k e^{-\tau k + rk} = \frac{1}{2e\Delta}\sum_{k \geq 1} e^{-k} < 1,$$

where the first inequality uses the fact that, given a degree Δ graph and a vertex v, there are at most $(e\Delta)^{k-1}/2$ connected size-k subgraphs containing v – a fact proved by Borgs, Chayes, Kahn, and Lovász [5, Lemma 2.1].

We next show that the output of the algorithm has distribution ν_v. Given $\gamma \in \mathcal{A}(v)$, to output γ we must choose $\mathbf{k} \geq |\gamma|$. This happens with probability $e^{-r|\gamma|}$ by the distribution of $\mathbf{k}$. Conditioned on choosing such a $\mathbf{k}$, the probability we output γ is $w_\gamma e^{r|\gamma|}$, and multiplying these probabilities together gives w_γ as desired. Since this is true for all $\gamma \in \mathcal{A}(v)$, the output distribution is exactly ν_v.

Finally we analyze the expected running time. To do this, we appeal to Lemma 3.7 of [23] which gives an algorithm with running time $O(k^5(e\Delta)^{2k})$ for listing all connected subgraphs containing a given vertex v of size at most k (given a graph of degree at most Δ). Consequently, conditioned on the event that $\mathbf{k} = k$, the enumeration step of our algorithm takes time $O(k^5(e\Delta)^{2k})$, and the time taken to determine which polymers are allowed and to compute their weights is $O(k^c(q-1)^k(e\Delta)^{k-1}/2)$ for some $c > 0$, since the polymer model is computationally feasible. In expectation therefore, the running time is

$$O\left(1+\sum_{k\geq 1}\Pr[\mathbf{k}=k]\left(k^5(e\Delta)^{2k}+k^c(e(q-1)\Delta)^k\right)\right)$$
$$=O\left(1+\sum_{k\geq 1}e^{-rk}\,k^c(e(q-1)\Delta)^{2k}\right)=O\left(1+\sum_{k\geq 1}k^c\,e^{-(\tau'+1)k}\right)=O(1)\,,$$

where $\tau'=\tau-5-3\log((q-1)\Delta)\geq 0$. ◀

Proof of Theorem 5. By Theorem 2, there is $T_\varepsilon=O(n\log(n/\varepsilon))$ so that if we start with the empty configuration $\Gamma_0=\emptyset$ and run the polymer dynamics, then Γ_{T_ε} has distribution within $\varepsilon/2$ total variation distance of μ_G. By Lemma 16, in expectation the running time will be $O(n\log(n/\varepsilon))$, but we want an upper bound on the worst case running time as well. To do this, we will simply stop the algorithm and output the empty configuration if the total running time exceeds L for some $L=O(n\log(n/\varepsilon))$ with a sufficiently large leading constant. We next show that the probability that the algorithm terminates in L steps is at most $\varepsilon/2$, which therefore yields that the output distribution has total variation distance at most ε from μ_G.

The randomness in the running time comes from the choice of the geometric random variable $\mathbf{k}$ at each step and the time taken to enumerate polymers in $\mathcal{A}_{\mathbf{k}}(v)$. By the choice of r, the random variable that takes the value $k^5(e\Delta)^{2k}+k^c(e(q-1)\Delta)^k$ with probability $(1-e^{-r})e^{-rk}$ has exponential tails, and so a Chernoff bound shows that the probability that the sum of $\Theta(n\log(n/\varepsilon))$ independent copies of such a random variable is at least twice its expectation is bounded by $e^{-\Theta(n\log(n/\varepsilon))}$ which is at most $\varepsilon/2$ (for large enough choice of constants), finishing the proof. ◀

3 Approximate counting algorithm

In this section we show how to use a sampling oracle to approximately compute the partition function of the polymer model. One standard way is by self-reducibility. In [14] an efficient sampling algorithm for polymer models is derived from an efficient approximate counting algorithm by applying self-reducibility on the level of polymers. While we could apply polymer self-reducibility in the other direction to obtain counting algorithms from our sampling algorithm, here we use the simulated annealing method instead (see [2, 15, 25]) to obtain a faster implementation of counting from sampling.

Suppose that $(\mathcal{C}(G),w)$ is a computationally feasible polymer model. Let ρ be a parameter and define a weight function $w_\gamma(\rho)=w_\gamma e^{-\rho|\gamma|}$ for all $\gamma\in\mathcal{C}(G)$. Then for each $\rho\geq 0$ this defines a computationally feasible polymer model $(\mathcal{C}(G),w(\rho))$ on G, where setting $\rho=0$ recovers the original model $(\mathcal{C}(G),w)$. If the original model $(\mathcal{C}(G),w)$ satisfies the polymer sampling condition (3), then so does $(\mathcal{C}(G),w(\rho))$ for every $\rho\geq 0$ as the weight function $w_\gamma(\rho)$ is monotone decreasing in ρ. Given the graph G, we write the partition function of the polymer model $(\mathcal{C}(G),w(\rho))$ as a function of ρ:

$$Z(\rho)=Z(G;\rho)=\sum_{\Gamma\in\Omega}\prod_{\gamma\in\Gamma}w_\gamma(\rho)=\sum_{\Gamma\in\Omega}\prod_{\gamma\in\Gamma}w_\gamma e^{-\rho|\gamma|}.$$

The associated Gibbs distribution is denoted by $\mu_\rho=\mu_{G;\rho}$. Since $\lim_{\rho\to\infty}w_\gamma(\rho)=0$, we have $\lim_{\rho\to\infty}Z(\rho)=1$ (only the empty configuration Γ contributes to this limit), and so we will use simulated annealing to interpolate between $Z(\infty)=1$ and our goal $Z(0)$, assuming

access to a sampling oracle for $(\mathcal{C}(G), w(\rho))$ for all $\rho \geq 0$. To apply the simulated annealing method, roughly speaking, we find a sequence of parameters $0 = \rho_0 < \rho_1 < \cdots < \rho_\ell < \infty$ called a *cooling schedule* where $\ell \in \mathbb{N}^+$, and then estimate $Z(0)$ using the telescoping product

$$\frac{1}{Z(0)} = \frac{1}{Z(\rho_0)} = \frac{Z(\rho_1)}{Z(\rho_0)}\frac{Z(\rho_2)}{Z(\rho_1)} \cdots \frac{Z(\rho_\ell)}{Z(\rho_{\ell-1})}\frac{1}{Z(\rho_\ell)}.$$

To estimate each term $Z(\rho_{i+1})/Z(\rho_i)$, we define independent random variables $W_i = \prod_{\gamma \in \Gamma_i} \frac{w_\gamma(\rho_{i+1})}{w_\gamma(\rho_i)}$, where $\Gamma_i \sim \mu_{\rho_i}$. It is straightforward to see that $\mathbb{E}[W_i] = Z(\rho_{i+1})/Z(\rho_i)$ (see Lemma 17 of the full version, where we also require the variance). Using the sampling oracle for μ_{ρ_i}, we can sample W_i for all i, and by taking the product we get an estimate for $1/Z(0)$.

The key ingredient of simulated annealing is finding a good cooling schedule. There are nonadaptive schedules [2] that depend only on n, and adaptive schedules [15, 25] that also depend on the structure of $Z(\cdot)$. Usually the latter leads to faster algorithms than the former. In this paper we use a simple nonadaptive schedule: $\rho_i = i/n$ for $i = 1, \ldots, \ell$ where $\ell = O(n \log(n/\varepsilon))$. We show that this cooling schedule already gives us a fast algorithm for the polymer model. The reason behind it is that the weight function $w_\gamma(\rho)$ decays exponentially fast, and so (see Lemma 18 of the full version) the partition function $Z(\rho_\ell)$ is bounded by a constant when $\rho_\ell = O(\log n)$, leading to a short cooling schedule. Our algorithm is as follows.

Polymer approximate counting algorithm

1. Let $\rho_i = i/n$ for $i = 0, 1, \ldots, \ell$ where $\ell = \lceil n \log(4e(q-1)\Delta n/\varepsilon) \rceil$;
2. For $j = 1, \ldots, m$ where $m = \lceil 64\varepsilon^{-2} \rceil$:
   a. For $0 \leq i \leq \ell - 1$:
      (i) Sample $\Gamma_i^{(j)}$ from μ_{ρ_i};
      (ii) Let $W_i^{(j)} = \prod_{\gamma \in \Gamma_i^{(j)}} e^{-|\gamma|/n}$;
   b. Let $W^{(j)} = \prod_{i=0}^{\ell-1} W_i^{(j)}$;
3. Let $\widehat{W} = \frac{1}{m}\sum_{j=1}^{m} W^{(j)}$ and output $\widehat{Z} = 1/\widehat{W}$.

For $0 \leq i \leq \ell - 1$ we define Γ_i to be an independent random sample from μ_{ρ_i} and $W_i = \prod_{\gamma \in \Gamma_i} e^{-|\gamma|/n}$. Finally, we let $W = \prod_{i=0}^{\ell-1} W_i$.

Proof of Theorem 6. In this version, we assume that we have access to an exact sampler $\mathcal{S}_{\text{exact}}$ that samples from μ_ρ for all $\rho \geq 0$ (in the full version we show how to adapt the argument to the situation where we only have an approximate sampler). Using this sampler in the Polymer approximate counting algorithm, we find that, for each j and each i, $\Gamma_i^{(j)}$ is an exact sample from the distribution μ_{ρ_i} and hence $W_i^{(j)}$ is an exact sample of W_i, independently for every j and i. Thus, $W^{(j)}$ is a sample of W independently for every j, and $\widehat{W}$ is the sample mean of $W^{(j)}$'s. We deduce from Lemmas 17 and 18 of the full version that

$$(1 + \varepsilon/2)\mathbb{E}[W] \leq \frac{e^{\varepsilon/2} Z(\rho_\ell)}{Z(0)} \leq \frac{e^\varepsilon}{Z(0)} \text{ and } (1 - \varepsilon/2)\mathbb{E}[W] \geq \frac{e^{-\varepsilon} Z(\rho_\ell)}{Z(0)} \geq \frac{e^{-\varepsilon}}{Z(0)}$$

where we use $1 + \varepsilon/2 \leq e^{\varepsilon/2}$ and $e^{-\varepsilon} \leq 1 - \varepsilon/2$ for all $0 < \varepsilon < 1$. Then

$$\Pr\left(\frac{e^{-\varepsilon}}{Z(0)} \leq \widehat{W} \leq \frac{e^\varepsilon}{Z(0)}\right) \geq \Pr\left(\left|\widehat{W} - \mathbb{E}[W]\right| \leq (\varepsilon/2)\mathbb{E}[W]\right).$$

By Chebyshev's inequality we have

$$\Pr\left(\left|\widehat{W} - \mathbb{E}[W]\right| \geq (\varepsilon/2)\mathbb{E}[W]\right) \leq \frac{4\operatorname{Var}(W)}{\varepsilon^2 m \left(\mathbb{E}[W]\right)^2} \leq \frac{4(e-1)}{\varepsilon^2 m} \leq \frac{1}{8}$$

where the second to last inequality follows from Lemmas 17 and 19 of the full version which enable us to show that

$$\frac{\operatorname{Var}(W)}{\left(\mathbb{E}[W]\right)^2} = \frac{\mathbb{E}[W^2]}{\left(\mathbb{E}[W]\right)^2} - 1 = \frac{Z(0)}{Z(\rho_1)}\frac{Z(\rho_{\ell+1})}{Z(\rho_\ell)} - 1 \leq e - 1.$$

Thus, we deduce that

$$\Pr\left(e^{-\varepsilon}Z(0) \leq \widehat{Z} \leq e^{\varepsilon}Z(0)\right) = \Pr\left(\frac{e^{-\varepsilon}}{Z(0)} \leq \hat{W} \leq \frac{e^{\varepsilon}}{Z(0)}\right) \geq \frac{7}{8}$$

(so the error probability is at most 1/8). Note that the number of samples that we used is ℓm. Finally, we consider the running time of our algorithm. By Theorem 5, the running time of step 2(a)(i) is $O(n\log(8\ell mn)) = O(n\log(n/\varepsilon))$, and for step 2(a)(ii) the running time is $O(n)$. Thus, the running time of the algorithm is upper bounded by $\ell m \cdot O(n\log(n/\varepsilon)) = O((n/\varepsilon)^2\log^2(n/\varepsilon))$. ◀

4 Applications

In this section, we prove Theorem 9 for the Potts model. The proof of Theorem 11 (for the hard-core model) can be found in Section 4.2 of the full version. Throughout this section, we will work under the assumptions/conditions of Theorem 9. That is, we fix a real number $\alpha > 0$, integers $q \geq 3$ and $\Delta \geq 3$ and a real number $\beta \geq \frac{5+3\log((q-1)\Delta)}{\alpha}$. We let $\mathcal{G}$ be the class of α-expander graphs G with maximum degree at most Δ.

Consider the polymer model defined in Example 2 on an n-vertex graph $G \in \mathcal{G}$ with $M = n/2$ and ground state color $g \in [q]$. We will use $\mathcal{C}^g = \mathcal{C}^g(G)$ to denote the polymers and w^g_γ to denote the weight of a polymer $\gamma \in \mathcal{C}^g$; recall that $w^g_\gamma = e^{-\beta B(\gamma)}$, where $B(\gamma)$ counts the number of external edges of γ plus the number of bichromatic internal edges. Let $Z^g(G)$ be the partition function of the polymer model $(\mathcal{C}^g(G), w^g)$.

▶ **Lemma 20.** *Under the conditions of Theorem 9, the polymer model $(\mathcal{C}^g(\cdot), w^g, \mathcal{G})$ satisfies the polymer sampling condition (3) with $\tau = \alpha\beta$.*

Proof. Since every $G \in \mathcal{G}$ is an α-expander, for $\gamma \in \mathcal{C}^g$ we have $B(\gamma) \geq \alpha|\gamma|$ and hence $w^g_\gamma \leq e^{-\tau|\gamma|}$. ◀

▶ **Lemma 21** ([17, Lemma 12])**.** *For any n-vertex α-expander graph G and $\beta \geq 2\log(eq)/\alpha$, $qZ^g(G)$ is an e^{-n}-approximation of the Potts partition function $Z_{G,\beta}$.*

Proof of Theorem 9. Let $\mathcal{G}$ be the class of α-expander graphs of maximum degree at most Δ. Clearly, the polymer models $(\mathcal{C}^g(\cdot), w^g, \mathcal{G})$ are computationally feasible. By Lemma 20, the models also satisfy the polymer sampling condition and therefore Theorems 5 and 6 apply. Consider any n-vertex graph $G \in \mathcal{G}$. Since $\beta \geq \frac{5+3\log((q-1)\Delta)}{\alpha} > \frac{2\log(eq)}{\alpha}$, Lemma 21 applies to G.

For the sampling algorithm, we pick a color $g \in [q]$ uniformly at random and generate an (ε/q)-approximate sample from the Gibbs measure associated to $Z^g(G)$ using the algorithm of Theorem 5, in time $O(n\log(n/\varepsilon))$. By Lemma 21, we conclude that the resulting output is an ε-approximate sample for the Potts model.

For the counting algorithm, we pick an arbitrary $g \in [q]$ and produce using the algorithm of Theorem 6 a number $\hat{Z}$ in time $O((n/\varepsilon)^2 \log^2(n/\varepsilon))$, which is an $\varepsilon/(2q)$-approximation to $Z^g(G)$ with probability $\geq 3/4$. By Lemma 21, we conclude that $q\hat{Z}$ is an ε-approximation for the partition function of the Potts model (with the same probability). ◀

References

1. A. Barvinok. *Combinatorics and Complexity of Partition Functions*. Algorithms and Combinatorics. Springer International Publishing, 2017.
2. I. Bezáková, D. Štefankovič, V. V. Vazirani, and E. Vigoda. Accelerating simulated annealing for the permanent and combinatorial counting problems. *SIAM Journal on Computing*, 37(5):1429–1454, 2008.
3. C. Borgs. Absence of zeros for the chromatic polynomial on bounded degree graphs. *Combinatorics, Probability and Computing*, 15(1-2):63–74, 2006.
4. C. Borgs, J. T. Chayes, A. Frieze, J. H. Kim, P. Tetali, E. Vigoda, and V. H. Vu. Torpid mixing of some Monte Carlo Markov chain algorithms in statistical physics. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 218–229, 1999.
5. C. Borgs, J. T. Chayes, J. Kahn, and L. Lovász. Left and right convergence of graphs with bounded degree. *Random Structures & Algorithms*, 42(1):1–28, 2013.
6. C. Borgs and J. Z. Imbrie. A unified approach to phase diagrams in field theory and statistical mechanics. *Communications in mathematical physics*, 123(2):305–328, 1989.
7. R. L. Dobrushin. Estimates of semi-invariants for the Ising model at low temperatures. *Translations of the American Mathematical Society-Series 2*, 177:59–82, 1996.
8. M. E. Dyer and C. S. Greenhill. On Markov Chains for Independent Sets. *J. Algorithms*, 35(1):17–49, 2000. doi:10.1006/jagm.1999.1071.
9. Martin Dyer and Catherine Greenhill. Random walks on combinatorial objects. *London Mathematical Society Lecture Note Series*, pages 101–136, 1999.
10. C. Efthymiou, T. P. Hayes, D. Štefankovič, E. Vigoda, and Y. Yin. Convergence of MCMC and Loopy BP in the Tree Uniqueness Region for the Hard-Core Model. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 704–713, 2016.
11. R. Fernández, P. A. Ferrari, and N. L. Garcia. Loss network representation of Peierls contours. *Annals of Probability*, 29(2):902–937, 2001.
12. D. Galvin and P. Tetali. Slow mixing of Glauber dynamics for the hard-core model on regular bipartite graphs. *Random Structures & Algorithms*, 28(4):427–443, 2006.
13. C. Gruber and H. Kunz. General properties of polymer systems. *Communications in Mathematical Physics*, 22(2):133–161, 1971.
14. T. Helmuth, W. Perkins, and G. Regts. Algorithmic Pirogov-Sinai theory. *arXiv preprint*, arXiv:1806.11548, 2018. arXiv:1806.11548.
15. M. Huber. Approximation algorithms for the normalizing constant of Gibbs distributions. *The Annals of Applied Probability*, 25(2):974–985, 2015.
16. M. Jenssen, P. Keevash, and W. Perkins. Algorithms for #BIS-hard problems on expander graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2235–2247. SIAM, 2019.
17. M. Jenssen, P. Keevash, and W. Perkins. Algorithms for #BIS-hard problems on expander graphs. *arXiv preprint*, 2019. arXiv:1807.04804v2.
18. R. Kotecký and D. Preiss. Cluster expansion for abstract polymer models. *Communications in Mathematical Physics*, 103(3):491–498, 1986. URL: http://projecteuclid.org/euclid.cmp/1104114796.

19 L. Laanait, A. Messager, S. Miracle-Solé, J. Ruiz, and S. Shlosman. Interfaces in the Potts model I: Pirogov-Sinai theory of the Fortuin-Kasteleyn representation. *Communications in Mathematical Physics*, 140(1):81–91, 1991.

20 C. Liao, J. Lin, P. Lu, and Z. Mao. Counting independent sets and colorings on random regular bipartite graphs. *arXiv preprint*, 2019. `arXiv:1903.07531`.

21 J. Liu and P. Lu. FPTAS for #BIS with degree bounds on one side. In *Proceedings of the 47th Annual ACM on Symposium on Theory of Computing (STOC)*, pages 549–556, 2015.

22 E. Mossel, D. Weitz, and N. Wormald. On the hardness of sampling independent sets beyond the tree threshold. *Probability Theory and Related Fields*, 143(3-4):401–439, 2009.

23 V. Patel and G. Regts. Deterministic polynomial-time approximation algorithms for functions and graph polynomials. *SIAM Journal on Computing*, 46(6):1893–1919, 2017.

24 S. A. Pirogov and Ya. G. Sinai. Phase diagrams of classical lattice systems. *Teoret. Mat. Fiz.*, 25(3):358–369, 1975.

25 D. Štefankovič, S. Vempala, and E. Vigoda. Adaptive simulated annealing: A near-optimal connection between sampling and counting. *Journal of the ACM*, 56(3):18, 2009.

26 D. Weitz. Counting independent sets up to the tree threshold. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 140–149, 2006.

Deterministic Approximation of Random Walks in Small Space

Jack Murtagh
School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA
http://scholar.harvard.edu/jmurtagh
jmurtagh@g.harvard.edu

Omer Reingold
Computer Science Department, Stanford University, Stanford, CA USA
reingold@stanford.edu

Aaron Sidford
Management Science & Engineering, Stanford University, Stanford, CA USA
http://www.aaronsidford.com/
sidford@stanford.edu

Salil Vadhan
School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA
http://salil.seas.harvard.edu/
salil_vadhan@harvard.edu

Abstract

We give a deterministic, nearly logarithmic-space algorithm that given an undirected graph G, a positive integer r, and a set S of vertices, approximates the conductance of S in the r-step random walk on G to within a factor of $1 + \epsilon$, where $\epsilon > 0$ is an arbitrarily small constant. More generally, our algorithm computes an ϵ-spectral approximation to the normalized Laplacian of the r-step walk.

Our algorithm combines the derandomized square graph operation [21], which we recently used for solving Laplacian systems in nearly logarithmic space [16], with ideas from [5], which gave an algorithm that is time-efficient (while ours is space-efficient) and randomized (while ours is deterministic) for the case of even r (while ours works for all r). Along the way, we provide some new results that generalize technical machinery and yield improvements over previous work. First, we obtain a nearly linear-time randomized algorithm for computing a spectral approximation to the normalized Laplacian for odd r. Second, we define and analyze a generalization of the derandomized square for irregular graphs and for sparsifying the product of two distinct graphs. As part of this generalization, we also give a strongly explicit construction of expander graphs of every size.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization; Theory of computation → Random walks and Markov chains

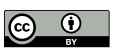
Keywords and phrases random walks, space complexity, derandomization, spectral approximation, expander graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.42

Category RANDOM

Related Version A full version of this paper is available at https://arxiv.org/abs/1903.06361.

Funding *Jack Murtagh*: Supported by NSF grant CCF-1763299.
Omer Reingold: Supported by NSF grant CCF-1763311.
Salil Vadhan: Supported by NSF grant CCF-1763299.



© Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan;
licensed under Creative Commons License CC-BY



1 Introduction

Random walks provide the most dramatic example of the power of randomized algorithms for solving computational problems in the space-bounded setting, as they only require logarithmic space (to store the current state or vertex). In particular, since undirected graphs have polynomial cover time, random walks give a randomized logspace (**RL**) algorithm for Undirected S-T Connectivity [1]. Reingold [19] showed that this algorithm can be derandomized, and hence that Undirected S-T Connectivity is in deterministic logspace (**L**). However, Reingold's algorithm does not match the full power of random walks on undirected graphs; in particular it does not allow us to approximate properties of the random walk at lengths below the mixing time.

In this work, we provide a nearly logarithmic-space algorithm for approximating properties of arbitrary-length random walks on an undirected graph, in particular the *conductance* of any set of vertices:

▶ **Definition 1.** *Let $G = (V, E)$ be an undirected graph, r a positive integer, and $S \subseteq V$ a set of vertices. The* conductance *of S under the r-step random walk on G is defined as*

$$\Phi_r(S) = \Pr[V_r \notin S | V_0 \in S],$$

where $V_0, V_1, \ldots, V_r$ is a random walk on G started at the stationary distribution $\Pr[V_0 = v] = \deg(v)/2|E|$.

▶ **Theorem 2.** *There is a deterministic algorithm that given an undirected multigraph G on n vertices, a positive integer r, a set of vertices S, and $\epsilon > 0$, computes a number $\tilde{\Phi}$ such that*

$$(1 - \epsilon) \cdot \Phi_r(S) \leq \tilde{\Phi} \leq (1 + \epsilon) \cdot \Phi_r(S)$$

and runs in space $O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)$, where N is the bit length of the input graph G.

Previously, approximating conductance could be done in $O(\log^{3/2}(N/\epsilon) + \log\log r)$ space, which follows from Saks' and Zhou's proof that **RL** is in $\mathbf{L}^{3/2}$ [22].

Two interesting parameter regimes where we improve the Saks-Zhou bound are when $r = 1/\epsilon = 2^{O(\sqrt{\log N})}$, in which case our algorithm runs in space $O(\log N)$, or when $\epsilon = 1/\text{polylog}(N)$ and $r \leq \text{poly}(N)$, in which case our algorithm runs in space $\tilde{O}(\log N)$. When r exceeds the $\text{poly}(N) \cdot \log(1/\epsilon)$ time for random walks on undirected graphs to mix to within distance ϵ of the stationary distribution, the conductance can be approximated in space $O(\log(N/\epsilon) + \log\log r)$ by using Reingold's algorithm to find the connected components of G, and the bipartitions of the components that are bipartite and calculating the stationary probability of S restricted to each of these pieces, which is proportional to the sum of degrees of vertices in S.

We prove Theorem 2 by providing a stronger result that with the same amount of space it is possible to compute an *ϵ-spectral approximation* to the *normalized Laplacian* of the r-step random walk on G.

▶ **Definition 3.** *Let G be an undirected graph with adjacency matrix A, diagonal degree matrix D, and transition matrix $T = AD^{-1}$. The* transition matrix for the r-step random walk *on G is T^r. The* normalized Laplacian *of the r-step random walk is the symmetric matrix $I - M^r$ for $M = D^{-1/2}AD^{-1/2}$.*

Note that the normalized Laplacian can also be expressed as $I - M^r = D^{-1/2}(I - T^r)D^{1/2}$, so it does indeed capture the behavior of r-step random walks on G.[1]

▶ **Theorem 4** (Main result). *There is a deterministic algorithm that given an undirected multigraph G on n vertices with normalized Laplacian $I - M$, a nonnegative integer r, and $\epsilon > 0$, constructs an undirected multigraph $\tilde{G}$ whose normalized Laplacian $\tilde{L}$ is an ϵ-spectral approximation of $L = I - M^r$. That is, for all vectors $v \in \mathbb{R}^n$*

$$(1 - \epsilon) \cdot v^T L v \leq v^T \tilde{L} v \leq (1 + \epsilon) \cdot v^T L v.$$

The algorithm runs in space $O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)$, where N is the bit length of the input graph G.

Theorem 2 follows from Theorem 4 by taking v to be $D^{1/2}e_S$ where e_S is the characteristic vector of the set S and normalizing appropriately (See Section 5).

Our main technique for proving Theorem 4 is the *derandomized product*, a new generalization of the *derandomized square*, which was introduced by Rozenman and Vadhan [21] to give an alternative proof that
Undirected S-T Connectivity is in **L**. Our main result follows from carefully applying the derandomized product and analyzing its properties with inequalities from the theory of spectral approximation. Specifically, our analysis is inspired by the work of Cheng, Cheng, Liu, Peng, and Teng [5], who studied the approximation of random walks by randomized algorithms running in nearly linear time. We emphasize that the work of [5] gives a randomized algorithm with high space complexity (but low time complexity) for approximating properties of even length walks while we give a deterministic, space-efficient algorithm for approximating properties of walks of every length. Interestingly, while the graphs in our results are all undirected, some of our analyses use techniques for spectral approximation of *directed* graphs introduced by Cohen, Kelner, Peebles, Peng, Rao, Sidford, and Vladu [7, 6].

The derandomized square can be viewed as applying the pseudorandom generator of Impagliazzo, Nisan, and Wigderson [10] to random walks on labelled graphs. It is somewhat surprising that repeated derandomized squaring does not blow up the error by a factor proportional to the length of the walk being derandomized. For arbitrary branching programs, the INW generator does incur error that is linear in the length of the program. Some special cases such as regular [3, 4, 8] and permutation [8, 24] branching programs of constant width have been shown to have a milder error growth as a function of the walk length. Our work adds to this list by showing that properties of random walks of length k on undirected graphs can be estimated in terms of spectral approximation without error accumulating linearly in k.

In our previous work [16], we showed that the Laplacian of the derandomized square of a *regular* graph spectrally approximates the Laplacian of the true square, $I - M^2$, and this was used in a recursion from [18] to give a nearly logarithmic-space algorithm for approximately solving Laplacian systems $Lx = b$. A natural idea to approximate the Laplacian of higher powers, $I - M^r$, is to repeatedly derandomized square. This raises three challenges, and we achieve our result by showing how to overcome each:

1. It is not guaranteed from [16] that repeated derandomized squaring preserves spectral approximation. For this, we use ideas from [5] to argue that it does.
2. When r is not a power of 2, the standard approach would be to write $r = b_0 + 2 \cdot b_1 + \ldots + 2^z \cdot b_z$ where b_i is the ith bit of r and multiply approximations to M^{2^i} for all i such that $b_i \neq 0$. The problem is that multiplying spectral approximations of matrices

[1] When G is irregular, the matrix $I - T^r$ is not necessarily symmetric. It is a *directed Laplacian* as defined in [7, 6]. See Definition 9.

does not necessarily yield a spectral approximation of their product. Our solution is to generalize the derandomized square to produce sparse approximations to the product of *distinct* graphs. In particular, given $I - M$ and an approximation $I - \tilde{M}$ to $I - M^k$, our derandomized product allows us to combine M and $\tilde{M}$ to approximate $I - M^{k+1}$. Although our generalized graph product is defined for undirected graphs, its analysis uses machinery for spectral approximation of directed graphs, introduced in [6].

3. We cannot assume that our graph is regular without loss of generality. In contrast, [19, 21, 16] could do so, since adding self-loops does not affect connectivity or solutions to Laplacian systems of G, however, it does affect random walks. Our solution is to define and analyze the derandomized product for irregular graphs.

A key element in the derandomized product is a strongly explicit (i.e. neighbor relations can be computed in space $O(\log N)$) construction of expander graphs whose sizes equal the degrees of the vertices in the graphs being multiplied. This is problematic when we are not free to add self loops to the graphs because strongly explicit constructions of expander graphs only exist for graph sizes that are certain subsets of $\mathbb{N}$ such as powers of 2 (Cayley graphs based on [17] and [2]), perfect squares [14, 9], and other size distributions [20] or are only explicit in the sense of running time or parallel work [13]. To address this issue, we give a strongly explicit construction of expander graphs of all sizes by giving a reduction from existing strongly explicit constructions in Section 3.

Many of our techniques are inspired by Cheng, Cheng, Liu, Peng, and Teng [5], who gave two algorithms for approximating random walks. One is a nearly linear time randomized algorithm for approximating random walks of *even* length and another works for all walk lengths r but has a running time that is quadratic in r, and so only yields a nearly linear time algorithm for r that is polylogarithmic in the size of the graph. In addition, [11] studied the problem of computing sparse spectral approximations of random walks but the running time in their work also has a quadratic dependence on r. We extend these results by giving a nearly linear time randomized algorithm for computing a spectral approximation to $I - M^r$ for *all* r. This is discussed in Section 5.

2 Preliminaries

2.1 Spectral Graph Theory

Given an undirected multigraph G the *Laplacian* of G is the symmetric matrix $D - A$, where D is the diagonal matrix of vertex degrees and A is the adjacency matrix of G. The *transition matrix* of the random walk on G is $T = AD^{-1}$. T_{ij} is the probability that a uniformly random edge from vertex j leads to vertex i (i.e. the number of edges between j and i divided by the degree of j). The *normalized Laplacian* of G is the symmetric matrix $I - M = D^{-1/2}(D - A)D^{-1/2}$. Note that when G is regular, the matrix $M = D^{-1/2}AD^{-1/2} = AD^{-1} = T$. The *transition matrix of the r-step random walk* on G is T^r. For all probability distributions π, $T^r\pi$ is the distribution over vertices that results from picking a random vertex according to π and then running a random walk on G for r steps. The transition matrix of the r-step random walk on G is related to the normalized Laplacian in the following way:

$$I - M^r = D^{-1/2}(I - T^r)D^{1/2}.$$

For undirected multigraphs, the matrix $M = D^{-1/2}AD^{-1/2}$ has real eigenvalues between -1 and 1 and so $I - M^r$ has eigenvalues in $[0, 2]$ and thus is positive semidefinite (PSD). The *spectral norm* of a real matrix M, denoted $\|M\|$, is the largest singular value of M. That is,

the square root of the largest eigenvalue of $M^T M$. When M is symmetric, $\|M\|$ equals the largest eigenvalue of M in absolute value. For an undirected graph G with adjacency matrix A, we write $k \cdot G$ to denote the graph with adjacency matrix $k \cdot A$, i.e. the multigraph G with all edges duplicated to have multiplicity k.

Given a symmetric matrix L, its *Moore-Penrose Pseudoinverse*, denoted $L^\dagger$, is the unique matrix with the same eigenvectors as L such that for each eigenvalue λ of L, the corresponding eigenvalue of $L^\dagger$ is $1/\lambda$ if $\lambda \neq 0$ and 0 otherwise. When L is a Laplacian, we write $L^{\dagger/2}$ to denote the unique symmetric PSD matrix square root of the pseudoinverse of L.

To measure the approximation between graphs we use spectral approximation[2][23]:

▶ **Definition 5.** *Let $L, \tilde{L} \in \mathbb{R}^{n \times n}$ be symmetric PSD matrices. We say that $\tilde{L}$ is an ϵ-approximation of L (written $\tilde{L} \approx_\epsilon L$) if for all vectors $v \in \mathbb{R}^n$*

$$(1-\epsilon) \cdot v^T L v \leq v^T \tilde{L} v \leq (1+\epsilon) \cdot v^T L v.$$

Note that Definition 5 is not symmetric in L and $\tilde{L}$. Spectral approximation can also be written in terms of the Loewner partial ordering of PSD matrices:

$$(1-\epsilon) \cdot L \preceq \tilde{L} \preceq (1+\epsilon) \cdot L$$

where for two matrices A, B, we write $A \preceq B$ if $B - A$ is PSD. Spectral approximation has a number of useful properties listed in the following proposition.

▶ **Proposition 6.** *If $W, X, Y, Z \in \mathbb{R}^{n \times n}$ are PSD symmetric matrices then:*

1. *If $X \approx_\epsilon Y$ for $\epsilon < 1$ then $Y \approx_{\epsilon/(1-\epsilon)} X$*
2. *If $X \approx_{\epsilon_1} Y$ and $Y \approx_{\epsilon_2} Z$ then $X \approx_{\epsilon_1+\epsilon_2+\epsilon_1 \cdot \epsilon_2} Z$,*
3. *If $X \approx_\epsilon Y$ and V is any $n \times n$ matrix then $V^T X V \approx_\epsilon V^T Y V$,*
4. *If $X \approx_\epsilon Y$ then $X + Z \approx_\epsilon Y + Z$,*
5. *If $W \approx_{\epsilon_1} X$ and $Y \approx_{\epsilon_2} Z$ then $W + Y \approx_{\max\{\epsilon_1,\epsilon_2\}} X + Z$, and*
6. *If $X \approx_\epsilon Y$ then $c \cdot X \approx_\epsilon c \cdot Y$ for all nonnegative scalars c*

For regular undirected graphs, we use the measure introduced by [15] for the rate at which a random walk converges to the uniform distribution.

▶ **Definition 7** ([15])**.** *Let G be a regular undirected graph with transition matrix T. Define*

$$\lambda(G) = \max_{\substack{v \perp \vec{1} \\ v \neq 0}} \frac{\|Tv\|}{\|v\|} = \text{2nd largest absolute value of the eigenvalues of } T \in [0,1].$$

$1 - \lambda(G)$ is called the spectral gap *of G.*

$\lambda(G)$ is known to be a measure of how well-connected a graph is. The smaller $\lambda(G)$, the faster a random walk on G converges to the uniform distribution. Graphs G with $\lambda(G)$ bounded away from 1 are called *expanders*. Expanders can equivalently be characterized as graphs that spectrally approximate the complete graph. This is formalized in the next lemma.

▶ **Lemma 8.** *Let H be a c-regular undirected multigraph on n vertices with transition matrix T and let $J \in \mathbb{R}^{n \times n}$ be a matrix with $1/n$ in every entry (i.e. J is the transition matrix of the complete graph with a self loop on every vertex). Then $\lambda(H) \leq \lambda$ if and only if $I - T \approx_\lambda I - J$.*

[2] In [16], we use an alternative definition of spectral approximation where $\tilde{L} \approx_\epsilon L$ if for all $v \in \mathbb{R}^n$, $e^{-\epsilon} \cdot v^T L v \leq v^T \tilde{L} v \leq e^\epsilon \cdot v^T L v$. We find Definition 5 more convenient for this paper.

A proof of Lemma 8 can be found in the full version of the paper. In [6] Cohen, Kelner, Peebles, Peng, Rao, Sidford, and Vladu introduced a definition of spectral approximation for *asymmetric matrices*. Although the results in our paper only concern undirected graphs, some of our proofs use machinery from the theory of directed spectral approximation.

▶ **Definition 9** (Directed Laplacian [7, 6]). *A matrix $L \in \mathbb{R}^{n\times n}$ is called a* directed Laplacian *if $L_{ij} \leq 0$ for all $i \neq j$ and $L_{ii} = -\sum_{j\neq i} L_{ji}$ for all $i \in [n]$. The associated directed graph has n vertices and an edge (i, j) of weight $-L_{ji}$ for all $i \neq j \in [n]$ with $L_{ji} \neq 0$.*

▶ **Definition 10** (Asymmetric Matrix Approximation [6]). *Let $\tilde{L}$ and L be (possibly asymmetric) matrices such that $U = (L + L^T)/2$ is PSD. We say that $\tilde{L}$ is a* directed ϵ-approximation *of L if:*

1. $\ker(U) \subseteq \ker(\tilde{L} - L) \cap \ker((\tilde{L} - L)^T)$, *and*
2. $\left\|U^{\dagger/2}(\tilde{L} - L)U^{\dagger/2}\right\|_2 \leq \epsilon$

Below we state some useful lemmas about directed spectral approximation. The first gives an equivalent formulation of Definition 10.

▶ **Lemma 11** ([6] Lemma 3.5). *Let $L \in \mathbb{R}^{n\times n}$ be a (possibly asymmetric) matrix and let $U = (L + L^T)/2$. A matrix $\tilde{L}$ is a directed ϵ-approximation of L if and only if for all vectors $x, y \in \mathbb{R}^n$*

$$x^T(\tilde{L} - L)y \leq \frac{\epsilon}{2} \cdot (x^T U x + y^T U y).$$

▶ **Lemma 12** ([6] Lemma 3.6). *Suppose $\tilde{L}$ is a directed ϵ-approximation of L and let $U = (L + L^T)/2$ and $\tilde{U} = (\tilde{L} + \tilde{L}^T)/2$. Then $\tilde{U} \approx_\epsilon U$.*

Lemma 12 says that directed spectral approximation implies the usual notion from Definition 5 for "symmetrized" versions of the matrices L and $\tilde{L}$. In fact, when the matrices L and $\tilde{L}$ are both symmetric, the two definitions are equivalent:

▶ **Lemma 13.** *Let $\tilde{L}$ and L be symmetric PSD matrices. Then $\tilde{L}$ is a directed ϵ-approximation of L if and only if $\tilde{L} \approx_\epsilon L$.*

A proof of Lemma 13 can be found in the full version of the paper.

2.2 Space Bounded Computation

We use a standard model of space-bounded computation where the machine $\mathcal{M}$ has a read-only input tape, a constant number of read/write work tapes, and a write-only output tape. If throughout every computation on inputs of length at most n, $\mathcal{M}$ uses at most $s(n)$ total tape cells on all the work tapes, we say $\mathcal{M}$ runs in space $s = s(n)$. Note that $\mathcal{M}$ may write more than s cells (in fact as many as $2^{O(s)}$) but the output tape is write-only. The following proposition describes the behavior of space complexity when space bounded algorithms are composed.

▶ **Proposition 14.** *Let f_1, f_2 be functions that can be computed in space $s_1(n), s_2(n) \geq \log n$, respectively, and f_1 has output of length at most $\ell_1(n)$ on inputs of length n. Then $f_2 \circ f_1$ can be computed in space*

$$O(s_2(\ell_1(n)) + s_1(n)).$$

2.3 Rotation Maps

In the space-bounded setting, it is convenient to use local descriptions of graphs. Such descriptions allow us to navigate large graphs without loading them entirely into memory. For this we use *rotation maps*, functions that describe graphs through their neighbor relations. Rotation maps are defined for graphs with labeled edges as described in the following definition.

▶ **Definition 15** ([20]). *A* two-way labeling *of an undirected multigraph $G = (V, E)$ with vertex degrees $(d_v)_{v \in V}$, is a labeling of the edges in G such that*

1. *Every edge $(u, v) \in E$ has two labels: one in $[d_u]$ as an edge incident to u and one in $[d_v]$ as an edge incident to v,*
2. *For every vertex $v \in V$, the labels of the d_v edges incident to v are distinct.*

In [21], two-way labelings are referred to as *undirected* two-way labelings. Note that every graph has a two-way labeling where each vertex "names" its neighbors uniquely in some canonical way based on the order they're represented in the input. We will describe multigraphs with two-way labelings using rotation maps:

▶ **Definition 16** ([20]). *Let G be an undirected multigraph on n vertices with a two-way labeling. The* rotation map *Rot_G is defined as follows: $Rot_G(v, i) = (w, j)$ if the ith edge to vertex v leads to vertex w and this edge is the jth edge incident to w.*

We will use expanders that have efficiently computable rotation maps. We call such graphs *strongly explicit*. The usual definition of strong explicitness only refers to time complexity, but we will use it for both time and space.

▶ **Definition 17.** *A family of two-way labeled graphs $\mathcal{G} = \{G_{n,c}\}_{(n,c)}$, where $G_{n,c}$ is a c-regular graph on n vertices, is called* strongly explicit *if given n, c, a vertex $v \in [n]$ and an edge label $a \in [c]$, $Rot_{G_{n,c}}(v, a)$ can be computed in time* $\mathrm{poly}(\log(nc))$ *and space* $O(\log nc)$.

3 The Derandomized Product and Expanders of All Sizes

In this section we introduce our derandomized graph product. The derandomized product generalizes the *derandomized square* graph operation that was introduced by Rozenman and Vadhan [21] to give an alternative proof that Undirected S-T Connectivity is in **L**. Unlike the derandomized square, the derandomized product is defined for *irregular* graphs and produces a sparse approximation to the product of any two (potentially different) graphs with the same vertex degrees.

Here, by the "product" of two graphs G_0, G_1, we mean the reversible Markov chain with transitions defined as follows: from a vertex v, with probability $1/2$ take a random step on G_0 followed by a random step on G_1 and with probability $1/2$ take a random step on G_1 followed by a random step on G_0.

When $G_0 = G_1 = G$, this is the same as taking a 2-step random walk on G. Note, however, that when G is irregular, a 2-step random walk is *not* equivalent to doing a 1-step random walk on the graph G^2, whose edges correspond to paths of length 2 in G. Indeed, even the stationary distribution of the random walk on G^2 may be different than on G.[3] Nevertheless, our goal in the derandomized product is to produce a relatively sparse graph whose 1-step random walk approximates the 2-step random walk on G.

[3] For example, let G be the graph on two vertices with one edge (u, v) connecting them and a single self loop on u. Then $[2/3, 1/3]$ is the stationary distribution of G and $[3/5, 2/5]$ is the stationary distribution of G^2.

The intuition behind the derandomized product is as follows: rather than build a graph with every such two-step walk, we use expander graphs to pick a pseudorandom subset of the walks. Specifically, we first pick $b \in \{0,1\}$ at random. Then, as before we take a truly random step from v to u in G_b. But for the second step, we don't use an arbitrary edge leaving u in $G_{\bar{b}}$, but rather correlate it to the edge on which we arrived at u using a c-regular expander on $\deg(u)$ vertices, where we assume that the vertex degrees in G_0 and G_1 are the same. When $c < \deg(u)$, the vertex degrees of the resulting two-step graph will be sparser than without derandomization. However using the pseudorandom properties of expander graphs, we can argue that the derandomized product is a good approximation of the true product.

▶ **Definition 18** (Derandomized Product). *Let G_0, G_1 be undirected multigraphs on n vertices with two-way labelings and identical vertex degrees $d_1, d_2, \ldots, d_n$. Let $\mathcal{H} = \{H_i\}$ be a family of two-way labeled, c-regular expanders of sizes including $d_1, \ldots, d_n$. The* derandomized product with respect to $\mathcal{H}$, *denoted $G_0 \circledcirc_{\mathcal{H}} G_1$, is an undirected multigraph on n vertices with vertex degrees $2 \cdot c \cdot d_1, \ldots, 2 \cdot c \cdot d_n$ and rotation map $Rot_{G_0 \circledcirc_{\mathcal{H}} G_1}$ defined as follows: For $v_0 \in [n], j_0 \in [d_{v_0}]$, $a_0 \in [c]$, and $b \in \{0,1\}$ we compute $Rot_{G_0 \circledcirc_{\mathcal{H}} G_1}(v_0, (j_0, a_0, b))$ as*

1. *Let $(v_1, j_1) = Rot_{G_b}(v_0, j_0)$*
2. *Let $(j_2, a_1) = Rot_{H_{d_{v_1}}}(j_1, a_0)$*
3. *Let $(v_2, j_3) = Rot_{G_{\bar{b}}}(v_1, j_2)$*
4. *Output $(v_2, (j_3, a_1, \bar{b}))$*

where $\bar{b}$ denotes the bit-negation of b.

Note that when $G_0 = G_1$ the derandomized product generalizes the derandomized square [21] to irregular graphs, albeit with each edge duplicated twice. To see that $G_0 \circledcirc_{\mathcal{H}} G_1$ is undirected, one can check that $\mathrm{Rot}_{G_0 \circledcirc_{\mathcal{H}} G_1}(\mathrm{Rot}_{G_0 \circledcirc_{\mathcal{H}} G_1}(v_0, (j_0, a_0, b))) = (v_0, (j_0, a_0, b)))$.

Note that Definition 18 requires that each vertex i has the same degree d_i in G_0 and G_1, ensuring that the random walks on G_0, G_1, and $G_0 \circledcirc_{\mathcal{H}} G_1$ all have the same stationary distribution. This can be generalized to the case that there is an integer k such that for each vertex v with degree d_v in G_1, v has degree $k \cdot d_v$ in G_0. For this, we can duplicate each edge in G_1 k times to match the degrees of G_0 and then apply the derandomized product to the result. In such cases we abuse notation and write $G_0 \circledcirc_{\mathcal{H}} G_1$ to mean $G_0 \circledcirc_{\mathcal{H}} k \cdot G_1$.

In [16] we showed that the derandomized square produces a spectral approximation to the true square. We now show that the derandomized product also spectrally approximates a natural graph product.

▶ **Theorem 19.** *Let G_0, G_1 be undirected multigraphs on n vertices with two-way labelings, and normalized Laplacians $I - M_0$ and $I - M_1$. Let G_0 have vertex degrees $d_1, \ldots, d_n$ and G_1 have vertex degrees $d'_1, \ldots, d'_n$ where for all $i \in [n]$, $d_i = k \cdot d'_i$ for a positive integer k. Let $\mathcal{H} = \{H_i\}$ be a family of two-way labeled, c-regular expanders with $\lambda(H_i) \leq \lambda$ for all $H_i \in \mathcal{H}$, of sizes including $d_1, \ldots, d_n$. Let $I - \tilde{M}$ be the normalized Laplacian of $\tilde{G} = G_0 \circledcirc_{\mathcal{H}} G_1$. Then*

$$I - \tilde{M} \approx_\lambda I - \frac{1}{2} \cdot (M_0 M_1 + M_1 M_0).$$

A proof of Theorem 19 can be found in Appendix A.

Note that for a graph G with normalized Laplacian $I - M$ and transition matrix T, approximating $I - \frac{1}{2} \cdot (M_0 M_1 + M_1 M_0)$ as in Theorem 19 for $M_0 = M^{k_0}$ and $M_1 = M^{k_1}$ gives a form of approximation to random walks of length $k_1 + k_2$ on G, as

$$\begin{aligned} I - T^{k_1+k_2} &= D^{1/2}(I - M^{k_1+k_2})D^{-1/2} \\ &= I - \frac{1}{2} \cdot D^{1/2}(M_0 M_1 + M_1 M_0)D^{-1/2}. \end{aligned}$$

To apply the derandomized product, we need an expander family $\mathcal{H}$ with sizes equal to all of the vertex degrees. However, existing constructions of strongly explicit expander families only give graphs of sizes that are subsets of $\mathbb{N}$ such as all powers of 2 or all perfect squares. In [21, 16] this was handled by adding self loops to make the vertex degrees all equal and matching the sizes of expanders in explicit families. Adding self loops was acceptable in those works because it does not affect connectivity (the focus of [21]) or the Laplacian (the focus of [16]). However it does affect long random walks (our focus), so we cannot add self loops. Instead, we show how to obtain strongly explicit expanders of all sizes. Our construction works by starting with a strongly explicit expander from one of the existing constructions and merging vertices to achieve any desired size:

▶ **Theorem 20.** *There exists a family of strongly explicit expanders $\mathcal{H}$ such that for all $n > 1$ and $\lambda \in (0, 1)$ there is a $c = \mathrm{poly}(1/\lambda)$ and a c-regular graph $H_{n,c} \in \mathcal{H}$ on n vertices with $\lambda(H_{n,c}) \leq \lambda$.*

A proof of Theorem 20 can be found in Appendix B.

4 Main Result

In this section we prove Theorem 4, our main result regarding space bounded computation of the normalized Laplacian of the r-step random walk on G.

The algorithm described below is inspired by techniques used in [5] to approximate random walks with a randomized algorithm in nearly linear time. Our analyses use ideas from the work of Cohen, Kelner, Peebles, Peng, Rao, Sidford, and Vladu on *directed* Laplacian system solvers even though all of the graphs we work with are undirected.

4.1 Algorithm Description and Proof Overview

Let $I - M$ be the normalized Laplacian of our input and r be the target power. We will first describe an algorithm for computing $I - M^r$ without regard for space complexity and then convert it into a space-efficient approximation algorithm. The algorithm iteratively approximates larger and larger powers of M. On a given iteration, we will have computed $I - M^k$ for some $k < r$ and we use the following operations to increase k:

- Square: $I - M^k \to I - M^{2k}$,
- Plus one: $I - M^k \to I - \frac{1}{2} \cdot (M \cdot M^k + M^k \cdot M) = I - M^{k+1}$.

Interleaving these two operations appropriately can produce any power r of M, invoking each operation at most $\log_2 r$ times. To see this, let $b_z b_{z-1} \dots b_0$ be the bits of r in its binary representation where b_0 is the least significant bit and $b_z = 1$ is the most significant. We are given $I - M = I - M^{b_z}$. The algorithm will have z iterations and each one will add one more bit from most significant to least significant to the binary representation of the exponent. So after iteration i we will have $I - M^{b_z b_{z-1} \dots b_{z-i}}$.

For iterations $1, \dots, z$, we read the bits of r from b_{z-1} to b_0 one at a time. On each iteration we start with some power $I - M^k$. If the corresponding bit is a 0, we square to create $I - M^{2k}$ (which adds a 0 to the binary representation of the current exponent) and proceed to the next iteration. If the corresponding bit is a 1, we square and then invoke a plus one operation to produce $I - M^{2k+1}$ (which adds a 1 to the binary representation of the current exponent). After iteration z we will have $I - M^r$.

Implemented recursively, this algorithm has $\log_2 r$ levels of recursion and uses $O(\log N)$ space at each level for the matrix multiplications, where N is the bit length of the input graph. This results in total space $O(\log r \cdot \log N)$, which is more than we want to use (cf. Theorem 4). We reduce the space complexity by replacing each square and plus one operation with the corresponding derandomized product, discussed in Section 3.

Theorem 19 says that the derandomized product produces spectral approximations to the square and the plus one operation. Since we apply these operations repeatedly on successive approximations, we need to maintain our ultimate approximation to a power of $I - M$. In other words, we need to show that given $\tilde{G}$ such that $I - \tilde{M} \approx_\epsilon I - M^k$ we have:

1. $I - \tilde{M}^2 \approx_\epsilon I - M^{2k}$
2. $I - \frac{1}{2} \cdot (M\tilde{M} + \tilde{M}M) \approx_\epsilon I - M^{k+1}$.

We prove these in Lemmas 21 and 22. The transitive property of spectral approximation (Proposition 6 Part 2) will then complete the proof of spectral approximation.

We only know how to prove items 1 and 2 when M^k is PSD. This is problematic because M is not guaranteed to be PSD for arbitrary graphs and so M^k may only be PSD when k is even. Simple solutions like adding self loops (to make the random walk lazy) are not available to us because loops may affect the random walk behavior in unpredictable ways. Another attempt would be to replace the plus one operation in the algorithm with a "plus two" operation

- Plus two: $I - M^k \to I - \frac{1}{2} \cdot (M^2 \cdot M^k + M^k \cdot M^2) = I - M^{k+2}$.

Interleaving the square and plus two would preserve the positive semidefiniteness of the matrix we're approximating and can produce any even power of M. If r is odd, we could finish with one plus one operation, which will produce a spectral approximation because $I - M^{r-1}$ is PSD. A problem with this approach is that the derandomized product is defined only for unweighted multigraphs and M^2 may not correspond to an unweighted multigraph when G is irregular. (When G is regular, the graph G^2 consisting of paths of length 2 in G does have normalized Laplacian $I - M^2$.)

For this reason we begin the algorithm by constructing an unweighted multigraph G_0 whose normalized Laplacian $I - M_0$ approximates $I - M^2$ and where M_0 is PSD. We can then approximate any power $I - M_0^{r'}$ using the square and plus one operation and hence can approximate $I - M^r$ for any even r (see Lemma 23). For odd powers, we again can finish with a single plus one operation.

Our main algorithm is presented below. Our input is an undirected two-way labeled multigraph G with normalized Laplacian $I - M$, $\epsilon \in (0, 1)$, and $r = b_z b_{z-1} \dots b_1 b_0$.

Algorithm 1 Computing a spectral approximation to the r-step random walk.

Input: G with normalized Laplacian $I - M$, $\epsilon \in (0, 1)$, $r = b_z b_{z-1} \dots b_1 b_0$
Output: G_z with normalized Laplacian $I - M_z$ such that $I - M_z \approx_\epsilon I - M^r$

1. Set $\mu = \epsilon/(32 \cdot z)$
2. Let $\mathcal{H}$ be family of expanders of every size such that $\lambda(H) \leq \mu$ for all $H \in \mathcal{H}$.
3. Construct G_0 such that $I - M_0 \approx_{\epsilon/(16 \cdot z)} I - M^2$ and M_0 is PSD.
4. For i in $\{1, \dots, z-1\}$
   a. If $b_{z-i} = 0$, $G_i = G_{i-1} \textcircled{\wp}_{\mathcal{H}} G_{i-1}$
   b. Else $G_i = (G_{i-1} \textcircled{\wp}_{\mathcal{H}} G_{i-1}) \textcircled{\wp}_{\mathcal{H}} G_0$
5. If $b_0 = 0$ (r even), $G_z = G_{z-1}$
6. Else (r is odd), $G_z = G_{z-1} \textcircled{\wp}_{\mathcal{H}} G$
7. Output G_z

Note that each derandomized product multiplies every vertex degree by a factor of $2 \cdot c$. So the degrees of $G, G_0, \ldots, G_z$ are all proportional to one another and the derandomized products in Algorithm 1 are well-defined.

4.2 Proof of Main Result

In this section we prove Theorem 4 by showing that Algorithm 1 yields a spectral approximation of our target power $I - M^r$ and can be implemented space-efficiently. First we show that our two operations, square and plus one, preserve spectral approximation.

▶ **Lemma 21** (Adapted from [5]). *Let N and $\tilde{N}$ be symmetric matrices such that $I - \tilde{N} \approx_\epsilon I - N$ and N is PSD, then $I - \tilde{N}^2 \approx_\epsilon I - N^2$.*

The proof of Lemma 21 can be found in [5] as well as the full version of this paper. Next we show that the plus one operation in our algorithm also preserves spectral approximation.

▶ **Lemma 22.** *Let $\tilde{N}$, N_1, and N_2 be symmetric matrices with spectral norm at most 1 and suppose that N_1 is PSD and commutes with N_2. If $I - \tilde{N} \approx_\epsilon I - N_1$ then*

$$I - \frac{1}{2} \cdot (\tilde{N} N_2 + N_2 \tilde{N}) \approx_\epsilon I - N_2 N_1.$$

A proof of Lemma 22 can be found in the full version of the paper.

Setting $N_1 = M^k$ and $N_2 = M$ in Lemma 22 shows that the plus one operation preserves spectral approximation whenever M^k is PSD. Recall that the first step in Algorithm 1 is to construct a graph G_0 with normalized Laplacian $I - M_0$ such that M_0 is PSD and $I - M_0$ approximates $I - M^2$. We can then approximate $I - M_0^k$ for any k using squaring and plus one because M_0^k will always be PSD. The following Lemma says that $I - M_0^k$ spectrally approximates $I - M^{2k}$.

▶ **Lemma 23.** *Let r be a positive integer with bit length $\ell(r)$ and A and B be symmetric PSD matrices with $\|A\|, \|B\| \leq 1$ such that $I - A \approx_\epsilon I - B$ and $I - B \approx_\epsilon I - A$ for $\epsilon \leq 1/(2 \cdot \ell(r))$. Then $I - A^r \approx_{2 \cdot \epsilon \cdot \ell(r)} I - B^r$.*

A proof of Lemma 23 can be found in the full version of the paper.

Now we can prove Theorem 4. We prove the theorem with three lemmas: Lemma 24 shows how to construct the graph G_0 needed in Algorithm 1, Lemma 25 argues that the algorithm produces a spectral approximation to $I - M^r$, and Lemma 26 shows that the algorithm can be implemented in space $O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)$.

4.2.1 Building G_0

▶ **Lemma 24.** *There is an algorithm that takes an undirected, unweighted multigraph G with normalized Laplacian $I - M$ and a parameter $\epsilon > 0$, and outputs a rotation map Rot_{G_0} for an undirected, unweighted multigraph G_0 with a two-way labeling and normalized Laplacian $I - M_0$ such that:*

1. *M_0 is PSD,*
2. *$I - M_0 \approx_\epsilon I - M^2$,*
3. *The algorithm uses space $O(\log N + \log(1/\epsilon))$, where N is the bit length of the input graph G.*

A proof of Lemma 24 can be found in Appendix C.

4.2.2 Proof of Spectral Approximation

▶ **Lemma 25.** *Let G be an undirected multigraph with normalized Laplacian $I - M$, r be a positive integer and $\epsilon \in (0, 1)$. Let G_z be the output of Algorithm 1 with normalized Laplacian $I - M_z$. Then*

$$I - M_z \approx_\epsilon I - M^r$$

Proof. Let $b_z b_{z-1} \dots b_1 b_0$ be the binary representation of r. Recall that for the derandomized products in our algorithm we use a family of c-regular expanders $\mathcal{H}$ from Theorem 20 such that for every $H \in \mathcal{H}$, $\lambda(H) \leq \mu = \epsilon/(32 \cdot z)$ (and hence $c = \text{poly}(1/\mu) = \text{poly}((\log r)/\epsilon)$).

We construct G_0 with normalized Laplacian $I - M_0$ as in Lemma 24 such that M_0 is PSD and $I - M_0 \approx_{\epsilon/(16 \cdot z)} I - M^2$. By Proposition 6 Part 1, and the fact that

$$\begin{aligned} \frac{\epsilon/(16 \cdot z)}{1 - \epsilon/(16 \cdot z)} &= \frac{\epsilon}{(16 \cdot z) - \epsilon} \\ &\leq \frac{\epsilon}{8 \cdot z}, \end{aligned}$$

we also have $I - M^2 \approx_{\epsilon/(8 \cdot z)} I - M_0$.

For each $i \in \{0, \dots z\}$ let r_i be the integer with binary representation $b_z b_{z-1} \dots b_{z-i}$ and let $I - M_i$ be the normalized Laplacian of G_i. We will prove by induction on i that G_i is a $(4 \cdot \mu \cdot i)$-approximation to $I - M_0^{r_i}$. Thus, G_{z-1} is a $4 \cdot \mu \cdot (z-1) \leq \epsilon/8$-approximation to $I - M_0^{r_{z-1}}$.

The base case is trivial since $r_0 = 1$. For the induction step, suppose that $I - M_{i-1} \approx_{4 \cdot \mu \cdot (i-1)} I - M_0^{r_{z-i+1}}$. On iteration i, if $b_{z-i} = 0$, then $G_i = G_{i-1} \textcircled{\wr}_{\mathcal{H}} G_{i-1}$. So we have

$$\begin{aligned} I - M_i &\approx_\mu I - M_{i-1}^2 \\ &\approx_{4 \cdot \mu \cdot (i-1)} I - M_0^{2 \cdot r_{i-1}} \\ &= I - M_0^{r_i} \end{aligned}$$

where the first approximation uses Theorem 19 and the second uses Lemma 21. By Proposition 6 Part 2 this implies that $I - M_i$ approximates $I - M_0^{r_i}$ with approximation factor

$$\mu + 4 \cdot \mu \cdot (i-1) + 4 \cdot \mu^2 \cdot (i-1) \leq 4 \cdot \mu \cdot i$$

where we used the fact that $\mu < 1/(32 \cdot (i-1))$.

If $b_{z-i} = 1$, $G_i = (G_{i-1} \textcircled{\wr}_{\mathcal{H}} G_{i-1}) \textcircled{\wr}_{\mathcal{H}} G_0$. Let $I - M_{\text{ds}}$ be the normalized Laplacian of $G_{i-1} \textcircled{\wr}_{\mathcal{H}} G_{i-1}$. By the analysis above, $I - M_{\text{ds}}$ is a $(\mu + 4 \cdot \mu \cdot (i-1) + 4 \cdot \mu^2 \cdot (i-1))$-approximation of $I - M_0^{2 \cdot r_{i-1}}$. By Theorem 19 and Lemma 22 we have

$$\begin{aligned} I - M_i &\approx_\mu I - \frac{1}{2} \cdot (M_{\text{ds}} M_0 + M_0 M_{\text{ds}}) \\ &\approx_{\mu + 4 \cdot \mu \cdot (i-1) + 4 \cdot \mu^2 \cdot (i-1)} I - M_0^{2 \cdot r_{i-1}} M_0 \\ &= I - M_0^{r_i} \end{aligned}$$

Applying Proposition 6 Part 2 and noting that $\mu \leq 1/(32 \cdot (i-1))$ we get

$$I - M_i \approx_{4 \cdot \mu \cdot i} I - M_0^{r_i}.$$

So we conclude that $I - M_{z-1} \approx_{\epsilon/8} I - M_0^{r_{z-1}}$. Furthermore, by Lemma 23 we have

$$I - M_0^{r_{z-1}} \approx_{\epsilon/8} I - M^{2 \cdot r_{z-1}}.$$

By Proposition 6 Part 2, and the fact that $\epsilon \leq 1$, this gives

$$I - M_{z-1} \approx_{\epsilon/3} I - M^{2 \cdot r_{z-1}}$$

If $b_0 = 0$ then $2 \cdot r_{z-1} = r$ and we are done. If $b_0 = 1$ then we apply one more plus one operation using our original graph G to form $G_z = G_{z-1} \textcircled{+}_{\mathcal{H}} G$ such that

$$\begin{aligned} I - M_z &\approx_{\mu} I - \frac{1}{2} \cdot (M_{z-1}M + MM_{z-1}) \\ &\approx_{\epsilon/3} I - M^{2 \cdot r_{z-1}+1} \\ &= I - M^r. \end{aligned}$$

Applying Proposition 6 Part 2 then gives $I - M_z \approx_{\epsilon} I - M^r$. ◀

4.2.3 Analysis of Space Complexity

▶ **Lemma 26.** *Algorithm 1 can be implemented so that given an undirected multigraph G, a positive integer r, and $\epsilon \in (0,1)$, it computes its output G_z in space $O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)$, where N is the bit length of the input graph G.*

Proof. We show how to compute Rot_{G_z} in space $O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)$. Let $b_z b_{z-1} \dots b_0$ be the binary representation of r. Following Algorithm 1, G_0 is constructed with normalized Laplacian $I - M_0 \approx_{\epsilon/(16 \cdot z)} I - M^2$. From Lemma 24, we know Rot_{G_0} can be computed in space $O(\log N + \log(16 \cdot z/\epsilon)) = O(\log N + \log(1/\epsilon) + \log\log r)$. Let $d_1, \dots, d_n$ be the vertex degrees in G_0 and $d_{\max}$ be the maximum degree.

The algorithm is presented to have z iterations, where on iteration $i \in [z-1]$, if $b_{z-i} = 0$ the derandomized product is invoked once, and if $b_{z-i} = 1$, it is invoked twice. On iteration z it is either invoked once ($b_0 = 1$) or not at all ($b_0 = 0$). It will be simpler for us to think of each derandomized product happening in its own iteration. So we will consider $\tau = z + w = O(\log r)$ iterations where w is the number of ones in $b_{z-1}, \dots, b_0$. On iterations $1, \dots, z-1$, there are $z-1$ derandomized square operations and w plus one operations. The final iteration will either have a plus one operation with the graph G (if $b_0 = 1$) or no operation.

We copy the bits of r into memory and expand them into τ bits as follows: for $i \in \{1, \dots z-1\}$ if $b_{z-i} = 0$, record a 0 (corresponding to a derandomized square) and if $b_{z-i} = 1$, record a 0 followed by a 1 (corresponding to a derandomized square followed by a plus one operation). Finish by just recording b_z at the end. Now we have τ bits $t_1, \dots, t_\tau$ in memory where for $i < \tau$, $t_i = 0$ if the ith derandomized product in our algorithm is a derandomized square and $t_i = 1$ if the ith derandomized product is a plus one with the graph G_0. If $t_\tau = 0$, we do no derandomized product on the last iteration and if $t_\tau = 1$ we apply the plus one operation using G instead of G_0 as described in the algorithm.

We also re-number our graphs to be $G_1, \dots, G_\tau$ where G_i is the graph produced by following the derandomized products corresponding to $t_1, \dots, t_i$. For each $i \in [\tau]$ and $v \in [n]$, vertex v in graph G_i has degree $(2 \cdot c)^i \cdot d_v$ because each derandomized product multiplies every vertex degree by a factor of $2 \cdot c$.

Since our graphs can be irregular, the input to a rotation map may have a different length than its output. To simplify the space complexity analysis, when calling a rotation map, we will pad the edge labels to always have the same length as inputs and outputs to the rotation map. For each graph G_i, we pad its edge labels to have length $\ell_i = \lceil \log_2 d_{\max} \rceil + i \cdot \lceil \log_2(2 \cdot c) \rceil$.

Sublogarithmic-space complexity can depend on the model, so we will be explicit about the model we use. We compute the rotation map of each graph G_i on a multi-tape Turing machine with the following input/output conventions:

- Input Description:
  - Tape 1 (read-only): Contains the input G, r, and ϵ with the head at the leftmost position of the tape.
  - Tape 2 (read-write): Contains the input to the rotation map (v_0, k_0), where $v_0 \in [n]$ is a vertex of G_i, and k_0 is the label of an edge incident to v_0 padded to have total length ℓ_i. The tapehead is at the rightmost end of k_0. The rest of the tape may contain additional data.
  - Tape 3: (read-write) Contains the bits $t_1, \ldots, t_\tau$ with the head pointing at t_i.
  - Tapes 4+: (read-write): Blank worktapes with the head at the leftmost position.
- Output Description:
  - Tape 1: The head should be returned to the leftmost position.
  - Tape 2: In place of (v_0, k_0), it should contain $(v_2, k_2) = \mathrm{Rot}_{G_i}(v_0, k_0)$, where $v_2 \in [n]$, and k_2 is padded to have total length ℓ_i. The head should be at the rightmost position of k_2 and the rest of the tape should remain unchanged from its state at the beginning of the computation.
  - Tape 3: Contains the bits $t_1, \ldots, t_\tau$ with the head pointing at t_i.
  - Tapes 4+: (read-write): Are returned to the blank state with the heads at the leftmost position.

Let $\mathrm{Space}(G_i)$ be the space used on tapes other than tape 1 to compute Rot_{G_i}. We will show that $\mathrm{Space}(G_i) = \mathrm{Space}(G_{i-1}) + O(\log c)$. Recalling that $\mathrm{Space}(G_0) = O(\log N + \log(1/\epsilon) + \log\log r)$ and unraveling the recursion gives

$$\begin{aligned}\mathrm{Space}(G_z) &= O(\log N + \log(1/\epsilon) + \log\log r + \tau \cdot \log c)\\ &= O(\log N + \log(1/\epsilon) + \log\log r + \log r \cdot \log(\mathrm{poly}(\log r)/\epsilon))\\ &= O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)\end{aligned}$$

as desired. Now we prove the recurrence on $\mathrm{Space}(G_i)$. We begin with (v_0, k_0) on Tape 2 (possibly with additional data) and the tapehead at the far right of k_0. We parse k_0 into $k_0 = (j_0, a_0, b)$ where j_0 is an edge label in $[(2\cdot c)^{i-1} \cdot d_{v_0}]$ padded to have length ℓ_{i-1}, $a_0 \in [c]$, and $b \in \{0, 1\}$.

Note that $G_i = G_{i-1} \textcircled{r}_{\mathcal{H}} G'$ where for $i \neq \tau$, we have $G' = G_{i-1}$ if $t_{i-1} = 0$ and $G' = G_0$ when $t_{i-1} = 1$. We compute Rot_{G_i} according to Definition 18. We move the head left to the rightmost position of j_0. If $b = 0$, we move the third tapehead to t_{i-1} and recursively compute $\mathrm{Rot}_{G_{i-1}}(v_0, j_0)$ so that Tape 2 now contains (v_1, j_1, a_0, b) (with j_1 padded to have the same length as j_0). The vertex v_1 in the graph G_{i-1} has degree $d' = (2 \cdot c)^{i-1} \cdot d_{v_1}$ so we next compute $\mathrm{Rot}_{H_{d'}}(j_1, a_0)$ so that (v_1, j_2, a_1, b) is on the tape. Finally we compute $\mathrm{Rot}_{G'}(v_1, j_2)$ and flip b to finish with $(v_2, j_3, a_1, \bar{b})$ on the second tape. We then move the third tapehead to t_i. If $b = 1$ then we just swap the roles of G_{i-1} and G' above.

So computing Rot_{G_i} involves computing the rotation maps of G_{i-1}, $H_{d'}$, and G' each once. Note that each of the rotation map evaluations occur in succession and can therefore reuse the same space. Clearly $\mathrm{Space}(G') \leq \mathrm{Space}(G_{i-1})$ because either $G' = G_{i-1}$ or G' is either G_0 or G, both of whose rotation maps are subroutines in computing $\mathrm{Rot}_{G_{i-1}}$. Computing $\mathrm{Rot}_{H_{d'}}$ adds an overhead of at most $O(\log c)$ space to store the additional edge label a_0 and the bit b. So we can compute the rotation map of G_τ in space $O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)$. ◀

5 Corollaries

5.1 Random Walks

Our algorithm immediately implies Theorem 2, which we prove below.

Proof of Theorem 2. Let D be the diagonal degree matrix and $I - M$ be the normalized Laplacian of G. Let $v = D^{1/2}e_S$ where e_S is the characteristic vector of the set S. Let d_S be the sum of the degrees of vertices in S. Then using the fact that $I - M^r = D^{-1/2}(I - T^r)D^{1/2}$ where T is the transition matrix of G gives:

$$\begin{aligned}\frac{1}{d_S} \cdot v^T(I - M^r)v &= \frac{1}{d_S} \cdot e_S^T D^{1/2} D^{-1/2}(I - T^r)D^{1/2} D^{1/2} e_S \\ &= \frac{1}{d_S} \cdot e_S^T D e_S - e_S^T(T^r(De_S/d_S)) \\ &= 1 - \Pr[V_r \in S | V_0 \in S] \\ &= \Phi_r(S)\end{aligned}$$

where the penultimate equality follows from the fact that De_S/d_S is the probability distribution over vertices in S where each vertex has mass proportional to its degree, i.e. the probability distribution $V_0 \| (V_0 \in S)$. Multiplying this distribution by T^r gives the distribution of $V_r \| (V_0 \in S)$. Multiplying this resulting distribution on the left by e_S^T, sums up the probabilities over vertices in S, which gives the probability that our random walk ends in S.

From Theorem 4, we can compute a matrix $\tilde{L}$ such that $\tilde{L} \approx_\epsilon I - M^r$ in space $O(\log N + (\log r) \cdot \log(1/\epsilon) + (\log r) \cdot \log\log r)$. It follows from Proposition 6, Part 6 and the definition of spectral approximation that

$$(1 - \epsilon) \cdot \Phi_r(S) \leq \frac{1}{d_S} \cdot v^T \tilde{L} v \leq (1 + \epsilon) \cdot \Phi_r(S). \qquad \blacktriangleleft$$

5.2 Odd Length Walks in Nearly Linear Time

Our approach to approximating odd length walks deterministically and space-efficiently also leads to a new result in the context of nearly linear-time (randomized) spectral sparsification algorithms. Specifically, we extend the following Theorem of Cheng, Cheng, Liu, Peng, and Teng [5].

▶ **Theorem 27** ([5]). *There is a randomized algorithm that given an undirected weighted graph G with n vertices, m edges, and normalized Laplacian $I - M$,* even *integer r, and $\epsilon > 0$ constructs an undirected weighted graph $\tilde{G}$ with normalized Laplacian $\tilde{L}$ containing $O(n \log n/\epsilon^2)$ non-zero entries, in time $O(m \cdot \log^3 n \cdot \log^5 r/\epsilon^4)$, such that $\tilde{L} \approx_\epsilon I - M^r$ with high probability.*

Our approach to approximating odd length walks can be used to extend Theorem 27 to odd r.

▶ **Corollary 28.** *There is a randomized algorithm that given an undirected weighted graph G with n vertices, m edges, and normalized Laplacian $I - M$,* odd *integer r, and $\epsilon > 0$ constructs an undirected weighted graph $\tilde{G}$ with normalized Laplacian $\tilde{L}$ containing $O(n \log n/\epsilon^2)$ non-zero entries, in time $O(m \cdot \log^3 n \cdot \log^5 r/\epsilon^4)$, such that $\tilde{L} \approx_\epsilon I - M^r$ with high probability.*

Our proof of Corollary 28 uses Theorem 27 as a black box. So in fact, given G with normalized Laplacian $I - M$ and any graph $\tilde{G}$ whose normalized Laplacian approximates $I - M^r$ for even r, we can produce an approximation to $I - M^{r+1}$ in time nearly linear in

the sparsities of G and $\tilde{G}$. To prove the corollary, we use the same method used in [18] and [6] for sparsifying two-step walks on undirected and directed graphs, respectively. The idea is that the graphs constructed from two-step walks can be decomposed into the union of *product graphs*: graphs whose adjacency matrices have the form xy^T for vectors $x, y \in \mathbb{R}^n$. We use the following fact from [6] that says that product graphs can be sparsified in time that is nearly-linear in the number of non-zero entries of x and y rather than the number of non-zero entries in xy^T, which may be much larger.

▶ **Lemma 29** (Adapted from [6] Lemma 3.18)**.** *Let x, y be non-negative vectors with $\|x\|_1 = \|y\|_1 = r$ and let $\epsilon \in (0,1)$. Furthermore, let s denote the total number of non-zero entries in x and y and let $L = \mathrm{diag}(y) - \frac{1}{r} \cdot xy^T$. Then there is an algorithm that in time $O(s \cdot \log s/\epsilon^2)$ computes a matrix $\tilde{L}$ with $O(s \cdot \log s/\epsilon^2)$ non-zeros such that $\tilde{L}$ is a directed ϵ-approximation of L with high probability.*

After using Lemma 29 to sparsify each product graph in our decomposition, we then apply an additional round of graph sparsification.

▶ **Lemma 30** ([12])**.** *Given an undirected graph G with n vertices, m edges, and Laplacian L and $\epsilon > 0$, there is an algorithm that computes a graph $\tilde{G}$ with Laplacian $\tilde{L}$ containing $O(n \cdot \log n/\epsilon^2)$ non-zero entries in time $O(m \cdot \log^2 n/\epsilon^2)$ such that $\tilde{L} \approx_\epsilon L$ with high probability.*

Now we are able to prove Corollary 28. See Appendex D for the proof.

References

1 Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *20th Annual Symposium on Foundations of Computer Science (San Juan, Puerto Rico, 1979)*, pages 218–223. IEEE, New York, 1979.

2 Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992. See also addendum in issue 4(1), 1993, pp. 199–120. doi:10.1002/rsa.3240030308.

3 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom Generators for Regular Branching Programs. In *FOCS*, pages 40–47. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.11.

4 Joshua Brody and Elad Verbin. The Coin Problem and Pseudorandomness for Branching Programs. In *FOCS*, pages 30–39. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.10.

5 Dehua Cheng, Yu Cheng, Yan Liu, Richard Peng, and Shang-Hua Teng. Spectral sparsification of random-walk matrix polynomials. *arXiv preprint*, 2015. arXiv:1502.03496.

6 Michael B Cohen, Jonathan Kelner, John Peebles, Richard Peng, Anup Rao, Aaron Sidford, and Adrian Vladu. Almost-Linear-Time Algorithms for Markov Chains and New Spectral Primitives for Directed Graphs. *arXiv preprint*, 2016. arXiv:1611.00755.

7 Michael B Cohen, Jonathan Kelner, John Peebles, Richard Peng, Aaron Sidford, and Adrian Vladu. Faster algorithms for computing the stationary distribution, simulating random walks, and more. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 583–592. IEEE, 2016.

8 Anindya De. Pseudorandomness for Permutation and Regular Branching Programs. In *IEEE Conference on Computational Complexity*, pages 221–231. IEEE Computer Society, 2011. doi:10.1109/CCC.2011.23.

9 Ofer Gabber and Zvi Galil. Explicit Constructions of Linear-Sized Superconcentrators. *J. Comput. Syst. Sci.*, 22(3):407–420, 1981. doi:10.1016/0022-0000(81)90040-4.

10 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for Network Algorithms. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 356–364, Montréal, Québec, Canada, 1994.

11 Gorav Jindal, Pavel Kolev, Richard Peng, and Saurabh Sawlani. Density Independent Algorithms for Sparsifying k-Step Random Walks. In Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017)*, volume 81 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 14:1–14:17, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.14.

12 Rasmus Kyng, Jakub Pachocki, Richard Peng, and Sushant Sachdeva. A Framework for Analyzing Resparsification Algorithms. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, volume abs/1611.06940. ACM, 2016. arXiv:1611.06940.

13 Yin Tat Lee, Richard Peng, and Daniel A. Spielman. Sparsified Cholesky Solvers for SDD linear systems. *CoRR*, abs/1506.08204, 2015. arXiv:1506.08204.

14 G. A. Margulis. Explicit constructions of expanders. *Problemy Peredači Informacii*, 9(4):71–80, 1973.

15 Milena Mihail. Conductance and Convergence of Markov Chains-A Combinatorial Treatment of Expanders. In *30th Annual Symposium on Foundations of Computer Science (Research Triangle Park, North Carolina)*, pages 526–531. IEEE, 1989.

16 Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil P. Vadhan. Derandomization Beyond Connectivity: Undirected Laplacian Systems in Nearly Logarithmic Space. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 801–812, 2017. doi:10.1109/FOCS.2017.79.

17 Joseph Naor and Moni Naor. Small-Bias Probability Spaces: Efficient Constructions and Applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

18 Richard Peng and Daniel A. Spielman. An Efficient Parallel Solver for SDD Linear Systems. *STOC*, 2014.

19 Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM*, 55(4):Art. 17, 24, 2008.

20 Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy Waves, the Zig-Zag Graph Product, and New Constant-Degree Expanders. *Annals of Mathematics*, 155(1), January 2001.

21 Eyal Rozenman and Salil Vadhan. Derandomized Squaring of Graphs. In *Proceedings of the 8th International Workshop on Randomization and Computation (RANDOM '05)*, number 3624 in Lecture Notes in Computer Science, pages 436–447, Berkeley, CA, August 2005. Springer.

22 Michael Saks and Shiyu Zhou. $\mathrm{BP_HSPACE}(S) \subseteq \mathrm{DSPACE}(S^{3/2})$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999.

23 Daniel A Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 81–90. ACM, 2004.

24 Thomas Steinke. Pseudorandomness for Permutation Branching Programs Without the Group Theory. Technical Report TR12-083, Electronic Colloquium on Computational Complexity (ECCC), July 2012. URL: http://eccc.hpi-web.de/report/2012/083/.

A Proof of Theorem 19

Proof. Note that $k \cdot G_1$ has the same transition matrix and normalized Laplacian as G_1. So we can replace G_1 with $k \cdot G_1$ and assume $k = 1$ without loss of generality.

Since G_0 and G_1 have the same vertex degrees, we can we write

$$I - \frac{1}{2} \cdot (M_0 M_1 + M_1 M_0) = I - D^{-1/2} \frac{1}{2} \cdot (T_0 T_1 + T_1 T_0) D^{1/2} \tag{1}$$

where T_0 and T_1 are the transition matrices of G_0 and G_1, respectively.

Following the proofs in [21] and [16], we can write the transition matrix for the random walk on $\tilde{G}$ as $\tilde{T} = \frac{1}{2} \cdot (PR_0\tilde{B}R_1Q + PR_1\tilde{B}R_0Q)$, where each matrix corresponds to a step in the definition of the derandomized product. The two terms correspond to $b = 0$ and $b = 1$ in the derandomized product and, setting $\bar{d} = \sum_{i \in [n]} d_i$,

- Q is a $\bar{d} \times n$ matrix that "lifts" a probability distribution over $[n]$ to one over $[\bar{d}]$ where the mass on each coordinate $i \in [n]$ is divided uniformly over the corresponding degree d_i. That is, $Q_{(u,i),v} = 1/d_i$ if $u = v$ and 0 otherwise where the rows of Q are ordered $(1,1), (1,2), \ldots, (1,d_1), (2,1), \ldots, (2,d_2)$, $\ldots (n,1), \ldots, (n,d_n)$.
- R_0 and R_1 are the $\bar{d} \times \bar{d}$ symmetric permutation matrices corresponding to the rotation maps of G_0 and G_1, respectively. That is, entry $(u,i), (v,j)$ in R_a is 1 if $\mathrm{Rot}_{G_a}(u,i) = (v,j)$ and 0 otherwise for $a \in \{0,1\}$.
- $\tilde{B}$ is a $\bar{d} \times \bar{d}$ symmetric block-diagonal matrix with n blocks where block i is the transition matrix for the random walk on $H_{d_i} \in \mathcal{H}$, the expander in our family with d_i vertices.
- $P = DQ^T$ is the $n \times \bar{d}$ matrix that maps any $\bar{d}$-vector to an n-vector by summing all the entries corresponding to edges incident to the same vertex in G_0 and G_1. This corresponds to projecting a distribution on $[\bar{d}]$ back down to a distribution over $[n]$. $P_{v,(u,i)} = 1$ if $u = v$ and 0 otherwise where the columns of P are ordered $(1,1), (1,2), \ldots, (1,d_1), (2,1), \ldots, (2,d_2), \ldots (n,1), \ldots, (n,d_n)$.

Likewise, we can write

$$(T_0 T_1 + T_1 T_0) = (PR_0\tilde{J}R_1Q + PR_1\tilde{J}R_0Q) \tag{2}$$

where $\tilde{J}$ is a $\bar{d} \times \bar{d}$ symmetric block-diagonal matrix with n blocks where block i is J_i, the transition matrix for the complete graph on d_i vertices with a self loop on every vertex. That is, every entry of J_i is $1/d_i$.

We will show that

$$I_{\bar{d}} - \frac{1}{2} \cdot (R_0\bar{B}R_1 + R_1\bar{B}R_0) \approx_\lambda I_{\bar{d}} - \frac{1}{2} \cdot (R_0\bar{J}R_1 + R_1\bar{J}R_0).$$

From this the theorem follows by multiplying by $D^{-1/2}P$ on the left and $(D^{-1/2}P)^T = QD^{1/2}$ on the right and applying Proposition 6 Part 3. Since $D^{-1/2}PQD^{1/2} = I_n$, the left-hand side becomes

$$\begin{aligned} I_n - D^{-1/2}\tilde{T}D^{1/2} &= I_n - \tilde{D}^{-1/2}\tilde{T}\tilde{D}^{1/2} \\ &= I_n - \tilde{M} \end{aligned}$$

where $\tilde{D} = 2 \cdot c \cdot D$ is the diagonal matrix of vertex degrees of $\tilde{G}$. By Equations (1) and (2), the right-hand side becomes $I_n - \frac{1}{2}(M_0 M_1 + M_1 M_0)$.

By Lemma 8, each graph in $\mathcal{H}$ is a λ-approximation of the complete graph on the same number of vertices. It follows that $I_{\bar{d}} - \tilde{B} \approx_\lambda I_{\bar{d}} - \tilde{J}$ because the quadratic form of a block diagonal matrix equals the sum of the quadratic forms of its blocks. By Lemma 13 and the fact that $I_d - \tilde{J}$ is PSD, $I_{\bar{d}} - \tilde{B}$ is also a directed λ-approximation of $I_{\bar{d}} - \tilde{J}$. So for all vectors $x, y \in \mathbb{R}^{\bar{d}}$ we have

$$\begin{aligned}\left|x^T(\tilde{B} - \tilde{J})y\right| &\leq \frac{\lambda}{2} \cdot (x^T(I_{\bar{d}} - \tilde{J})x + y^T(I_{\bar{d}} - \tilde{J})y) \\ &\leq \frac{\lambda}{2} \cdot (x^T x + y^T y - 2x^T \tilde{J} y).\end{aligned}$$

The first inequality uses Lemma 11. We can add the absolute values on the left-hand side since the right-hand side is always nonnegative ($I_d - \tilde{J}$ is PSD) and invariant to swapping x with $-x$. The second inequality follows from the fact that $\tilde{J}$ is PSD and so

$$0 \leq (x-y)^T \tilde{J}(x-y) = x^T \tilde{J} x + y^T \tilde{J} y - 2 \cdot x^T \tilde{J} y.$$

Fix $v \in \mathbb{R}^{\bar{d}}$ and set $x = R_0 v$ and $y = R_1 v$. Recall that R_0 and R_1 are symmetric permutation matrices and hence $R_0^2 = R_1^2 = I_{\bar{d}}$. Also note that for all square matrices A and vectors x, $x^T A x = x^T(A + A^T)x/2$. Combining these observations with the above gives

$$\begin{aligned}\left|v^T \left(\frac{1}{2} \cdot \left(R_0(\tilde{B} - \tilde{J})R_1 + R_1(\tilde{B} - \tilde{J})R_0\right)\right) v\right| &= \left|v^T R_0(\tilde{B} - \tilde{J})R_1 v\right| \\ &\leq \frac{\lambda}{2} \cdot (v^T R_0^2 v + v^T R_1^2 v - 2v^T R_0 \tilde{J} R_1 v) \\ &= \lambda \cdot (v^T v - v^T R_0 \tilde{J} R_1 v) \\ &= \lambda \cdot v^T \left(I - \frac{1}{2} \cdot \left(R_0 \tilde{J} R_1 + R_1 \tilde{J} R_0\right)\right) v\end{aligned}$$

Rearranging the above shows that

$$I_{\bar{d}} - \frac{1}{2} \cdot (R_0 \bar{B} R_1 + R_1 \bar{B} R_0) \approx_\lambda I_{\bar{d}} - \frac{1}{2} \cdot (R_0 \bar{J} R_1 + R_1 \bar{J} R_0),$$

which proves the theorem. ◀

B Proof of Theorem 20

Proof. Let H' be a c'-regular expander on m vertices such that $n \leq m \leq 2n$, c' is a constant independent of n and $\lambda(H') \leq \lambda' < 1/4$. H' can be constructed using already known strongly explicit constructions such as [9, 20] followed by squaring the graph a constant number of times to achieve $\lambda' < 1/4$. We will construct H as follows: Pair off the first $(m-n)$ vertices with the last $(m-n)$ vertices in H' and merge each pair into a single vertex (which will then have degree $2 \cdot c'$). To make the graph regular, add c' self loops to all of the unpaired vertices. More precisely, given $u' \in [n]$ and $i' \in [c] = [2 \cdot c']$ we compute $\mathrm{Rot}_H(u', i')$ as follows:

1. If $1 \leq u' \leq m - n$ [u' is a paired vertex]:
   a. If $1 \leq i' \leq c'$, let $u = u'$, $i = i'$ [u' is the first vertex in pair]
   b. else let $u = m - u'$, $i = i' - c'$ [u' is the second vertex in pair]
   c. let $(v, j) = \mathrm{Rot}_{H'}(u, i)$

2. else (if $m - n < u' \leq n$) [u' is an unpaired vertex]
   a. If $1 \leq i' \leq c'$, let $u = u'$, $i = i'$, and $(v, j) = \mathrm{Rot}_H(u, j)$ [original edge]
   b. else let $(v, j) = (u', i')$ [new self loop]
3. a. If $v \leq n$, let $(v', j') = (v, j)$
   b. else let $v' = m - v$, $j' = j + c'$.
4. Output (v', j')

Next we show that $\lambda(H)$ is bounded below 1 by a constant. The theorem then follows by taking the $O(\log 1/\lambda)$th power to drive $\lambda(H)$ below λ. This gives the graph degree poly$(1/\lambda)$.

Let A' be the adjacency matrix of H' and K' be the $m \times m$ all ones matrix. Since $\lambda(H') \leq \lambda'$, Lemma 8 implies that

$$\frac{1}{c'} \cdot (c' \cdot I - A') \approx_{\lambda'} \frac{1}{m} \cdot (m \cdot I - K').$$

Define B to be the $m \times n$ matrix such that $B_{u',u} = 1$ if and only if vertex $u' \in V(H')$ was merged into vertex $u \in V(H)$ or vertex $u \in V(H')$ was not merged and is labeled vertex u' in H. That is, $B_{u',u} = 1$ if and only if $u = u'$ or $n \leq u = m - u'$. Then the unnormalized Laplacian of the expander after the merging step is $B^T(c' \cdot I - A')B$. Adding self loops to a graph does not change its Laplacian. So applying Proposition 6 parts 3 and 6 we get

$$L(H) = \frac{1}{2c'} \cdot B^T(c' \cdot I - A')B \approx_{\lambda'} \frac{1}{2m} \cdot B^T(m \cdot I - K)B$$

Note that the righthand side is the normalized Laplacian of the graph U that results from starting with the complete graph on m vertices, merging the same pairs of vertices that are merged in H and adding m self loops to all of the unmerged vertices for regularity.

We finish the proof by showing that $\lambda(U) \leq 1/2$ and thus H is a $(\lambda' + 1/2 + \lambda'/2)$-approximation of the complete graph by Proposition 6 Part 2 and Lemma 8. Recalling that $\lambda' < 1/4$ completes the proof.

U has at least m edges between every pair of vertices so we can write its transition matrix T_u as

$$T_u = \frac{1}{2} \cdot J_m + \frac{1}{2} \cdot E$$

where J_m is the transition matrix of the complete graph on m vertices with self loops on every vertex and E is the transition matrix for an m-regular multigraph. Since the uniform distribution is stationary for all regular graphs, $\vec{1}$ is an eigenvector of eigenvalue 1 for T_u, J_m, and E. Thus

$$\begin{aligned}
\lambda(U) &= \sup_{v \perp \vec{1}} \frac{\|T_u v\|}{\|v\|} \\
&\leq \sup_{v \perp \vec{1}} \frac{\frac{1}{2} \cdot (\|J_m v\| + \|Ev\|)}{\|v\|} \\
&\leq \frac{1}{2} \cdot 0 + \frac{1}{2} \cdot 1,
\end{aligned}$$

which completes the proof. ◀

C Proof of Lemma 24

Proof. Let $\delta = 1/\lceil 4/\epsilon \rceil$ and $t = 1/\delta$, an integer. Let $\mathcal{H}$ be a family of c-regular expanders of every size from Theorem 20, such that for every $H \in \mathcal{H}$, $\lambda(H) \leq \delta$ (and hence $c = \text{poly}(1/\delta)$).

Let $\tilde{G} = G \textcircled{\scriptsize z}_{\mathcal{H}} G$ be the derandomized square of G with normalized Laplacian $I - \tilde{M}$. Each vertex v in $\tilde{G}$ has degree $\tilde{d}_v = 2 \cdot c \cdot d_v$, where d_v is the degree of v in G. We construct G_0 as follows: duplicate every edge of $\tilde{G}$ to have multiplicity t and then for each vertex v, add $\tilde{d}_v$ self loops. So for each vertex v in G_0, v has degree $(t+1) \cdot 2 \cdot c \cdot d_v$ and hence G_0 has the same stationary distribution as G. Note that we can write

$$M_0 = (t \cdot \tilde{M} + I)/(t+1).$$

First we show that M_0 is PSD. From Theorem 19, we have $I - \tilde{M} \approx_\delta I - M^2$, so $I - \tilde{M} \preceq (1+\delta) \cdot (I - M^2) \preceq (1+\delta) \cdot I$, since M^2 is PSD. Thus $\tilde{M} \succeq -\delta \cdot I$ and

$$M_0 \succeq \frac{t \cdot (-\delta \cdot I) + I}{t+1} \succeq 0.$$

Next we prove that $I - M_0 \approx_\epsilon I - M^2$

$$\begin{aligned} I - M_0 &= (t/(t+1)) \cdot (I - \tilde{M}) \\ &= \left(\frac{1}{1+\delta}\right) \cdot (I - \tilde{M}) \\ &\preceq I - M^2. \end{aligned}$$

Observe that since $I - \tilde{M} \approx_\delta I - M^2$, we also have

$$\begin{aligned} I - M_0 &= \left(\frac{1}{1+\delta}\right) \cdot (I - \tilde{M}) \\ &\succeq \left(\frac{1-\delta}{1+\delta}\right) \cdot (I - M^2) \\ &\succeq (1-\epsilon) \cdot (I - M^2). \end{aligned}$$

We can construct a two-way labeling of G in space $O(\log N)$ by arbitrarily numbering the edges incident to each vertex. Computing $\text{Rot}_{\tilde{G}}$ involves computing Rot_G twice and the rotation map of an expander in $\mathcal{H}$ once. For a given vertex degree d in G, Rot_{H_d} can be computed in space $O(\log(d \cdot c)) = O(\log N + \log(1/\epsilon))$. Duplicating the edges and adding self loops for Rot_{G_0} adds at most $O(\log N + \log(1/\epsilon))$ overhead for a total of $O(\log N + \log(1/\epsilon))$ space. ◀

D Proof of Corollary 28

Proof. Theorem 27 says that we can compute a graph $\tilde{G}$ with normalized Laplacian $I - \tilde{M}$ with $O(n \log n/\epsilon^2)$ non-zero entries, in time $O(m \cdot \log^3 n \cdot \log^5 r/\epsilon^4)$, such that $I - \tilde{M} \approx_{\epsilon/8} I - M^{r-1}$ with high probability. By Lemma 22 we have

$$I - \frac{1}{2} \cdot (\tilde{M}M + M\tilde{M}) \approx_{\epsilon/8} I - M^r. \tag{3}$$

Our goal is to sparsify the lefthand side. Note that since $I - \tilde{M}$ spectrally approximates $I - M^{r-1}$, the corresponding graphs must have the same stationary distribution and hence proportional vertex degrees. In other words there is a number k such that for all vertices

$v \in [n]$ we have $\deg_{\tilde{G}}(v) = k \cdot \deg_G(v)$. We will think of the graph that adds one step to our walk as $k \cdot G$ rather than G because $k \cdot G$ and $\tilde{G}$ have the same degrees and the normalized Laplacian of $k \cdot G$ is the same as the normalized Laplacian of G.

Let A and $\tilde{A}$ be the adjacency matrices of $k \cdot G$ and $\tilde{G}$, respectively and let D be the diagonal matrix of vertex degrees. Let $Q = D - AD^{-1}\tilde{A}$ and note that Q is the Laplacian of a weighted directed graph. We will show how to compute a sparse directed approximation of Q and use this to show how to compute a sparse approximation to the lefthand side of Equation 3. Our approach is inspired by similar arguments from [18, 6]. We decompose Q into n product graphs as follows. For each $i \in [n]$ let

$$Q_i = \operatorname{diag}(\tilde{A}_{i,:}) - \frac{1}{D_{i,i}} \cdot A_{:,i}\tilde{A}^T_{i,:}$$

where $\tilde{A}_{i,:}$ and $A_{:,i}$ denote the ith row of $\tilde{A}$ and the ith column of A, respectively. Observe that Q_i is a directed Laplacian of a bipartite graph between the neighbors of vertex i in $k \cdot G$ and the neighbors of i in $\tilde{G}$ and that $Q = \sum_{i\in[n]} Q_i$. Furthermore, each Q_i is a product graph and hence can be sparsified using Lemma 29. Set $x_i = A_{:,i}$, $y_i = \tilde{A}_{i,:}$, $r_i = D_{i,i}$, and let s_i be the total number of non-zero entries in x and y. Note that $\|x_i\|_1 = \|y_i\|_1 = r_i$ because $k \cdot G$ and $\tilde{G}$ have the same vertex degrees. By Lemma 29, for each $i \in [n]$ we can compute a directed $\epsilon/8$-approximation $\tilde{Q}_i$ of Q_i containing $O(s_i \cdot \log s_i/\epsilon^2)$ entries in time $O(s_i \cdot \log s_i/\epsilon^2)$. Applying the lemma to each Q_i yields $\tilde{Q} = \sum_{i\in[n]} \tilde{Q}_i$, which contains $O(m \cdot \log m/\epsilon^2)$ non-zero entries and can be computed in time $O(m \cdot \log m/\epsilon^2)$ because $\sum_{i\in[n]} s_i = O(m)$. By Lemma 12 we have

$$\frac{1}{2} \cdot (\tilde{Q}_i + \tilde{Q}_i^T) \approx_{\epsilon/8} \frac{1}{2} \cdot (Q_i + Q_i^T)$$

for all $i \in [n]$ with high probability. It follows from Proposition 6 Part 5 that

$$\begin{aligned}\frac{1}{2} \cdot (\tilde{Q} + \tilde{Q}^T) &= \frac{1}{2} \cdot \sum_{i\in[n]} (\tilde{Q}_i + \tilde{Q}_i^T) \\ &\approx_{\epsilon/8} \frac{1}{2} \cdot \sum_{i\in[n]} (Q_i + Q_i^T) \\ &= \frac{1}{2} \cdot (Q + Q^T)\end{aligned}$$

with high probability. From Proposition 6 Part 3, we then get

$$\begin{aligned}D^{-1/2}\frac{1}{2} \cdot (\tilde{Q} + \tilde{Q}^T)D^{-1/2} &\approx_{\epsilon/8} D^{-1/2}\frac{1}{2} \cdot (Q + Q^T)D^{-1/2} \\ &= I - \frac{1}{2} \cdot (\tilde{M}M + M\tilde{M})\end{aligned}$$

with high probability. Applying Lemma 30 we can re-sparsify the graph corresponding to $D^{-1/2}\frac{1}{2} \cdot (\tilde{Q} + \tilde{Q}^T)D^{-1/2}$ to produce a graph G' whose normalized Laplacian $I - M'$ has $O(n \cdot \log n/\epsilon^2)$ non-zero entries and $I - M' \approx_{\epsilon/8} D^{-1/2}\frac{1}{2} \cdot (\tilde{Q} + \tilde{Q}^T)D^{-1/2}$ with high probability. This takes additional time $O(m \cdot \log^2 n/\epsilon^2)$ due to Theorem 1.1 of [12]. Applying Proposition 6 Part 2 twice we get that $I - M' \approx_\epsilon I - M^r$ and the total running time for the procedure was $O(m \cdot \log^3 n \cdot \log^5 r/\epsilon^4)$. ◀

Two-Source Condensers with Low Error and Small Entropy Gap via Entropy-Resilient Functions

Avraham Ben-Aroya
The Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv, Israel

Gil Cohen
The Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv, Israel
https://www.gilcohen.org
gil@tauex.tau.ac.il

Dean Doron
Department of Computer Science, University of Texas at Austin, USA
deandoron@utexas.edu

Amnon Ta-Shma
The Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv, Israel
https://www.cs.tau.ac.il/~amnon
amnon@tau.ac.il

Abstract

In their seminal work, Chattopadhyay and Zuckerman (STOC'16) constructed a two-source extractor with error ε for n-bit sources having min-entropy polylog(n/ε). Unfortunately, the construction's running-time is poly(n/ε), which means that with polynomial-time constructions, only polynomially-small errors are possible. Our main result is a poly$(n, \log(1/\varepsilon))$-time computable two-source *condenser*. For any $k \geq$ polylog(n/ε), our condenser transforms two independent (n, k)-sources to a distribution over $m = k - O(\log(1/\varepsilon))$ bits that is ε-close to having min-entropy $m - o(\log(1/\varepsilon))$. Hence, achieving entropy gap of $o(\log(1/\varepsilon))$.

The bottleneck for obtaining low error in recent constructions of two-source extractors lies in the use of resilient functions. Informally, this is a function that receives input bits from r players with the property that the function's output has small bias even if a bounded number of corrupted players feed adversarial inputs after seeing the inputs of the other players. The drawback of using resilient functions is that the error cannot be smaller than $\ln r/r$. This, in return, forces the running time of the construction to be polynomial in $1/\varepsilon$.

A key component in our construction is a variant of resilient functions which we call *entropy-resilient functions*. This variant can be seen as playing the above game for several rounds, each round outputting one bit. The goal of the corrupted players is to reduce, with as high probability as they can, the min-entropy accumulated throughout the rounds. We show that while the bias decreases only polynomially with the number of players in a one-round game, their success probability decreases *exponentially* in the entropy gap they are attempting to incur in a repeated game.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Condensers, Extractors, Resilient functions, Explicit constructions

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.43

Category RANDOM

Funding *Avraham Ben-Aroya*: Israel Science Foundation Grant 994/14 and by Len Blavatnik and the Blavatnik Family Foundation.
Dean Doron: Israel Science Foundation Grant 994/14 and by Len Blavatnik and the Blavatnik Family Foundation. This work was done while being at Tel-Aviv University.
Amnon Ta-Shma: Israel Science Foundation Grant 994/14.



© Avraham Ben-Aroya, Gil Cohen, Dean Doron, and Amnon Ta-Shma;
licensed under Creative Commons License CC-BY

1 Introduction

The problem of extracting randomness from imperfect random sources can be traced back to von Neumann [45]. Ideally, and somewhat informally, a randomness extractor is an algorithm that produces, or extracts, truly random bits from an imperfect source of randomness. Going beyond that particular task, randomness extractors have found dozens of applications for error correcting codes, cryptography, combinatorics, and circuit lower bounds to name a few.

An imperfect source of randomness is modelled by a random variable X that, for convenience sake, is assumed to be supported on n-bit strings. The standard measure for the amount of randomness in X is its *min-entropy* [18], which is the maximum $k \geq 0$ for which one cannot guess X with probability larger than 2^{-k}. For any such k, we say that X is an (n, k)-*source*, or a k-*source* for short.

Ideally, a randomness extractor would have been defined as a function $\mathsf{Ext}\colon \{0,1\}^n \to \{0,1\}^m$ with the property that for every random variable X with sufficiently high min-entropy, the output $\mathsf{Ext}(X)$ is ε-close to the uniform distribution on $\{0,1\}^m$ in the statistical distance, which we write as $\mathsf{Ext}(X) \approx_\varepsilon U_m$. Unfortunately, such a function Ext, even for very high min-entropy $k = n - 1$ and, when set with a modest error guarantee $\varepsilon = 1/4$ and a single output bit $m = 1$, does not exist. In light of that, several types of randomness extractors, that relax in different ways the above ideal definition, have been introduced and studied in the literature. In this work, we focus on one such well-studied instantiation.

▶ **Definition 1** (Two-source extractors [18]). *A function* $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ *is a* two-source extractor for min-entropy k with error guarantee ε *if for every pair of independent* (n, k)*-sources* X, Y*, the output distribution* $\mathsf{Ext}(X, Y) \approx_\varepsilon U_m$.

The existence of a two-source extractor for any min-entropy $k = \Omega(\log(n/\varepsilon))$ with $m = 2k - O(\log(1/\varepsilon))$ output bits was proved in [18]. In the same paper, an explicit construction of a two-source extractor for min-entropy $k > n/2$ was obtained. Remarkably, despite much attention [13, 42, 12, 31] and progress on relaxed settings [6, 41, 33, 32, 34, 20], the problem of constructing two-source extractors even for min-entropy as high as $k = 0.1n$ with $m = 1$ output bits remained open for 30 years. To appreciate the difficulty of constructing two-source extractors, we remark that such constructions yield explicit constructions of Ramsey graphs, a notoriously hard problem in combinatorics [1, 38, 25, 19, 26, 3, 27, 39, 5, 6, 7, 23].

In their breakthrough result, Chattopadhyay and Zuckerman [17] were finally able to obtain an explicit two-source extractor for min-entropy $k = \mathrm{polylog}(n/\varepsilon)$. Partially motivated by the problem of constructing Ramsey graphs, a line of followup works [24, 16, 21, 9, 22, 35, 36] focused on the case of constant error ε and was devoted for reducing the min-entropy requirement as a function of n. The state of the art result in this line of work is due to Li [36] and requires min-entropy $\frac{\log n \cdot \log\log n}{\log\log\log n} \cdot \mathrm{poly}(1/\varepsilon)$.

1.1 Resilient Functions – The Barrier for Obtaining Extractors With Low Error

Unfortunately, despite the fact that the dependence of the min-entropy of the Chattopadhyay-Zuckerman extractor on ε is polynomially-close to optimal, the running-time of their construction depends polynomially on $1/\varepsilon$ rather than the desired $\mathrm{polylog}(1/\varepsilon)$ dependence. The same holds for all subsequent constructions. That is, these constructions are not strongly polynomial-time and, in particular, the error guarantee cannot be taken to be sub-polynomial in n while maintaining running-time $\mathrm{poly}(n)$. This stands in contrast to classical extractors for high min-entropy [18, 42, 13, 31] that are strongly polynomial-time, and can support exponentially-small error.

Informally speaking, the reason for this undesired dependence of the running-time on ε lies in the use of a so-called *resilient function* [11]. A q-resilient function $f\colon \{0,1\}^r \to \{0,1\}$ can be thought of as an r-player game. If all players feed uniform and independent inputs to f, the output distribution has small bias, and, furthermore, this property is retained even if any q players decide to deviate from the rules of the game and choose their inputs as a function of all other inputs to f.

Majority on r input bits is an example of a q-resilient function with $q = O(\sqrt{r})$. Ajtai and Linial proved, using the probabilistic method, the existence of a q-resilient function for $q = O(\frac{r}{\log^2 r})$ [2]. The KKL Theorem [30] implies that the Ajtai-Linial function is tight up to a $\log r$ factor. Chattopadhyay and Zuckerman [17] constructed a derandomized version of the Ajtai-Linial function with $q = r^{1-\delta}$, for any constant $\delta > 0$. Their construction has further desirable properties. In a subsequent work, Meka obtained a derandomized version of the Ajtai-Linial function with the same parameters as the randomized construction [37]. However, no matter what function is chosen, [30] showed that there is always a single corrupted player that has influence $p = \Omega(\frac{\log r}{r})$, i.e., with probability p over the input fed by the other players, the single corrupted player can fully determine the result.

Almost all constructions of randomness extractors following [17] can be divided into two steps. First, the two n-bit sources X, Y are "transformed" to a single r-bit source $Z = h(X,Y)$ with some structure, called a *non-oblivious bit-fixing source*. A resilient function $f\colon \{0,1\}^r \to \{0,1\}$ is then applied to Z so to obtain the output $\mathsf{Ext}(X,Y) = f(h(X,Y))$. In all works, the function h is based on non-malleable extractors or on related primitives such as correlation breakers. As mentioned above, the use of the resilient function implies that even a single corrupted player has influence $\Omega(\frac{\log r}{r})$ and so to obtain an error guarantee ε, the number of players r must be taken larger than $1/\varepsilon$. This results in running-time $\Omega(1/\varepsilon)$.[1]

1.2 Entropy-Resilient Functions

To obtain our condenser, we extend the notion of resilient functions to functions outputting many bits. Informally speaking, instead of considering an r-player game in which the bad players try to bias the output, we study a repeated game version in which the r players play for m rounds. The bad players attempt to decrease, with as high probability as they can, the min-entropy of the m-bit outcome (and we will even allow the bad players to cast their votes after the good players played all rounds).

Recall that, by [30], when $m = 1$, even a single player can bias the result by $\Omega(\frac{\log r}{r})$. Put differently, viewing this bias as the error of a deterministic extractor, the error is bound to be at least polynomially-small in the number of players. Our key insight is that when m becomes large, the probability that the bad players can reduce g bits of entropy from the output (creating an "entropy gap" of g) is *exponentially small* in g. We further show that this holds for a specific function f, induced by the Ajtai-Linial function, even when the honest players are only t-wise independent (for $t = \mathrm{polylog}(r/\varepsilon)$). Our analysis uses and extends ideas from the work of Chattopadhyay and Zuckerman [17].

[1] There is one exception to the above scheme. In [8], it is shown that if very strong t-non-malleable extractors can be explicitly constructed then the function f can be replaced with the parity function (which is not resilient at all) and low error two-source extractors with low min-entropy requirement can be obtained. However, it is not known how to explicitly construct such t-non-malleable extractors.

1.3 The Two-Source Condensers We Obtain

The main contribution of this work is an explicit construction of a two-source *condenser* with low error and small entropy gap, outputting almost all of the entropy from one source.

▶ **Definition 2** (Two-source condensers). *A function* $\mathsf{Cond}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ *is a* two-source condenser for min-entropy k with min-entropy gap g and error guarantee ε *if for every pair of independent* (n,k)*-sources,* $\mathsf{Cond}(X,Y)$ *is* ε*-close to an* $(m, m-g)$*-source.*

Note that a two-source extractor is a two-source condenser with entropy gap $g = 0$. Thus, condensers can be seen as a relaxation of extractors in which some, hopefully small, "gap" of min-entropy in the output distribution is allowed. Despite having a weaker guarantee, condensers play a key role in the construction of many types of randomness extractors, including two-source extractors [9], their variants [42, 6, 47, 41, 32], and seeded-extractors [28]. Most related to our work is a paper by Rao [40] that, for every $\delta > 0$, constructed a $\mathrm{poly}(n, \log(1/\varepsilon))$-time computable two-source condenser[2] for min-entropy $k = \delta n$ having $m = \Omega(\delta n)$ output bits with entropy gap $g = \mathrm{poly}(1/\delta, \log(1/\varepsilon))$.

In this work, we obtain a strongly polynomial-time construction of a two-source condenser with low error and small min-entropy gap.

▶ **Theorem 3** (Main result). *For all integers* n, k *and every* $\varepsilon > 0$ *such that* $n \geq k \geq \mathrm{polylog}(\frac{n}{\varepsilon})$*, there exists a* $\mathrm{poly}(n, \log(1/\varepsilon))$*-time computable two-source condenser*

$$\mathsf{Cond}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$$

for min-entropy k*, with error guarantee* ε*, min-entropy gap* $g = o(\log \frac{1}{\varepsilon})$*, and* $m = k - O(\log(1/\varepsilon))$ *output bits.*

Note that the entropy gap g is independent of the entropy k and scales *sub*-logarithmically with $1/\varepsilon$. We prove Theorem 3, whose formal statement is given in Theorem 31, in two steps. First, we construct a two-source condenser with the same guarantees as provided by Theorem 3, though only with $m = k^\alpha$ output bits, where $0 < \alpha < 1$ is some small universal constant (see Theorem 27). This part of the construction is based on our study of entropy-resilient functions (Section 3) and on the adaptation of the Chattopadhyay-Zuckerman construction for entropy-resilient functions. To reduce the huge entropy-loss we incur (i.e., to increase the output length from k^α to $k - O(\log(1/\varepsilon))$), in the second step, we construct a seedless condenser for block-sources–a result that we believe is of independent interest on which we now elaborate.

1.4 Seedless Condensers for a Single Block-Source

A (k_1, k_2)*-block-source* is a pair of random variables X_1, X_2 that, although may be dependent, have the following guarantee. First, X_1 is a k_1-source, and second, conditioned on any fixing of X_1, the random variable X_2 has min-entropy k_2. Throughout this section, we denote the length of X_1 by n_1 and the length of X_2 by n_2. Informally, the notion of a block-source "lies between" a single source and two independent sources. Indeed, any (k_1, k_2)-block-source is a $(k_1 + k_2)$-source. Moreover, if X_1 is a k_1-source and X_2 is an independent k_2-source then X_1, X_2 is a (k_1, k_2)-block-source.

[2] To the matter of fact, Rao entitled his construction a "two-source almost extractor" – a suitable name given its small entropy gap.

Block-sources are key to almost all constructions of seeded extractors as well as to the construction of Ramsey graphs. As mentioned above, there is no one-source extractor, whereas two-source extractors exist even for very low min-entropy. Despite being more structured than a general source, it is a well-known fact that there is no extractor for a single block-source (with non-trivial parameters).

A key component that allows us to increase the output length of our condenser discussed above is a seedless condenser for a single block-source. Let X_1, X_2 be a (k_1, k_2)-block-source. Write $g = n_2 - k_2$ for the entropy gap of X_2. For any given $\varepsilon > 0$, we show how to *deterministically* transform X_1, X_2 to a single m-bit random variable, where $m = k_1 - g - O(\log(1/\varepsilon))$, that is ε-close to having min-entropy $m - g - 1$. That is, informally, we are able to condense X_1 roughly to its entropy content k_1 using (the dependent random variable) X_2 while inheriting the entropy gap of X_2 both in the resulted entropy gap and entropy loss. We stress that this transformation is deterministic. This demonstrates that despite the well-known fact that a block-source extractor does not exist, a block-source condenser does. For a formal treatment, see Section 5.

1.5 A Three-Source Extractor

An immediate implication of Theorem 3 are low error three-source extractors supporting min-entropies $k_1 = k_2 = \text{polylog}(n/\varepsilon)$ and $k_3 = \Omega(\log(1/\varepsilon))$. This is achieved by feeding our condenser's output $Y = \mathsf{Cond}(X_1, X_2)$ as a seed to a seeded extractor that supports small entropies (see, e.g., Theorem 10), outputting $\mathsf{Ext}(X_3, Y)$.

▶ **Corollary 4.** *For all integers n, k, k' and every $\varepsilon > 0$ such that $n \geq k \geq \text{polylog}(\frac{n}{\varepsilon})$ and $n \geq k' \geq \Omega(\log \frac{1}{\varepsilon})$ there exists a $\text{poly}(n, \log(1/\varepsilon))$-time computable three-source extractor*

$$\mathsf{3Ext}\colon \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$$

for min-entropies k, k, k' and error guarantee ε, where $m = k' - O(\log(1/\varepsilon))$.

Proof. Set $\varepsilon' = \varepsilon^2$, let $k' \geq 2\log(1/\varepsilon') + O(1)$ and let $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be the (k', ε')-strong-seeded-extractor guaranteed to us by Theorem 10, where $m = k' - 2\log(1/\varepsilon) - O(1)$ and $d = O(\log n \log(n/\varepsilon'))$.

Let k be large enough for $\mathsf{Cond}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^d$ given to us by Theorem 31 to output d bits with error $\varepsilon/2$ and entropy gap $g = o(\log(1/\varepsilon))$, so indeed $k \geq \text{polylog}(\frac{n}{\varepsilon})$.

Denote $Y = \mathsf{Cond}(X_1, X_2)$, so Y is $(\varepsilon/2)$-close to some random variable Y' having min-entropy at least $d - g$. Then:

$$\begin{aligned}
|\mathsf{Ext}(X_3, Y) - U_m \times Y| &\leq |\mathsf{Ext}(X_3, Y') - U_m \times Y'| + \frac{\varepsilon}{2} \\
&= \sum_{y \in \mathsf{supp}(Y')} \Pr[Y' = y] \cdot |\mathsf{Ext}(X_3, y) - U_m| + \frac{\varepsilon}{2} \\
&\leq \sum_{y \in \{0,1\}^d} 2^{-(d-g)} \cdot |\mathsf{Ext}(X_3, y) - U_m| + \frac{\varepsilon}{2} \\
&\leq 2^g \sum_{y \in \{0,1\}^d} 2^{-d} |\mathsf{Ext}(X_3, y) - U_m| + \frac{\varepsilon}{2} \\
&= 2^g \cdot |\mathsf{Ext}(X_3, U_d) - U_m \times U_d| + \frac{\varepsilon}{2} \;\leq\; 2^g \varepsilon' + \frac{\varepsilon}{2} \;\leq\; \varepsilon,
\end{aligned}$$

thereby also showing that $\mathsf{3Ext}(X_1, X_2, X_3) = \mathsf{Ext}(X_3, Y)$ is strong in Y. ◀

When ε is sub-polynomial in n (which is an interesting regime of parameters because then the two-source extractor of [17] is not polynomial-time computable) Corollary 4 improves upon known three-source extractors that either require all three-sources to have min-entropy $\text{polylog}(\frac{n}{\varepsilon})$ [34] or require, for any parameter of choice $\delta > 0$, min-entropies δn, $\text{poly}(\frac{1}{\delta})\log(\frac{n}{\varepsilon})$, $\text{poly}(\frac{1}{\delta})\log(\frac{\log n}{\varepsilon})$ [20].

We remark that the proof of Corollary 4 goes through because the tiny entropy gap g of $Y = \mathsf{Cond}(X_1, X_2)$ (where $g = o(\log \frac{1}{\varepsilon})$) allows us to use Y as a replacement to a truly uniform seed with only a minor loss in parameters. We believe this should also be true in other circumstances where random variables with a negligible entropy gap can replace uniform random variables. A recent example to this is the use of samplers with multiplicative error instead of standard samplers in [9].

To conclude, we believe that the use of entropy-resilient functions as a tool to extract almost all the entropy from bit-fixing sources while suffering only a small error is both natural and interesting on its own. We hope the tools and constructions developed in this paper will be of further use, possibly for constructing low error two-source extractors. In particular, we have seen in Corollary 4 that by using an independent third source and outputting $\mathsf{Ext}(X_3, \mathsf{Cond}(X_1, X_2))$ we get an excellent three-source extractor. An open problem left by our work is whether outputting $\mathsf{Ext}(X_2, \mathsf{Cond}(X_1, X_2))$ gives a low-error two-source extractor. We remark that a similar idea has been used in previous constructions [34, 10] and elsewhere. We were not able to prove that $\mathsf{Ext}(X_2, \mathsf{Cond}(X_1, X_2))$ gives a low-error two-source extractor and we leave this as an intriguing open problem.

2 Preliminaries

We use $\log(x)$ for $\log_2(x)$. For an integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$. The density of a subset $B \subseteq A$ is denoted by $\mu(B) = \frac{|B|}{|A|}$.

2.1 Random Variables, Min-Entropy

The *statistical distance* between two distributions X and Y over the same domain Ω is defined by $\mathsf{SD}(X, Y) = \max_{A \subseteq \Omega}(\Pr[X \in A] - \Pr[Y \in A])$. If $\mathsf{SD}(X, Y) \leq \varepsilon$ we say X is ε-close to Y and denote it $X \approx_\varepsilon Y$. We denote by U_n the random variable distributed uniformly over $\{0,1\}^n$.

For a function $f\colon \Omega_1 \to \Omega_2$ and a random variable X distributed over Ω_1, $f(X)$ is the random variable distributed over Ω_2 obtained by choosing $x \sim X$ and outputting $f(x)$. For every $f\colon \Omega_1 \to \Omega_2$ and two random variables X, Y over Ω_1 it holds that $\mathsf{SD}(f(X), f(Y)) \leq \mathsf{SD}(X, Y)$.

The *min-entropy* of a random variable X is defined by

$$H_\infty(X) \;=\; \min_{x \in \mathsf{supp}(X)} \log \frac{1}{\Pr[X = x]}.$$

A random variable X is an (n, k)-source if X is distributed over $\{0,1\}^n$ and has min-entropy at least k. When n is clear from the context we sometimes omit it and simply say that X is a k-source.

2.2 Limited Independence

▶ **Definition 5.** *A distribution X over $\{0,1\}^n$ is called* (t, γ)-wise independent *if the restriction of X to every t coordinates is γ-close to U_t.*

▶ **Lemma 6** ([4])**.** *Let $X = X_1, \ldots, X_n$ be a distribution over $\{0,1\}^n$ that is (t, γ)-wise independent. Then, X is $(n^t\gamma)$-close to a t-wise independent distribution.*

2.3 Seeded Extractors

▶ **Definition 7** (Seeded extractors). *A function*

$$\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$$

is a (k,ε)-seeded-extractor if the following holds. For every (n,k)-source X, the output $\mathsf{Ext}(X,Y) \approx_\varepsilon U_m$, where Y is uniformly distributed over $\{0,1\}^d$ and is independent of X. Further, Ext is a (k,ε)-strong-seeded-extractor if $(\mathsf{Ext}(X,Y),Y) \approx_\varepsilon (U_m,Y)$.

▶ **Theorem 8** ([28]). *There exists a universal constant $c_{\mathsf{GUV}} \geq 2$ for which the following holds. For every integers $n \geq k$ and $\varepsilon > 0$ there exists a $\mathrm{poly}(n, \log(1/\varepsilon))$-time computable (k,ε)-strong-seeded-extractor*

$$\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$$

with seed length $d = c_{\mathsf{GUV}} \log(n/\varepsilon)$ and $m = k/2$ output bits.

Extractors can be used for sampling using weak sources.

▶ **Theorem 9** ([46]). *Let $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (k_1,ε)-seeded-extractor. Identify $\{0,1\}^d$ with $[2^d]$ and let $S(X) = \left\{\mathsf{Ext}(X,1),\ldots,\mathsf{Ext}(X,2^d)\right\}$. Then, for every (n,k_2)-source X and any set $T \subseteq \{0,1\}^m$,*

$$\Pr_{x\sim X}\left[\left|\frac{|S(x)\cap T|}{2^d} - \mu(T)\right| > \varepsilon\right] \;\leq\; 2^{-(k_2-k_1)}.$$

The following extractor allows us to extract all the min-entropy, at the cost of a larger seed-length.

▶ **Theorem 10** ([28]). *There exists a universal constant c such that the following holds. For all integers $n \geq k$ and any $\varepsilon > 0$ such that $k \geq 2\log(1/\varepsilon) + O(1)$, there exists a $\mathrm{poly}(n, \log(1/\varepsilon))$-time computable (k,ε)-strong-seeded-extractor*

$$\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$$

with seed length $d = c\log n \cdot \log\frac{n}{\varepsilon}$ and $m = k - 2\log\frac{1}{\varepsilon} - O(1)$ output bits.

2.4 Two-Source Condensers

▶ **Definition 11** (Condensers). *A function*

$$\mathsf{Cond}\colon \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \{0,1\}^m$$

is an $((n_1,k_1),(n_2,k_2)) \to_\varepsilon (m, k' = m-g)$ condenser *if the following holds. For every (n_1,k_1)-source X_1 and an independent (n_2,k_2)-source X_2, the output $\mathsf{Cond}(X_1,X_2)$ is ε-close to an (m,k')-source. We refer to ε as the* error guarantee *and to g as the* entropy gap *of* Cond.

▶ **Definition 12** (Strong condensers). *A function*

$$\mathsf{Cond}\colon \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \{0,1\}^m$$

is a $((n_1,k_1),(n_2,k_2)) \to_{\varepsilon_1,\varepsilon_2} (m,k')$-strong-condenser (in the first source) *if the following holds. For every (n_1,k_1)-source X_1 and an independent (n_2,k_2)-source X_2, with probability $1-\varepsilon_1$ over $x_1 \sim X_1$, the output $\mathsf{Cond}(x_1,X_2)$ is ε_2-close to an (m,k')-source.*

Similarly, one can define, in the natural way, a condenser that is strong in the second source.

2.5 Non-Malleable Extractors

▶ **Definition 13.** *A function* $\mathsf{nmExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a* (k,ε) t-non-malleable extractor*, if for every* (n,k)*-source* X*, for every independent random variable* Y *that is uniform over* $\{0,1\}^d$ *and every functions* $f_1,\ldots,f_t\colon \{0,1\}^d \to \{0,1\}^d$ *with no fixed-points*[3] *it holds that:*

$$(\mathsf{nmExt}(X,Y),\mathsf{nmExt}(X,f_1(Y)),\ldots,\mathsf{nmExt}(X,f_t(Y),Y) \approx_\varepsilon (U_m,\mathsf{nmExt}(X,f_1(Y)),\ldots,\mathsf{nmExt}(X,f_t(Y),Y)).$$

We will need the following lemma concerning the existence of a set of good seeds of a non-malleable extractor, given in [17].

▶ **Lemma 14** ([17])**.** *Let* $\mathsf{nmExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a* (k,ε) t*-non-malleable extractor. Let* X *be any* (n,k)*-source. Let* BAD *be the set defined by*

$$\mathrm{BAD} = \{r \in [D] \mid \exists \text{ distinct } r_1,\ldots,r_t \in [D], \forall i \in [t]\ r_i \neq r,\ |(\mathsf{nmExt}(X,r), \mathsf{nmExt}(X,r_1),\ldots,\mathsf{nmExt}(X,r_t)) - (U_m,\mathsf{nmExt}(X,r_1),\ldots,\mathsf{nmExt}(X,r_t))| > \sqrt{\varepsilon}\}.$$

Then, $\mu(\mathrm{BAD}) \leq \sqrt{\varepsilon}$*. We refer to the set* $[D] \setminus \mathrm{BAD}$ *as the set of* good seeds *(with respect to the underlying distribution of* X*).*

▶ **Lemma 15.** *Let* $X_1,\ldots,X_t$ *be random variables over* $\{0,1\}^m$*. Further suppose that for any* $i \in [t]$*,*

$$\left(X_i, \{X_j\}_{j\neq i}\right) \approx_\varepsilon \left(U_m, \{X_j\}_{j\neq i}\right).$$

Then, $(X_1,\ldots,X_t) \approx_{t\varepsilon} U_{tm}$*.*

Finally, good explicit constructions of t-non-malleable extractors exist. The following choice of parameters will be sufficient for us.

▶ **Theorem 16** ([15, 22, 35])**.** *There exists a universal constant* $c_{\mathsf{nm}} \geq 2$ *such that for all integers* n,k,t*, and every* $\varepsilon > 0$ *such that* $n \geq k \geq c_{\mathsf{nm}} t^2 \log^2(n/\varepsilon)$*, there exists a* $\mathrm{poly}(n,\log(1/\varepsilon))$*-time computable* (k,ε) t*-non-malleable extractor*

$$\mathsf{nmExt}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$$

with $m = \frac{k}{3t}$ *output bits and seed length* $d = c_{\mathsf{nm}} t^2 \log^2(n/\varepsilon)$*.*

2.6 Fooling AC Circuits

A Boolean circuit is an $\mathsf{AC}[d,s]$ circuit if it has depth d, size s and unbounded fan-in. We say that a circuit C with n input bits is ε*-fooled* by a distribution D if $\mathsf{SD}(C(D),D(U_n)) \leq \varepsilon$.

Harsha and Srinivasan [29], improving upon Braverman's seminal result [14] (see also [44]) proved:

▶ **Theorem 17** ([29])**.** *There exists a constant* $c > 0$ *such that the following holds. For every integers* s,d,t*, any* $\mathsf{AC}[d,s]$ *circuit is* ε*-fooled by any* t*-wise independent distribution, where* $\varepsilon = 2^{-\frac{t}{(\log s)^{c\cdot d}}}$*.*

[3] That is, for every i and every x, we have $f_i(x) \neq x$.

We need a slight generalization of Theorem 17:

▶ **Lemma 18.** *There exists a constant $c > 0$ such that the following holds for every integers n, m, d, s, where $m \leq s$. Let $C\colon \{0,1\}^n \to \{0,1\}^m$ be an $\mathsf{AC}[d,s]$ circuit. Then, C is ε-fooled by any t-wise independent distribution, where $\varepsilon = 2^{m-\frac{t}{(\log s)^{c\cdot d}}}$.*

Proof. Fix some $z \in \{0,1\}^m$ and consider the circuit $C_z\colon \{0,1\}^n \to \{0,1\}$ that given an input $x \in \{0,1\}^n$ checks whether $C(x) = z$. C_z can be constructed by adding an AND gate and m comparators on top of C, so clearly C_z is an $\mathsf{AC}[d+2, s']$ circuit for $s' = s + O(m)$. By Theorem 17, every t-wise distribution ε'-fools C_z, where

$$\varepsilon' = 2^{-\frac{t}{(\log s')^{c'\cdot(d+2)}}} \leq 2^{-\frac{t}{(\log s)^{c\cdot d}}}$$

for some universal constants $c, c' > 0$ (using the fact that $m \leq s$). That is, for every t-wise distribution D and $z \in \{0,1\}^m$, $\mathsf{SD}(C_z(D), C_z(U_n)) \leq \varepsilon'$. Now,

$$\begin{aligned}\varepsilon = \mathsf{SD}(C(D), C(U_n)) &= \frac{1}{2}\sum_{z\in\{0,1\}^m} |\Pr[C(D)=z] - \Pr[C(U_n)=z]| \\ &= \sum_{z\in\{0,1\}^m} \frac{1}{2}|\mathbb{E}[C_z(D)] - \mathbb{E}[C_z(U_n)]| \leq 2^m\varepsilon',\end{aligned}$$

as desired. ◀

3 Entropy-Resilient Functions

▶ **Definition 19** (Non-oblivious sources). *Let $\Sigma = \{0,1\}^m$. A (q,t)-non-oblivious Σ-fixing source $X = (X_1, \ldots, X_r)$ is a random variable over $\Sigma^r = \{0,1\}^{rm}$ for which there exists a set $R_{\mathsf{bad}} \subseteq [r]$ of cardinality $q' \leq q$ such that:*

- *The joint distribution of $\{(X_i)_j \mid i \in [r] \setminus R_{\mathsf{bad}}, j \in [m]\}$, denoted by G_X, is t-wise independent over $\{0,1\}^{(r-q')m}$; and*
- *Each of the random variables in $B_X \triangleq \{(X_i)_j\}$ with $i \in R_{\mathsf{bad}}$ and $j \in [m]$ may depend arbitrarily on all other random variables in G_X and B_X.*

If $t = (r-q')m$ we say X is a q-non-oblivious Σ-fixing source. If $m = 1$ we say X is a bit-fixing source and the definition coincides with the standard definition of non-oblivious bit-fixing sources [11]. When X is clear from context, we write G and B for G_X and B_X, respectively.

▶ **Definition 20** (Entropy-resilient functions). *Let $\Sigma = \{0,1\}^m$. A function $f\colon \Sigma^r \to \Sigma$ is a* (q,t,g,ε)-entropy-resilient function *if for every (q,t)-non-oblivious Σ-fixing source X over Σ^r, the output $f(X)$ is ε-close to an $(m, m-g)$-source. If $g = 0$ we say f is (q,t,ε)-resilient.*

3.1 Functions With One Output Bit

▶ **Definition 21.** *Let $f\colon \{0,1\}^r \to \{0,1\}$ be an arbitrary function. Let X be a (q,t)-non-oblivious bit-fixing source over $\{0,1\}^r$. Define $E(f)$ to be the event in which the bits tossed by the good players do not determine the value of the function f. We define the influence of the bad players by $I(f) = \Pr[E(f)]$.*

Balanced resilient functions can be seen as deterministic extractors against non-oblivious bit-fixing sources outputting one bit. Chattopadhyay and Zuckerman [17], followed by an improvement by Meka [37], derandomized the Ajtai-Linial function [2] and obtained an explicit construction of an almost-balanced resilient function which is also computable by monotone AC^0 circuits.

▶ **Theorem 22** ([17, 37]). *For every constant $0 < \delta < 1$, there exists a constant $c_\delta \geq 1$ such that for every constant $c \geq c_\delta$ and integer r there exists a monotone function $\mathsf{Res}\colon \{0,1\}^r \to \{0,1\}$ such that for every $t \geq c \log^4 r$,*

- *For every (q,t)-non-oblivious bit-fixing source X, $I(\mathsf{Res}) \leq c \cdot \frac{q}{r^{1-\delta}}$.*
- *For every t-wise independent distribution D, $\mathrm{bias}(\mathsf{Res}(D)) \leq r^{-1/c}$.*

The function Res is computable by a uniform depth 3 monotone circuit of size r^c. Further, the function $c_\delta(\delta)$ is continuous and monotonically decreasing.

Throughout the paper we make use of the following corollary.

▶ **Corollary 23.** *For every constant $0 < \gamma < 1$ there exist constants $0 < \alpha < \beta < 1$ such that for every integer r there exists a function $\mathsf{Res}\colon \{0,1\}^r \to \{0,1\}$ which for every $t \geq \frac{1}{\beta} \log^4 r$ satisfies: For every $(r^{1-\gamma}, t)$-non-oblivious bit-fixing source X,*

$$I(\mathsf{Res}) \leq \frac{1}{\beta} \cdot r^{-\alpha},$$
$$\mathrm{bias}\left(\mathsf{Res}(X) \mid \neg E(\mathsf{Res})\right) \leq \frac{3}{\beta} \cdot r^{-\alpha}.$$

The function Res is computable by a uniform depth 3 monotone circuit of size $r^{\frac{1}{\beta}}$.

Proof. Using the notations of Theorem 22, assume that for every η, $c_\eta > \frac{1}{2\eta}$ (if not, we can always increases c_η). Given $\gamma > 0$, set δ to be the constant satisfying the equation $f(\delta) = \delta - \gamma + \frac{1}{2c_\delta} = 0$. Such a δ exists, as $f(\delta) \leq 2\delta - \gamma$ and therefore $f(\delta) < 0$ when δ approaches 0, and $f(\delta) > 0$ when δ approaches γ. Note that by our choice of δ, it holds that

$$\delta < \gamma = \delta + \frac{1}{2c_\delta} < \delta + \frac{1}{c_\delta}.$$

Set $\alpha = \gamma - \delta > 0$ and $\beta = \frac{1}{c_\delta}$. Note that indeed $\beta > \alpha$.

By Theorem 22, applied with the constant δ, it holds that $I(\mathsf{Res}) \leq c_\delta \frac{r^{1-\gamma}}{r^{1-\delta}} = \frac{1}{\beta} r^{-\alpha}$. Further, $\mathrm{bias}(\mathsf{Res}(D)) \leq r^{-\beta}$.

Following similar arguments as in [17], we have that $\mathrm{bias}(\mathsf{Res}(X)) \leq \frac{1}{\beta} r^{-\alpha} + r^{-\beta}$, so

$$\mathrm{bias}\left(\mathsf{Res}(X) \mid \neg E(\mathsf{Res})\right) \leq \frac{\frac{1}{\beta} r^{-\alpha} + r^{-\beta}}{1 - \frac{1}{\beta} r^{-\alpha}} \leq \frac{3}{\beta} r^{-\alpha}. \qquad \blacktriangleleft$$

3.2 Functions With Multiple Output Bits

The output bit of a (q,t,ε)-resilient function $f\colon \{0,1\}^r \to \{0,1\}$ applied to a (q,t)-non-oblivious bit-fixing source is indeed ε-close to uniform, but, as shown by [30] even when $q = 1$, ε cannot be smaller that $\frac{\ln r}{r}$ (and the simpler bound $\varepsilon \geq \frac{1}{r}$ is almost trivial). We show that when we output many bits, and allow $o(\log \frac{1}{\varepsilon})$ entropy gap, we may obtain much smaller error. We do that by exhibiting an entropy-resilient function based on a parallel application of the (derandomized version of the) Ajtai-Linial function.

A construction of an entropy-resilient function. Given a constant $0 < \gamma < 1$ and integers $r \geq m$ let $\mathsf{Res}\colon \{0,1\}^r \to \{0,1\}$ be the function guaranteed by Corollary 23 with respect to γ. Define $\Sigma = \{0,1\}^m$ and $\mathsf{EntRes}\colon \Sigma^r \to \Sigma$ as follows. On input $x \in \Sigma^r$,

$$\mathsf{EntRes}(x) = (\mathsf{Res}(x^{(1)}), \ldots, \mathsf{Res}(x^{(m)})),$$

where x_i stands for the i-th column of x, when we view x as a $r \times m$ table.

▶ **Theorem 24.** *For every constant $0 < \gamma < 1$ there exist constants $0 < \alpha < 1$ and $c' \geq 1$ such that the following holds. For every integers r, $m \leq r^{\alpha/2}$, every $\varepsilon > 0$, and for every integer $t \geq m \cdot (\log r)^{c'}$, the function $\mathsf{EntRes}\colon \Sigma^r \to \Sigma$ is $(q = r^{1-\gamma}, t, g, \varepsilon)$-entropy-resilient with entropy gap $g = o(\log(1/\varepsilon))$.*

The proof of Theorem 24 is done in two steps. First, in Section 3.2.1, we analyze the theorem for the special case in which the distribution G_X of the given non-oblivious Σ-fixing source X is uniform. Then, based on that result, in Section 3.2.2 we prove Theorem 24.

3.2.1 The Uniform Case

In this section, we prove the following lemma.

▶ **Lemma 25.** *Keeping the notations of Theorem 24, the function $\mathsf{EntRes}\colon \Sigma^r \to \Sigma$ is $(q = r^{1-\gamma}, g, \varepsilon)$-entropy-resilient with entropy gap*

$$g = c_{\mathsf{ent}_1} \frac{\ln \frac{1}{\varepsilon}}{\ln \ln \frac{1}{\varepsilon} + c_{\mathsf{ent}_2} \ln r} = o(\log(1/\varepsilon))$$

for some universal constant $c_{\mathsf{ent}_1} > 0$ and a constant $c_{\mathsf{ent}_2} > 0$ that depends only on γ.

Proof of Lemma 25. Let X be a $(q = r^{1-\gamma})$-non-oblivious Σ-fixing source. Let $R_{\mathsf{bad}} \subseteq [r]$ be the set of bad players, and E_i the event that the values of the good players in $X^{(i)}$ do not determine the value of Res. Note that we shall also denote E_i as an indicator for that event.

By Corollary 23, there exists constants $0 < \alpha < \beta < 1$ such that $\Pr[E_i = 1] \leq \frac{1}{\beta} \cdot r^{-\alpha}$ for every $i \in [m]$. Observe that the random variables $E_1, \ldots, E_m$ are independent, as the value of E_i depends only on the values of the good players in the i-th column, and by assumption all these values are independent of the corresponding values in the other columns. Write $\mu = m \cdot \frac{1}{\beta} \cdot r^{-\alpha}$ and note that since $m \leq r^{\alpha/2}$, $\mu < 1$. Set

$$c = \frac{4 \ln \frac{1}{\varepsilon}}{\mu} \cdot \frac{1}{\ln \frac{\ln \frac{1}{\varepsilon}}{\mu}}$$

and observe that $c > 1$. By the Chernoff bound,

$$\Pr\left[\sum_{i=1}^{m} E_i > c\mu\right] \leq \left(\frac{e^{c-1}}{c^c}\right)^{\mu} \leq e^{-\frac{1}{2}\mu c \ln c} \leq \varepsilon,$$

where the last inequality follows from the fact that $c \ln c \geq \frac{2 \ln \frac{1}{\varepsilon}}{\mu}$.

By Corollary 23, for every $i \in [m]$,

$$\mathrm{bias}\left(\mathsf{Res}(X_i) \mid E_i = 0\right) \leq \frac{3}{\beta} \cdot r^{-\alpha}.$$

Assume that the event $\sum_{i=1}^{m} E_i \leq c\mu$ holds, and let $I \subseteq [m]$, $|I| \geq m - c\mu$ be the set of good columns I for which $E_i = 0$. For every $w \in \{0,1\}^m$, since the random variables $\{\mathsf{EntRes}(X)_i\}_{i \in I}$ are independent, we have:

$$\Pr[\mathsf{EntRes}(X) = w] \leq \Pr[\mathsf{EntRes}(X)_I = w_I] \leq \left(\frac{1}{2} + \frac{3}{\beta} \cdot r^{-\alpha}\right)^{m-c\mu}$$
$$\leq 2^{-m+c\mu} e^{\frac{6}{\beta} r^{-\alpha} m} \leq 2^{-m+c\mu} 2^{10\mu}.$$

Now, we have

$$c\mu + 10\mu \;\leq\; 2c\mu \;\leq\; \frac{8 \ln \frac{1}{\varepsilon}}{\ln \ln \frac{1}{\varepsilon} + \ln \frac{1}{\mu}} \;\leq\; \frac{8 \ln \frac{1}{\varepsilon}}{\ln \ln \frac{1}{\varepsilon} + \frac{4}{\alpha} \ln r} \;=\; o\left(\log \frac{1}{\varepsilon}\right).$$

We have shown that except with probability ε, the output $\mathsf{EntRes}(X)$ has min-entropy $m - o(\log(1/\varepsilon))$, as desired. More specifically, the min-entropy in the good columns alone is at least $m - o(\log(1/\varepsilon))$, and we stress that the good columns are not fixed but depend on the sample itself. ◀

3.2.2 The Bounded-Independence Case – Proof of Theorem 24

Throughout this section, we use the same notations as in Lemma 25. We are given X that is a (q,t)-non-oblivious Σ-fixing source. We use a similar approach to the one taken in [17]. For the sake of the proof, we:

- Let GU be the distribution in which the good players are jointly uniform, and the bad players are arbitrary.
- Define a small-depth circuit C' that is related to EntRes so that $H_\infty(\mathsf{EntRes}(X)) \geq H_\infty(C'(X))$.

We will show that $C'(X)$ and $C'(GU)$ are statistically close to each other. Finally, the results of Section 3.2.1 proves that except for a small probability, $H_\infty(C'(GU)) \geq m - o(\log(1/\varepsilon))$.

Proof of Theorem 24. Fix a (q,t)-non-oblivious Σ-fixing source X. Let GU be the distribution where the good players are jointly uniform, and the bad players are arbitrary. We construct a circuit $C' \colon \{0,1\}^{rm} \to \{0,1\}^m$ such that:

$$(C'(x))_i = \begin{cases} \mathsf{EntRes}(x)_i & \text{If } E_i(x) = 0, \\ 0 & \text{Otherwise.} \end{cases}$$

Recall that E_i is *fully determined* by the good players, and so does $\mathsf{EntRes}(X)_i$ when $E_i = 0$. Hence, C' is fully determined by the good players.

We can write a small-depth circuit computing C'. Let C be the depth-3 size $r^{1/\beta}$ circuit that computes the function $\mathsf{Res} \colon \{0,1\}^r \to \{0,1\}$ as guaranteed by Theorem 22. Construct a circuit for C' as follows:

- For $i \in [m]$ and $b \in \{0,1\}$ let $C_{i,b}$ be a copy of C where we wire $(x_i)_j$ for every good player $j \in [r]$, and the value b for every bad player.
- The top part contains m comparators, outputting the output of $C_{i,0}$ if the output of $C_{i,0}$ is the same as the output of $C_{i,1}$, and 0 otherwise.

The circuit has depth 4 and size $s'' = O(mr^{1/\beta})$ and its correctness is guaranteed by the fact that Res is monotone (so it is sufficient to consider the case where the bad players voted unanimously).

By Lemma 18, $\mathsf{SD}(C'(GU), C'(X)) \leq 2^{m - \frac{t}{(\log(mr))^{c''}}}$ for some large enough universal constant $c'' > 0$. For every $w \in \{0,1\}^m$:

$$\begin{aligned}\Pr\left[\mathsf{EntRes}(X) = w\right] &\leq \Pr\left[\mathsf{EntRes}(X)_I = w_I\right] = \Pr\left[C'(X)_I = w_I\right] \\ &\leq \Pr\left[C'(GU)_I = w_I\right] + 2^{m - \frac{t}{(\log(mr))^{c''}}} \\ &\leq 2^{-m + \frac{8\ln\frac{1}{\varepsilon}}{\ln\ln\frac{1}{\varepsilon} + \frac{4}{\alpha}\ln r}} + 2^{m - \frac{t}{(\log(mr))^{c''}}},\end{aligned}$$

where in the last inequality we have used Lemma 25. We can set the constant c' stated in the theorem to be larger than c'' and get that

$$\Pr\left[\mathsf{EntRes}(X) = w\right] \;\leq\; 2 \cdot 2^{-m + \frac{8\ln\frac{1}{\varepsilon}}{\ln\ln\frac{1}{\varepsilon} + \frac{4}{\alpha}\ln r}}.$$

To conclude, note that the above holds with probability at least $1-\varepsilon$, and then $\mathsf{EntRes}(X)$ has min-entropy at least $-1 + m - \frac{8\ln\frac{1}{\varepsilon}}{\ln\ln\frac{1}{\varepsilon} + \frac{4}{\alpha}\ln r} = m - o(\log(1/\varepsilon))$, as desired. ◀

4 Low Error Two-Source Condensers With High Entropy Loss

Chattopadhyay and Zuckerman [17] showed a reduction from two independent sources to non-oblivious bit-fixing sources. In Section 4.1 we extend this to many output bits and show a reduction from two independent sources to non-oblivious Σ-fixing sources. Our reduction is similar to the one in [17], and here:

- We let the non-malleable extractors output m bits rather than a single bit, obtaining a non-oblivious Σ-fixing source for $\Sigma = \{0,1\}^m$.
- Correspondingly, we apply our entropy-resilient function EntRes whereas in [17] the function Res is applied.

In Section 4.2 we use this together with the results of Section 3 to get a low error two-source condenser with many output bits, yet still far from getting almost all of the possible entropy from the two sources.

4.1 From Two Independent Sources to a Non-Oblivious Σ-Fixing Source

In this section, we revisit the [17] transformation of two independent sources to a non-oblivious bit-fixing source (i.e., with $m = 1$), and extend it to sources with several bits. Throughout this section, we refer to $c_{\mathsf{GUV}}, c_{\mathsf{nm}}$ as the constants that appear in Theorem 8 and Theorem 16, respectively.

▶ **Theorem 26.** *For every integers n, t, m, k, with $n \geq k \geq (tm\log n)^5$ and set $\Sigma = \{0,1\}^m$, there exists a $\mathrm{poly}(n)$-time computable function $\mathsf{TwoSourcesToNOF}\colon \{0,1\}^n \times \{0,1\}^n \to \Sigma^r$, where $r = n^{2c_{\mathsf{GUV}}}$ such that the following holds. Let X_1, X_2 be a pair of independent (n,k)-sources. Then, with probability at least $1 - 2^{-k/2}$ over $x_2 \sim X_2$, the output*

$$\mathsf{TwoSourcesToNOF}(X_1, x_2)$$

is (n^{-mt})-close to an $\left(r^{1-\frac{1}{4c_{\mathsf{GUV}}}}, t\right)$-non-oblivious Σ-fixing source.

Proof. We start by setting the following parameters:

Setting of parameters.

- Set $\varepsilon_{\mathsf{GUV}} = \frac{1}{n}$.
- Set $d_{\mathsf{GUV}} = c_{\mathsf{GUV}} \log\left(\frac{n}{\varepsilon_{\mathsf{GUV}}}\right) = 2c_{\mathsf{GUV}} \log n$.
- Set $\varepsilon_{\mathsf{nm}} = 2^{-4mt(d_{\mathsf{GUV}}+\log m)}$.
- Set $d_{\mathsf{nm}} = c_{\mathsf{nm}} t^2 \log^2\left(\frac{n}{\varepsilon_{\mathsf{nm}}}\right)$.

Note that $\varepsilon_{\mathsf{nm}} = 2^{-\Theta(mt\log n)}$ and that $d_{\mathsf{nm}} = \Theta(t^4 m^2 \log^2 n)$.

Building blocks. For the construction of $\mathsf{TwoSourcesToNOF}$, we make use of the following ingredients:

- Let $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^{d_{\mathsf{GUV}}} \to \{0,1\}^{d_{\mathsf{nm}}}$ be the $(k/2, \varepsilon_{\mathsf{GUV}})$-strong-seeded-extractor, guaranteed by Theorem 8. One can verify that $k/2 \geq 2d_{\mathsf{nm}}$ as required by Theorem 8.
- Let $\mathsf{nmExt}\colon \{0,1\}^n \times \{0,1\}^{d_{\mathsf{nm}}} \to \{0,1\}^m$ be the $(k, \varepsilon_{\mathsf{nm}})$ t-non-malleable extractor, guaranteed by Theorem 16. Note that $k \geq 3tm$ so the hypothesis of Theorem 16 is met with our choice of parameters.

The construction. We identify $[r]$ with $\{0,1\}^{d_{\mathsf{GUV}}}$. On inputs $x_1, x_2 \in \{0,1\}^n$, we define $\mathsf{TwoSourcesToNOF}(x_1, x_2)$ to be the $r \times m$ matrix whose i-th row is given by

$$\mathsf{TwoSourcesToNOF}(x_1, x_2)_i = \mathsf{nmExt}(x_1, \mathsf{Ext}(x_2, i)).$$

Analysis. Write $D_{\mathsf{nm}} = 2^{d_{\mathsf{nm}}}$ and identify $[D_{\mathsf{nm}}]$ with $\{0,1\}^{d_{\mathsf{nm}}}$. Let $G \subseteq [D_{\mathsf{nm}}]$, $|G| \geq (1 - \sqrt{\varepsilon_{\mathsf{nm}}})D_{\mathsf{nm}}$, be the set of good seeds guaranteed by Lemma 14. By Lemma 15, for any distinct $r_1, \ldots, r_t \in G$,

$$(\mathsf{nmExt}(X_1, r_1), \ldots, \mathsf{nmExt}(X_1, r_t)) \approx_{t\sqrt{\varepsilon_{\mathsf{nm}}}} U_{tm}.$$

Let $S(X_2) = \left\{\mathsf{Ext}(X_2, 1), \ldots, \mathsf{Ext}(X_2, 2^{d_{\mathsf{nm}}})\right\}$. By Theorem 9,

$$\Pr_{x_2 \sim X_2}\left[|S(x_2) \cap G| \leq (1 - \sqrt{\varepsilon_{\mathsf{nm}}} - \varepsilon_{\mathsf{GUV}}) \cdot r\right] \leq 2^{-k/2}.$$

We say that $x_2 \in \mathsf{supp}(X_2)$ is good if it induces a good sample, that is if $|S(x_2) \cap G| > (1 - \sqrt{\varepsilon_{\mathsf{nm}}} - \varepsilon_{\mathsf{GUV}})r$. Fix a good x_2 and let $Z = \mathsf{TwoSourcesToNOF}(X_1, x_2)$. In the good seeds, every t elements of Z are $(t\sqrt{\varepsilon_{\mathsf{nm}}})$-close to uniform, and there are at most $q \leq (\sqrt{\varepsilon_{\mathsf{nm}}} + \varepsilon_{\mathsf{GUV}})r$ bad rows. Applying Lemma 6, we get that Z is $\zeta = t\sqrt{\varepsilon_{\mathsf{nm}}}(rm)^{mt}$-close to a (q,t)-non-oblivious bit-fixing source. By our choice of $\varepsilon_{\mathsf{nm}}$,

$$\zeta = 2^{-2mt(d_{\mathsf{GUV}}+\log m)} 2^{mt\log(rm)} \leq 2^{-mt\log r} \leq n^{-mt}.$$

Further,

$$q \leq (\sqrt{\varepsilon_{\mathsf{nm}}} + \varepsilon_{\mathsf{GUV}})r \leq 2\varepsilon_{\mathsf{GUV}} r = 2r^{-\frac{1}{2c_{\mathsf{GUV}}}+1} \leq r^{1-\frac{1}{4c_{\mathsf{GUV}}}}.$$

We now analyse the running-time. We first apply Ext to compute $S(x_2)$, which takes time $\mathrm{poly}(n, \log(1/\varepsilon_{\mathsf{GUV}})) = \mathrm{poly}(n)$. Then, applying each nmExt takes $\mathrm{poly}(n, \log(1/\varepsilon_{\mathsf{nm}})) = \mathrm{poly}(n, m, t, d_{\mathsf{GUV}}) = \mathrm{poly}(n)$ time and we do it for $r = \mathrm{poly}(n)$ times. Overall, the running time is $\mathrm{poly}(n)$, as required. In particular, as $n \geq k \geq m$, the running time is also poly-logarithmic in the errors of the construction, $2^{-k/2}$ and n^{-mt}. ◀

4.2 Low Error Condensers With High Entropy Loss

▶ **Theorem 27.** *There exists a universal constant $c \geq 1$ such that the following holds. For every integers n, k, m and every $\varepsilon > 0$ such that $n \geq k \geq (m \log(n/\varepsilon))^c$ there exists a* $\mathrm{poly}(n)$*-time computable* $((n,k),(n,k)) \to_{\varepsilon, 2^{-k/2}} (m, m-g)$*-condenser*

$$\mathsf{Cond}' \colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m ,$$

that is strong in the second source, with entropy gap $g = o(\log(1/\varepsilon))$.

Proof. We start by describing the construction of our condenser Cond' and then turn to the analysis. As usual, we let c_{GUV} be the constant that is given by Theorem 8.

Setting of parameters.

- Set $\gamma = \frac{1}{4c_{\mathsf{GUV}}}$ and let $0 < \alpha < \beta < 1$ and c' be the constants from Theorem 24 with respect to this γ.
- Set $r = n^{2c_{\mathsf{GUV}}}$.
- Set $t = m \cdot (\log(r/\varepsilon))^{c'}$.
- Set c, the constant stated in this theorem, to $c = \max(10c', 2/\alpha)$.

Building blocks.

- Let $\mathsf{TwoSourcesToNOF} \colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{r \times m}$ be the function that is given by Theorem 26. We are about to apply $\mathsf{TwoSourcesToNOF}$ to (n,k)-sources, and indeed k is large enough to satisfy the hypothesis of Theorem 26.
- Let $\mathsf{EntRes} \colon \{0,1\}^{r \times m} \to \{0,1\}^m$ be the function from Theorem 24 when set with the parameter γ as defined above. Note that the hypothesis of Theorem 24 holds, as since $c \geq 2/\alpha$ we have that $m < r^{\alpha/2}$, and t is large enough.

The construction. On inputs $x_1, x_2 \in \{0,1\}^n$, we define

$$\mathsf{Cond}'(x_1, x_2) = \mathsf{EntRes}\left(\mathsf{TwoSourcesToNOF}(x_1, x_2)\right).$$

Analysis. Clearly, EntRes is computable in $\mathrm{poly}(m, r) = \mathrm{poly}(n)$ time. Let X_1, X_2 be a pair of independent (n,k)-sources. By Theorem 26, except with probability $2^{-k/2}$ over $x_2 \sim X_2$, the output $\mathsf{TwoSourcesToNOF}(X_1, x_2)$ is n^{-mt}-close to an $(r^{1-\gamma}, t)$-non-oblivious bit-fixing source. For every x_2 for which this event holds, the output $\mathsf{EntRes}(\mathsf{TwoSourcesToNOF}(X_1, x_2))$ is $(n^{-mt} + \varepsilon)$-close to an $(m, m - o(\log(1/\varepsilon)))$-source, and $n^{-mt} \leq \varepsilon$. ◀

5 Deterministically Condensing a Single Block-Source

A distribution (X, Y) is a blockwise source if both X has sufficient min-entropy and also for every $x \in \mathsf{supp}(X)$, $(Y \mid X = x)$ has sufficient min-entropy. In this section we show how to deterministically condense a blockwise source into a source having very small entropy gap, using the connection between condensers with small entropy gap and samplers with multiplicative error and ideas from [43]. In the next section we will use it to significantly increase the output length of the condenser from Section 4.2.

▶ **Lemma 28** (Deterministically condensing a blockwise source)**.** *Let X be an (n,k)-source. Let Y be a d-bit random variable (that may depend on X) such that for every $x \in \mathsf{supp}(X)$, the random variable $(Y \mid X = x)$ is ε_{B}-close to a $(d, d-g)$-source.*

Let $\mathsf{Ext} \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a* $(k_{\mathsf{Ext}}, \varepsilon_{\mathsf{Ext}})$*-seeded-extractor. Suppose* $k \geq k_{\mathsf{Ext}} + \log(1/\varepsilon_{\mathsf{Ext}})$*. Then,* $\mathsf{Ext}(X, Y)$ *is* $(2^{g+2}\varepsilon_{\mathsf{Ext}} + 2\varepsilon_{\mathsf{B}})$*-close to an* $(m, m-g-1)$*-source.*

Proof. Fix any $T \subseteq \{0,1\}^m$. Define the set

$$\mathsf{OverHit}_T = \left\{ x \in \{0,1\}^n : \Pr_{y \sim U_d}[\mathsf{Ext}(x,y) \in T] > \mu(T) + \varepsilon_{\mathsf{Ext}} \right\}.$$

▷ Claim 29. $|\mathsf{OverHit}_T| < 2^{k_{\mathsf{Ext}}}$.

Proof. Suppose towards a contradiction that $|\mathsf{OverHit}_T| \geq 2^{k_{\mathsf{Ext}}}$ and let B denote the random variable that is uniform over the set $\mathsf{OverHit}_T$. Since B has min-entropy at least k_{Ext}, the output $\mathsf{Ext}(B, U_d)$ is $\varepsilon_{\mathsf{Ext}}$-close to uniform, and therefore $\Pr_{x \sim B, y \sim U_d}[\mathsf{Ext}(x,y) \in T] \leq \mu(T) + \varepsilon_{\mathsf{Ext}}$. This stands in contradiction to the definition of B. ◁

Now,

$$\Pr[\mathsf{Ext}(X,Y) \in T] \leq \Pr[\mathsf{Ext}(X,Y) \in T \mid X \notin \mathsf{OverHit}_T] + \Pr[X \in \mathsf{OverHit}_T].$$

By Claim 29, $\Pr[X \in \mathsf{OverHit}_T] \leq 2^{k_{\mathsf{Ext}} - k}$. Also, for every $x \notin \mathsf{OverHit}_T$ let

$$GY_x = \left\{ y \in \{0,1\}^d : \mathsf{Ext}(x,y) \in T \right\}.$$

By definition, $\mu(GY_x) \leq \mu(T) + \varepsilon_{\mathsf{Ext}}$. Also, $Y|(X=x)$ is ε_{B}-close to some random variable Y'_x having support size at least 2^{d-g}. Therefore,

$$\begin{aligned} \Pr_{y \sim (Y|X=x)}[\mathsf{Ext}(x,y) \in T] &= \Pr_{y \sim (Y|X=x)}[y \in GY_x] \leq \varepsilon_{\mathsf{B}} + \Pr[Y'_x \in GY_x] \\ &\leq \varepsilon_{\mathsf{B}} + \frac{|GY_x|}{2^{d-g}} \leq \varepsilon_{\mathsf{B}} + 2^g(\mu(T) + \varepsilon_{\mathsf{Ext}}). \end{aligned}$$

Thus,

$$\begin{aligned} \Pr[\mathsf{Ext}(X,Y) \in T] &\leq \Pr[\mathsf{Ext}(X,Y) \in T \mid X \notin \mathsf{OverHit}_T] + \Pr[X \in \mathsf{OverHit}_T] \\ &\leq 2^g \mu(T) + 2^g \varepsilon_{\mathsf{Ext}} + \varepsilon_{\mathsf{B}} + 2^{k_{\mathsf{Ext}} - k} \leq 2^g \mu(T) + (2^g + 1)\varepsilon_{\mathsf{Ext}} + \varepsilon_{\mathsf{B}}. \end{aligned}$$

But,

▷ Claim 30. Let Z be a random variable over n-bit strings such that for every $T \subseteq \{0,1\}^n$, $\Pr[Z \in T] \leq 2^g \mu(T) + \varepsilon$. Then, Z is 2ε-close to an $(n, n-g-1)$-source.

Proof. Set $H = \left\{ x : \Pr[Z = x] > 2^{-(n-g-1)} \right\}$. On the one hand,

$$\Pr[Z \in H] = \sum_{x \in H} \Pr[Z = x] \geq 2^{g+1} \mu(H).$$

On the other hand, by our assumption, $\Pr[Z \in H] \leq 2^g \mu(H) + \varepsilon$. Together, we get that $2^g \mu(H) \leq \varepsilon$. Thus, $\Pr[Z \in H] \leq 2\varepsilon$. As H are all the heavy elements, we conclude that Z is 2ε-close to a distribution with $n - g - 1$ min-entropy. ◁

We can therefore summarize that $\mathsf{Ext}(X,Y)$ is $(2^{g+2}\varepsilon_{\mathsf{Ext}} + 2\varepsilon_{\mathsf{B}})$-close to an $(m, m-g-1)$-source. ◀

6 Low Error Two-Source Condensers

In this section we will construct our low error condenser, with small entropy gap outputting many bits, by exploiting the block-wise structure of our previous construction. Roughly speaking, we are close to a scenario in which X_2 has sufficient min-entropy and also for every fixing of $x_2 \in \mathsf{supp}(X_2)$, the random variable $\mathsf{Cond}'(X_1, x_2)$ is close to uniform. The result of Section 5 can then be applied – allowing us to extract almost all the entropy from one of the sources. To that end, we prove the following theorem, which readily implies Theorem 3.

▶ **Theorem 31** (Main theorem). *There exists a universal constant $c \geq 1$ such that the following holds. For every integers $n \geq k$ and every $\varepsilon > 0$ such that $k \geq \log^c(n/\varepsilon)$ there exists a $\mathrm{poly}(n, \log(1/\varepsilon))$-time computable $((n,k),(n,k)) \to_\varepsilon (m, m-g)$ two-source condenser*

$$\mathsf{Cond}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$$

with $m = k - 5\log(1/\varepsilon) - O(1)$ and $g = o(\log(1/\varepsilon))$.

Proof. We start by setting some parameters.

Parameters.

- Set $\varepsilon_{\mathsf{Cond}'} = \varepsilon/8$.
- Set $\varepsilon_{\mathsf{Ext}} = \varepsilon^2/32$.
- Set $k_{\mathsf{Ext}} = k - \log(2/\varepsilon)$.
- Set $d_{\mathsf{Ext}} = c' \log n \cdot \log(n/\varepsilon_{\mathsf{Ext}})$ where c' is the constant that is given by Theorem 10.

For the construction we make use of the following building blocks.

- Let $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^{d_{\mathsf{Ext}}} \to \{0,1\}^m$ be the $(k_{\mathsf{Ext}}, \varepsilon_{\mathsf{Ext}})$-strong-seeded-extractor that is given by Theorem 10. By that theorem, $m = k_{\mathsf{Ext}} - 2\log(1/\varepsilon_{\mathsf{Ext}}) - O(1)$.
- Let $\mathsf{Cond}'\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{d_{\mathsf{Ext}}}$ be the $((n,k),(n,k)) \to_{\varepsilon_{\mathsf{Cond}'}, 2^{-k/2}} (d_{\mathsf{Ext}}, d_{\mathsf{Ext}} - g')$-condenser, strong in the second source, that is given by Theorem 27, with $g' = o(\log(1/\varepsilon_{\mathsf{Cond}'}))$. Note that our choice of parameters satisfies the hypothesis of Theorem 27 for a large enough constant c.

The construction. On inputs $x_1, x_2 \in \{0,1\}^n$, we define

$$\mathsf{Cond}(x_1, x_2) = \mathsf{Ext}(x_2, \mathsf{Cond}'(x_1, x_2)).$$

Analysis. Let X_1, X_2 be a pair of independent (n,k)-sources. By Theorem 27, with probability at least $1 - 2^{-k/2}$ over $x_2 \sim X_2$, the random variable $\mathsf{Cond}'(X_1, x_2)$ is $\varepsilon_{\mathsf{Cond}'}$-close to a $(d, d-g')$-source. Lemma 28 implies that $\mathsf{Ext}(X_2, \mathsf{Cond}'(X_1, X_2))$ is $2^{-k/2} + (2^{g'+2}\varepsilon_{\mathsf{Ext}} + 2\varepsilon_{\mathsf{Cond}'})$-close to an $(m, m-g'-1)$-source.

By our choice of parameters, $2^{-k/2} + 2^{g'+1}\varepsilon_{\mathsf{Ext}} + 2\varepsilon_{\mathsf{Cond}'} \leq \varepsilon$. Note that $k - m = \log(2/\varepsilon) + 2\log(1/\varepsilon_{\mathsf{Ext}}) = 5\log(1/\varepsilon) + O(1)$. The running-time of the construction readily follows from the running-times of Cond' and Ext. ◀

References

1 H. L. Abbott. Lower bounds for some Ramsey numbers. *Discrete Mathematics*, 2(4):289–293, 1972.

2 M. Ajtai and N. Linial. The influence of large coalitions. *Combinatorica*, 13(2):129–145, 1993.

3 N. Alon. The Shannon capacity of a union. *Combinatorica*, 18(3):301–310, 1998.

4 N. Alon, O. Goldreich, and Y. Mansour. Almost k-wise independence versus k-wise independence. *Information Processing Letters*, 88(3):107–110, 2003.

5 B. Barak. A simple explicit construction of an $n^{\widetilde{O}(\log n)}$-Ramsey graph. *arXiv preprint*, 2006. arXiv:math/0601651.

6 B. Barak, G. Kindler, R. Shaltiel, B. Sudakov, and A. Wigderson. Simulating independence: New constructions of condensers, Ramsey graphs, dispersers, and extractors. *Journal of the ACM (JACM)*, 57(4):20, 2010.

7 B. Barak, A. Rao, R. Shaltiel, and A. Wigderson. 2-source dispersers for $n^{o(1)}$ entropy, and Ramsey graphs beating the Frankl-Wilson construction. *Annals of Mathematics*, 176(3):1483–1544, 2012.

8 A. Ben-Aroya, E. Chattopadhyay, D. Doron, X. Li, and A. Ta-Shma. A New Approach for Constructing Low-Error, Two-Source Extractors. In *LIPIcs-Leibniz International Proceedings in Informatics*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

9 A. Ben-Aroya, D. Doron, and A. Ta-Shma. An efficient reduction from two-source to non-malleable extractors: achieving near-logarithmic min-entropy. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1185–1194. ACM, 2017.

10 A. Ben-Aroya, D. Doron, and A. Ta-Shma. Near-Optimal Erasure List-Decodable Codes. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2018.

11 M. Ben-Or and N. Linial. Collective coin flipping, robust voting schemes and minima of Banzhaf values. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 408–416. IEEE, 1985.

12 E. Ben-Sasson and N. Zewi. From affine to two-source extractors via approximate duality. In *Proceedings of the 43rd annual ACM Symposium on Theory of computing (STOC)*, pages 177–186. ACM, 2011.

13 J. Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 1(01):1–32, 2005.

14 M. Braverman. Polylogarithmic independence fools AC0 circuits. *Journal of the ACM (JACM)*, 57(5):28, 2010.

15 E. Chattopadhyay, V. Goyal, and X. Li. Non-malleable extractors and codes, with their many tampered extensions. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 285–298. ACM, 2016.

16 E. Chattopadhyay and X. Li. Explicit Non-Malleable Extractors, Multi-Source Extractors and Almost Optimal Privacy Amplification Protocols. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 158–167. IEEE, 2016.

17 E. Chattopadhyay and D. Zuckerman. Explicit two-source extractors and resilient functions. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, pages 670–683. ACM, 2016.

18 B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

19 F. R. K. Chung. A note on constructive methods for Ramsey numbers. *Journal of Graph Theory*, 5(1):109–113, 1981.

20 G. Cohen. Local correlation breakers and applications to three-source extractors and mergers. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 845–862. IEEE, 2015.

21 G. Cohen. Making the Most of Advice: New Correlation Breakers and Their Applications. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 188–196. IEEE, 2016.

22 G. Cohen. Two-source extractors for quasi-logarithmic min-entropy and improved privacy amplification protocols. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2016.

23 G. Cohen. Two-source dispersers for polylogarithmic entropy and improved Ramsey graphs. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 278–284. ACM, 2016.

24 G. Cohen and L. Schulman. Extractors for Near Logarithmic Min-Entropy. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, page 14, 2016.

25 P. Frankl. A constructive lower bound for Ramsey numbers. *Ars Combinatoria*, 3(297-302):28, 1977.

26 P. Frankl and R. M. Wilson. Intersection theorems with geometric consequences. *Combinatorica*, 1(4):357–368, 1981.

27 V. Grolmusz. Low rank co-diagonal matrices and Ramsey graphs. *Journal of combinatorics*, 7(1):R15–R15, 2001.

28 V. Guruswami, C. Umans, and S. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes. *Journal of the ACM*, 56(4):20, 2009.

29 P. Harsha and S. Srinivasan. On Polynomial Approximations to AC0. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, 2016.

30 K. Kahn, G. Kalai, and N. Linial. The influence of variables on Boolean functions. In *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 68–80. IEEE, 1988.

31 Mark Lewko. An explicit two-source extractor with min-entropy near 4/9. *arXiv preprint*, 2018. `arXiv:1804.05451`.

32 X. Li. Extractors for a constant number of independent sources with polylogarithmic min-entropy. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 100–109, 2013.

33 X. Li. New independent source extractors with exponential improvement. In *Proceedings of the 45th annual ACM Symposium on Theory of Computing (STOC)*, pages 783–792. ACM, 2013.

34 X. Li. Three-source extractors for polylogarithmic min-entropy. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 863–882. IEEE, 2015.

35 X. Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1144–1156. ACM, 2017.

36 X. Li. Non-malleable extractors and non-malleable codes: Partially optimal constructions. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2018.

37 R. Meka. Explicit resilient functions matching Ajtai-Linial. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1132–1148. SIAM, 2017.

38 Zs. Nagy. A constructive estimation of the Ramsey numbers. *Mat. Lapok*, 23:301–302, 1975.

39 M. Naor. Constructing Ramsey graphs from small probability spaces. *IBM Research Report RJ*, 8810, 1992.

40 A. Rao. A 2-source almost-extractor for linear entropy. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 549–556. Springer, 2008.

41 A. Rao. Extractors for a constant number of polynomially small min-entropy independent sources. *SIAM Journal on Computing*, 39(1):168–194, 2009.

42 R. Raz. Extractors with weak random seeds. In *Proceedings of the 37th annual ACM Symposium on Theory of Computing (STOC)*, pages 11–20. ACM, 2005.

43 R. Raz, O. Reingold, and S. Vadhan. Error reduction for extractors. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 191–201. IEEE, 1999.

44 A. Tal. Tight bounds on the Fourier spectrum of AC0. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 79. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

45 John von Neumann. Various techniques used in connection with random digits. *John von Neumann, Collected Works*, 5:768–770, 1963.

46 D. Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, 11(4):345–367, 1997.

47 D. Zuckerman. Linear Degree Extractors and the Inapproximability of Max Clique and Chromatic Number. *Theory of Computing*, 3:103–128, 2007.

Efficient Average-Case Population Recovery in the Presence of Insertions and Deletions

Frank Ban
UC Berkeley, Berkeley, CA, USA
fban@berkeley.edu

Xi Chen
Columbia University, New York, NY, USA
http://www.cs.columbia.edu/~xichen
xichen@cs.columbia.edu

Rocco A. Servedio
Columbia University, New York, NY, USA
http://www.cs.columbia.edu/~rocco
rocco@cs.columbia.edu

Sandip Sinha
Columbia University, New York, NY, USA
https://sites.google.com/view/sandips
sandip@cs.columbia.edu

Abstract

A number of recent works have considered the *trace reconstruction problem*, in which an unknown source string $x \in \{0,1\}^n$ is transmitted through a probabilistic channel which may randomly delete coordinates or insert random bits, resulting in a *trace* of x. The goal is to reconstruct the original string x from independent traces of x. While the asymptotically best algorithms known for worst-case strings use $\exp(O(n^{1/3}))$ traces [8, 21], several highly efficient algorithms are known [23, 13] for the *average-case* version of the problem, in which the source string x is chosen uniformly at random from $\{0,1\}^n$. In this paper we consider a generalization of the above-described average-case trace reconstruction problem, which we call *average-case population recovery in the presence of insertions and deletions*. In this problem, rather than a single unknown source string there is an unknown distribution over s unknown source strings $x^1, \ldots, x^s \in \{0,1\}^n$, and each sample given to the algorithm is independently generated by drawing some x^i from this distribution and returning an independent trace of x^i. Building on the results of [23] and [13], we give an efficient algorithm for the average-case population recovery problem in the presence of insertions and deletions. For any support size $1 \leq s \leq \exp(\Theta(n^{1/3}))$, for a $1-o(1)$ fraction of all s-element support sets $\{x^1, \ldots, x^s\} \subset \{0,1\}^n$, for every distribution $\mathcal{D}$ supported on $\{x^1, \ldots, x^s\}$, our algorithm can efficiently recover $\mathcal{D}$ up to total variation distance at most ε with high probability, given access to independent traces of independent draws from $\mathcal{D}$. The running time of our algorithm is $\mathrm{poly}(n, s, 1/\varepsilon)$ and its sample complexity is $\mathrm{poly}(s, 1/\varepsilon, \exp(\log^{1/3} n))$. This polynomial dependence on the support size s is in sharp contrast with the *worst-case* version of the problem (when $x^1, \ldots, x^s$ may be any strings in $\{0,1\}^n$), in which the sample complexity of the most efficient known algorithm [3] is doubly exponential in s.

2012 ACM Subject Classification Mathematics of computing → Information theory; Theory of computation → Machine learning theory

Keywords and phrases population recovery, deletion channel, trace reconstruction

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.44

Category RANDOM

Funding *Xi Chen*: Supported by NSF IIS-1838154 and NSF CCF-1703925.
Rocco A. Servedio: Supported by NSF grants CCF-1563155, CCF-1814873, IIS-1838154, and by the Simons Collaboration on Algorithms and Geometry.
Sandip Sinha: Supported by NSF awards CCF-1563155, CCF-1420349, CCF-1617955, CCF-1740833, CCF-1421161, CCF-1714818 and Simons Foundation (#491119).



© Frank Ban, Xi Chen, Rocco A. Servedio, and Sandip Sinha;
licensed under Creative Commons License CC-BY



1 Introduction

Background: Worst-case and average-case trace reconstruction. In the problem of *trace reconstruction in the presence of insertions and deletions*, there is an unknown and arbitrary n-bit source string $x \in \{0,1\}^n$ and the goal is to reconstruct x given access to independent *traces* of x. A *trace* of x is a copy that has been passed through a noise channel which independently removes each bit of x with some probability q (the *deletion rate*) and also independently inserts random bits according to some *insertion rate* q'.[1] Intuitively, the insertion-deletion channel (or even just the deletion channel with no insertions) is challenging to deal with because it is difficult to determine which coordinate of the source string (if any, if insertions are possible) is responsible for a given coordinate of a received trace.

The insertion / deletion trace reconstruction problem is motivated by connections to recovery problems arising in biology (see e.g. [2, 6, 1]) and has been the subject of considerable research, especially in recent years. The worst-case version of this problem, in which the source string x can be an arbitrary element of $\{0,1\}^n$, appears to be quite difficult even for small constant noise rates. In early work [4] gave an efficient algorithm that succeeds in the deletion-only model if the deletion rate q is quite low, at most $O(1/n^{1/2+\varepsilon})$. Also in the deletion-only model, [14] showed that $\exp(\tilde{O}(\sqrt{n}))$ many traces suffice for any constant deletion rate q bounded away from 1. More recently, this result was improved in simultaneous and independent works of [8] and [21], each of which showed that for any constant insertion and deletion rates q, q', $\exp(O(n^{1/3}))$ traces suffice to reconstruct any $x \in \{0,1\}^n$. These algorithms, which run in $\exp(O(n^{1/3}))$ time, give the best results to date for the worst-case problem. (On the lower bound side, recent work of [12] obtained an $\tilde{\Omega}(n^{5/4})$ lower bound on the number of traces required from the deletion channel, improving an earlier $\Omega(n)$ lower bound due to [19]. Later work of [5] improved this lower bound to $\tilde{\Omega}(n^{3/2})$.)

Since the worst-case trace reconstruction problem seems to be quite difficult, and since the assumption that the source string x is completely arbitrary may be overly pessimistic in various contexts, it is natural to consider an *average-case* version of the problem in which the source string x is assumed to be drawn uniformly at random from $\{0,1\}^n$. This average-case problem has been intensively studied, and interestingly it turns out to be significantly easier than the worst-case problem. [4] showed that for most source strings x, in the deletion-only setting only $O(\log n)$ traces suffice for deletion rates q as large as $O(1/\log n)$. [16] considered the insertion / deletion noise channel and obtained an $O(\log n)$-trace average-case algorithm for noise rates up to $O(1/\log^2 n)$, which was later improved to $O(1/\log n)$ in [25]. [14] were the first to give an efficient (using $\mathrm{poly}(n)$ traces) average-case algorithm, for the deletion-only model, that succeeds for some constant deletion rate (their algorithm could handle deletion rates up to about $q = 0.07$). Building on the worst-case results of [8] and [21], a number of significantly stronger average-case results have recently been established. [23] gave an average-case algorithm for the deletion-only problem which uses $\exp(O(\log^{1/2} n))$ many traces for any deletion rate $q < 1/2$. Improving on this, [13] gave an average-case algorithm which can handle both insertions and deletions at any constant rate and uses only $\exp(O(\log^{1/3} n))$ many traces. (A simple reduction shows that any improvement on this sample complexity for the average-case problem would imply an improvement of the $\exp(O(n^{1/3}))$-trace worst-case algorithms of [8] and [21].)

[1] A detailed description of the channel is given in Section 2. Augmented variants of this insertion / deletion noise model can also be considered, for example allowing for bit-flips as well as insertions and deletions, but unlike deletions and insertions bit-flips can typically be handled in a straightforward fashion. In this paper we confine our attention to the insertion / deletion channel.

Beyond trace reconstruction: Population recovery from the deletion channel. Inspired by a related problem known as *population recovery*, recent work of [3] has considered a challenging extension of the trace reconstruction problem. Population recovery is the problem of learning an unknown *distribution* over an unknown set of n-bit strings, given access to independent draws from the distribution that have been independently corrupted according to some noise channel. Most research in population recovery has focused on two noise models, namely the *bit-flip* noise channel (in which each coordinate is independently flipped with some fixed probability) and the *erasure* noise channel (in which each coordinate is independently replaced by '?' with some fixed probability), both of which have been intensively studied, see e.g. [10, 26, 24, 9, 20, 18, 7, 9]. [3] considered the problem of *population recovery from the deletion channel*. This is a generalization of the deletion-channel trace reconstruction problem: now there is an unknown distribution over s unknown source strings $x^1, \ldots, x^s \in \{0,1\}^n$, and each sample provided to the learner is obtained by first drawing a string x^i from this distribution and then passing it through the deletion noise channel. It is clear that this problem is at least as difficult as the trace reconstruction problem (which is the $s = 1$ case), and indeed having multiple source strings turns out to pose significant new challenges. [3] considered the worst-case version of this problem, and showed that any distribution $\mathcal{D}$ over any set of s unknown source strings can be recovered to total variation distance ε given $2^{\sqrt{n}\cdot(\log n)^{O(s)}}/\varepsilon^2$ many traces from the deletion channel. [3] also gave a lower bound, showing that for any $s \leq n^{0.49}$ at least $n^{\Omega(s)}$ many traces are required. Population recovery-type problems have also been studied in the computational biology literature, specifically for DNA storage (see e.g. [22, 27]). In these settings, the population of strings corresponds to a collection of DNA sequences.

Summarizing, while population recovery from the deletion channel is a natural problem, the above results (and the fact that it is at least as difficult as trace reconstruction) indicate that it is also a hard one. Thus it is natural to investigate *average-case* versions of this problem; this is the subject of the current work.

1.1 Our result: Average-case population recovery in the insertion / deletion model

In the average-case model we consider, there is a given *population size* $s \geq 1$, i.e. there is a set $x^1, \ldots, x^s$ of s strings which are assumed to be drawn independently and uniformly from $\{0,1\}^n$. Associated with this population is an *arbitrary* vector of non-negative probability values $p_1, \ldots, p_s$, where p_i is the probability that the distribution $\mathcal{D}$ puts on string x^i. Thus in our model the support of the distribution is "average-case" but the actual distribution over that support is "worst-case."

Building on the work of [13], our main result is a highly efficient algorithm for average-case population recovery in the presence of insertions and deletions. We show that even for extremely large population sizes s (up to $\exp(\Theta(n^{1/3}))$), the average-case population recovery problem can be solved by a highly efficient algorithm which has running time polynomial in n (the length of unknown strings), s (the population size), and $1/\varepsilon$ (where ε is the total variation distance between $\mathcal{D}$ and the distribution returned by the algorithm). The sample complexity of our algorithm is polynomial in s, $1/\varepsilon$, and $\exp(\log^{1/3} n)$. Thus our algorithm extends the average-case trace reconstruction results of [13] to the more challenging setting of s-string population recovery with essentially the best possible dependence on the new parameters s and $1/\varepsilon$ (which are not present in the original trace reconstruction problem but are inherent in the population recovery problem).

In more detail, we prove the following theorem (the exact definition of a random trace drawn from the insertion / deletion noise channel $\mathcal{C}_{q,q'}(\mathcal{D})$ will be given in Section 2):

▶ **Theorem 1.** *Fix any two constants $q, q' \in [0,1)$ as* deletion *and* insertion rates*, respectively. There is an algorithm A with the following property: Let $\delta_{hard} \geq \exp(-\Theta(n^{1/3}))$ be a* fraction of hard support sets*, let $\delta_{fail} \geq \exp(-\Theta(n^{1/3}))$ be a* failure probability*, let $\varepsilon \geq \exp(-\Theta(n^{1/3}))$ be an* accuracy parameter*, let $1 \leq s \leq \exp(\Theta(n^{1/3}))$ be a* support size*, and let $x^1, \ldots, x^s$ be a* support set *(viewed as an ordered list of strings in $\{0,1\}^n$). For at least a $(1-\delta_{hard})$-fraction of all 2^{ns} many possible s-element support sets, it is the case that for any probability distribution $\mathcal{D}$ supported on $\{x^1, \ldots, x^s\}$, given $n, s, \varepsilon, \delta_{hard}, \delta_{fail}$, and access to $\mathcal{C}_{q,q'}(\mathcal{D})$, algorithm A uses $\mathrm{poly}(s, 1/\varepsilon, \exp(\log^{1/3} n), \exp(\log^{1/3}(1/\delta_{hard})), \log(1/\delta_{fail}))$ random traces from $\mathcal{C}_{q,q'}(\mathcal{D})$, runs in time $\mathrm{poly}(n, s, 1/\varepsilon, 1/\delta_{hard}, \log(1/\delta_{fail}))$ and has the following property: with probability at least $1-\delta_{fail}$ it outputs a hypothesis distribution $\mathcal{D}'$ over $\{0,1\}^n$ such that $d_{\mathrm{TV}}(\mathcal{D}, \mathcal{D}') \leq \varepsilon$.*

Discussion. Taken together with the recent results of [3], Theorem 1 shows that the average-case and worse-case versions of population recovery in the presence of insertions and deletions have dramatically different complexities. The best known algorithm for the population size-s worst-case population recovery problem [3] has a doubly exponential dependence on s; even for s constant this sample complexity is significantly worse than the best known sample complexity for the $s = 1$ worst-case trace reconstruction problem, which is $\exp(\Theta(n^{1/3}))$ by [8, 21]. The $n^{\Omega(s)}$ sample complexity lower bound given in [3] shows that an exponential dependence on s is inherent for worst-case population recovery. In contrast, Theorem 1 shows that a *polynomial* sample complexity (and running time) dependence on s is achievable for the average-case problem, and that passing from $s = 1$ to larger values of s does not incur much increase in complexity for the average-case problem.

In independent work, [17] studied a different generalization of trace reconstruction which they called *matrix reconstruction*. Instead of reconstructing a string by sampling traces where each character of the string has some probability of being deleted, the goal in *matrix reconstruction* is to reconstruct a matrix by sampling traces where each row and column of the matrix has some probability of being deleted. They used similar techniques to those used in this paper.

1.2 Our techniques

A natural way to approach our problem is to attempt to reduce it to the $s = 1$ case, which as described above is just the average-case trace reconstruction problem which was solved by [13]. However, two challenges arise in carrying out such a reduction. The first challenge is that the analysis of [13] only gives an algorithm which succeeds on a $1 - \Theta(1/n)$ fraction of all source strings $x \in \{0,1\}^n$. So if the population size s is much larger than n, then a random population of s source strings will with high probability contain $\Theta(s/n)$ many "hard-to-reconstruct" strings. It is not clear how to proceed if these hard-to-reconstruct strings have significant weight under the distribution $\mathcal{D}$ (which they may, since the distribution $\mathcal{D}$ over the s source strings is assumed to be completely arbitrary).

We get around this challenge by showing that for any arbitrarily small δ, the algorithm of [13] can be extended in a black-box way to succeed on any $1 - \delta$ fraction of all source strings $x \in \{0,1\}^n$ (at the cost of a modest increase in running time and sample complexity depending on the value of δ). By taking $\delta \ll 1/s$, with high probability the random population will consist entirely of source strings x^i each of which could be reconstructed in isolation if we were given only traces coming from x^i.

The second (main) challenge, of course, is that we are not given traces from each individual string x^i in isolation, but rather we are given a mixture of traces from all s strings $x^1, \ldots, x^s$. The main contribution of our work is a clustering procedure which lets us (with high probability over a population of s random source strings) correctly group together traces that came from the same source string. Given the ability to do such clustering, we can indeed use the [13] algorithm on each obtained cluster to identify each of the source strings which has non-negligible weight under the distribution $\mathcal{D}$, and given the identity of these source strings that each trace came from, it is straightforward to output a high-accuracy hypothesis for the unknown distribution $\mathcal{D}$ over these strings.

The core clustering procedure which we develop is a simple algorithm which we call A_{cluster} (see Algorithm 1 in Section 4). This algorithm takes two traces a and b as input, and outputs either "same" or "different." Its performance guarantee is the following: If a and b were generated as two independent traces from *the same* randomly chosen source string $\boldsymbol{x}$, then with high probability A_{cluster} outputs "same," whereas if a, b were generated as two traces coming from *two independent* uniform random source strings $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ respectively, then with high probability A_{cluster} outputs "different." (See Theorem 5 for a detailed statement.)

The idea underlying A_{cluster} is as follows. Given a trace a (which we view as a string over $\{-1, 1\}$), imagine breaking it up into contiguous segments (which we call "*blocks*") and summing the ± 1 bits within each block, and let $\text{sum}(a, i)$ denote the sum of the bits in the i-th block. We do the same for the trace b and obtaining a value $\text{sum}(b, i)$ from the i-th block of b. The high-level idea is that, for a suitable choice of the block size, in general there will be significant overlap between the positions in $\{1, \ldots, n\}$ (of the source string) that gave rise to the elements of the i-th block of a and the i-th block of b. As a result,

- On the one hand, if a and b came from the same source $\boldsymbol{x}$, then there will be significant cancellation in the difference $\text{sum}(a, i) - \text{sum}(b, i)$ and this difference will tend to be "small" in magnitude.
- On the other hand, if a and b came from independent source strings $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$, then there will be no such cancellation and the difference $\text{sum}(a, i) - \text{sum}(b, i)$ will not be so "small" in magnitude.

Therefore by checking the magnitude of $\text{sum}(a, i) - \text{sum}(b, i)$ across many different blocks i, it is possible to determine with high confidence whether or not a and b came from the same source string or not.

2 Preliminaries

We write $[n] = \{1, \ldots, n\}$ for a positive integer n. We index strings $x \in \{0, 1\}^n$ as $x = (x_1, \ldots, x_n)$. We use **bold font** to denote random variables (which may be real-valued, integer-valued, $\{0, 1\}^*$-valued, etc.).

We consider an insertion-deletion noise channel $\mathcal{C}_{q,q'}$ defined as by [13]. Given a *deletion rate* q and an *insertion rate* q', both in $[0, 1)$, the insertion-deletion channel $\mathcal{C}_{q,q'}$ acts on an $x \in \{0, 1\}^n$ as follows: First, for each $j \in [n]$, $\boldsymbol{G}_j(q') - 1$ many independent and uniform bits from $\{0, 1\}$ are inserted before the j-th bit of x, where $\boldsymbol{G}_1(q'), \ldots, \boldsymbol{G}_n(q')$ are i.i.d. geometric random variables satisfying

$$\mathbf{Pr}\left[\boldsymbol{G}_j(q') = \ell\right] = (q')^{\ell-1}(1 - q')$$

(i.e. each $\boldsymbol{G}_j(q')$ is distributed as Geometric$(1 - q')$). Then each bit of the resulting string is independently deleted with probability q. The resulting string is the output from $\mathcal{C}_{q,q'}(x)$, and we write "$\boldsymbol{y} \sim \mathcal{C}_{q,q'}(x)$" to indicate that $\boldsymbol{y}$ is a random trace generated from x in this way. If $\mathcal{D}$ is a distribution over n-bit strings, we write "$\boldsymbol{y} \sim \mathcal{C}_{q,q'}(\mathcal{D})$" to indicate that $\boldsymbol{y}$ is obtained by first drawing $\boldsymbol{x} \sim \mathcal{D}$ and then drawing $\boldsymbol{y} \sim \mathcal{C}_{q,q'}(\boldsymbol{x})$.

3 Achieving an arbitrarily small fraction of "hard" strings in average-case trace reconstruction

Fix any constants $q, q' \in [0,1)$ as deletion and insertion rates, receptively. We will use asymptotic notation such as $O(\cdot)$ and $\Theta(\cdot)$ to hide constants that depend on q and q'.

The main result of [13] is an algorithm which successfully performs trace reconstruction on at least $(1 - O(1)/n)$-fraction of all n-bit strings (which is $1 - M/n$ for some constant $M = M(q, q')$ that only depends on q and q'). In more detail, their main result is the following:

▶ **Theorem 2.** *Fix any constants $q, q' \in [0,1)$. There is a deterministic algorithm $A_{\text{average-case}}$ with the following property: It is given (1) a confidence parameter $\delta > 0$, (2) the length n of an unknown string $x \in \{0,1\}^n$ and (3) access to $\mathcal{C}_{q,q'}(x)$, uses*

$$\exp\left(O\left(\log^{1/3} n\right)\right) \cdot \log(1/\delta) \tag{1}$$

traces drawn from $\mathcal{C}_{q,q'}(x)$, and runs in time $\text{poly}(n, \log(1/\delta))$. For at least $(1 - O(1/n))$-fraction of all strings $x \in \{0,1\}^n$,[2] it is the case that, algorithm $A_{\text{average-case}}(\delta, n, \mathcal{C}_{q,q'}(x))$ outputs the string x with probability at least $1 - \delta$ (over the randomness of traces drawn from $\mathcal{C}_{q,q'}(x)$).

Note that in the above theorem the fraction of "hard" strings $x \in \{0,1\}^n$ on which the [13] algorithm does not succeed is $\Theta(1/n)$. In our setting, to achieve results for general population sizes s, we may require the fraction of "hard" strings on which the reconstruction algorithm does not succeed to be smaller than this; to see this, suppose for example that we are considering a population of size $s = n^2$. If a $\Theta(1/n)$ fraction of strings are "hard" and n^2 strings are chosen uniformly at random to form the support of our distribution $\mathcal{D}$, then we would expect $\Theta(n)$ many hard strings to be present in the support set (i.e. the population) of n^2 strings. If the unknown distribution over the n^2 strings (which, recall, may be any distribution over that support) puts a significant amount of its probability mass on these hard strings, then it may not be possible to successfully recover the population.

In this section we show that the fraction of strings in $\{0,1\}^n$ that are "hard" can be driven down from $\Theta(1/n)$ to an arbitrarily small fraction in the [13] result, at the cost of a corresponding modest increase in the sample complexity and running time of the algorithm. (As suggested by the discussion given above, such an extension is crucial for us to be able to handle populations of size $s = \omega(n)$.) It may be possible to verify this directly via a careful reworking of the [13] proof, but that proof is involved and such a verification would be quite tedious. Instead we give a simple and direct argument which uses Theorem 2 in a black-box way to prove the following generalization of it, in which only an arbitrarily small fraction of strings are hard to reconstruct:

▶ **Theorem 3.** *Fix any constants $q, q' \in [0,1)$. There is a deterministic algorithm $A'_{\text{average-case}}$ with the following property: It is given (1) $\tau > 0$ as the desired fraction of hard strings, (2) a confidence parameter δ, (3) the length n of the unknown string $x \in \{0,1\}^n$ and (4) access to $\mathcal{C}_{q,q'}(x)$. It uses*

$$\exp\left(O\left(\left(\log \max\{n, 1/\tau\}\right)^{1/3}\right)\right) \cdot \log(1/\delta)$$

[2] Theorem 1 of [13] only claims a $1 - o_n(1)$ fraction of strings x, but the proof shows that the fraction is $1 - O_{q,q'}(1)/n$; see e.g. the discussion at the beginning of Section 1.3 of [13].

many traces drawn from $\mathcal{C}_{q,q'}(x)$*, and runs in time* $\mathrm{poly}(\max\{n, 1/\tau\}, \log(1/\delta))$*. For at least* $1-\tau$ *fraction of all strings* $x \in \{0,1\}^n$*, it is the case that algorithm* $A'_{\textit{average-case}}(\tau, \delta, n, \mathcal{C}_{q,q'}(x))$ *outputs the string* x *with probability at least* $1-\delta$*.*

We note that the sample complexity of Theorem 3 interpolates smoothly between the average-case result of [13], in which a $\tau = \Theta(1/n)$ fraction of strings are hard, and the worst-case results of [8, 21], in which no strings in $\{0,1\}^n$ (equivalently, at most a $\tau = 1/2^{n+1}$ fraction of strings) are hard.

The high-level idea underlying Theorem 3 is very simple: By padding the input string x (which should be thought of as uniformly random over $\{0,1\}^n$) with random bits, it is possible to obtain a uniformly random N-bit string, and by running algorithm $A_{\text{average-case}}$ over this string of length N, with probability $1-\Theta(1/N)$ it is possible to reconstruct this N-bit string, from which the original input string x can be reconstructed. Taking N to be suitably large this yields the desired result. We give a detailed proof below.

Let M be the constant hidden in the $O(1/n)$ in Theorem 2. We note that if $\tau \geq M/n$ then we may simply use $A_{\text{average-case}}$, so we henceforth assume that $\tau < M/n$.

The algorithm. Algorithm $A'_{\text{average-case}}(\tau, \delta, n, \mathcal{C}_{q,q'}(x))$ works by running an auxiliary algorithm $A^*(\tau, n, \mathcal{C}_{q,q'}(x))$ (which always outputs an n-bit string) $O(\log(1/\delta))$ many times. If at least $9/16$ of the $O(\log(1/\delta))$ runs of A^* yield the same n-bit string then this is the output of $A'_{\text{average-case}}$, and otherwise $A'_{\text{average-case}}$ outputs "failure." Below we will show that for at least $1-\tau$ fraction of all strings $x \in \{0,1\}^n$, $A^*(\tau, n, \mathcal{C}_{q,q'}(x))$ outputs the correct string x with probability at least $5/8$. It follows from the Chernoff bound that $A'_{\text{average-case}}$ achieves the desired $1-\delta$ success probability.

We turn to describing and analyzing $A^*(\tau, n, \mathcal{C}_{q,q'}(x))$, which works as follows:

1. It draws a string $\boldsymbol{z}$ uniformly from $\{0,1\}^{N-n}$ (the value of $N > n$ will be specified later).
2. Let $m = m(N)$ be the following parameter:

$$m(N) = \exp\left(O\left(\log^{1/3} N\right)\right) \cdot \log\left(1/\delta'\right),$$

   where $\delta' = 1/8$. This is the number of traces needed by $A_{\text{average-case}}$ to achieve confidence parameter δ' on strings of length N (as in (1)). For m times, algorithm A^* independently repeats the following: at the i-th repetition it draws a string $\boldsymbol{y}^{(i)} \sim \mathcal{C}_{q,q'}(x)$, constructs a string $\boldsymbol{y}'^{(i)}$ that is distributed according to $\mathcal{C}_{q,q'}(\boldsymbol{z})$, and constructs $\boldsymbol{a}^{(i)} := \boldsymbol{y}^{(i)} \circ \boldsymbol{y}'^{(i)}$ which is the concatenation of $\boldsymbol{y}^{(i)}$ and $\boldsymbol{y}'^{(i)}$.
3. Finally, it uses the m strings $\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(m)}$ to run algorithm $A_{\text{average-case}}$ with length N and confidence parameter δ'. Let $w \in \{0,1\}^N$ be the string that $A_{\text{average-case}}$ returns. The output of A^* is $w_1 w_2 \cdots w_n$, the first n characters of w.

Proof of correctness. We first observe that (as an immediate consequence of the definition of the noise channel $\mathcal{C}_{q,q'}$) each string $\boldsymbol{a}^{(i)} = \boldsymbol{y}^{(i)} \circ \boldsymbol{y}'^{(i)}$ generated as in Step 2 of $A^*(\tau, n, \mathcal{C}_{q,q'}(x))$ is distributed precisely as a draw from $\mathcal{C}_{q,q'}(x \circ \boldsymbol{z})$. By the choice of $m = m(N)$ in Step 2, the strings $\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(m)}$ constitute precisely the required traces for a run of $A_{\text{average-case}}$ on the N-bit string $x \circ \boldsymbol{z}$.

Let us say that the strings in $\{0,1\}^N$ which $A_{\text{average-case}}$ (with parameters δ' and N) correctly reconstructs with probability at least $1-\delta'$ are *good* strings, and that the other strings in $\{0,1\}^N$ are *bad* strings. By Theorem 2, at most an (M/N)-fraction of all strings in $\{0,1\}^N$ are bad. The value of N is set to $N := 4M/\tau$,[3] so $M/N = \tau/4$, and it is the case

[3] Note that $N \geq 4n$ using $\tau < M/N$, so $N - n > 0$ and indeed Step 1 makes sense.

that at most a $\tau/4$ fraction of strings in $\{0,1\}^N$ are bad. For each $x \in \{0,1\}^n$, let γ_x denote the fraction of strings $z \in \{0,1\}^{N-n}$ such that $x \circ z$ is bad. The average over all $x \in \{0,1\}^n$ of γ_x is at most $\tau/4$, and consequently at most a τ fraction of strings x have $\gamma_x \geq 1/4$.

▷ Claim 4. If $x \in \{0,1\}^n$ has $\gamma_x < 1/4$, then $A^*(\tau, n, \mathcal{C}_{q,q'}(x))$ outputs x with probability at least $5/8$.

Proof. The probability that $\boldsymbol{z} \sim \{0,1\}^{N-n}$ such that $x \circ \boldsymbol{z}$ is good is at least $3/4$. If $x \circ \boldsymbol{z}$ is good then with probability at least $1 - \delta' = 7/8$ the output of $A_{\text{average-case}}$ as run in Step 3 is the string $x \circ \boldsymbol{z}$ and hence the output of A^* is x. The claim follows since $(3/4) \cdot (7/8) > 5/8$. ◁

Hence for at least a $(1-\tau)$-fraction of $x \in \{0,1\}^n$, a run of $A^*(\tau, n, \mathcal{C}_{q,q'}(x))$ outputs x with probability at least $5/8$. For any such x, a simple Chernoff bound shows that with probability at least $1 - \delta$, at least $9/16$ of the $O(\log(1/\delta))$ many independent runs of A^* will output x. This concludes the proof of Theorem 3. ◀

4 The core clustering result

In this section we state and prove the key clustering result that is used in the main algorithm. Intuitively, it gives an efficient procedure with the following performance guarantee: Given two traces, the procedure can determine with high probability whether the two traces were both obtained as traces from the same uniform random string $\boldsymbol{x} \sim \{0,1\}^n$, or the two traces were obtained from two independent uniform random strings $\boldsymbol{x}^1, \boldsymbol{x}^2 \sim \{0,1\}^n$.

In more detail, the main result of this section is the following theorem:

▶ **Theorem 5.** *Fix any constants $q, q' \in [0,1)$. There is a deterministic algorithm $A_{cluster}$ with the following performance guarantee: It is given a positive integer n and a pair of binary strings z and z'. Let $\delta_{cluster} := \exp(-\Theta(n^{1/3}))$. Then $A_{cluster}(n, z, z')$ runs in time $O(n)$ and satisfies the following two properties:*

1. *Suppose that $\boldsymbol{x}$ is uniform random over $\{0,1\}^n$ and $\boldsymbol{z}, \boldsymbol{z}'$ are independent draws from $\mathcal{C}_{q,q'}(\boldsymbol{x})$. Then with probability at least $1 - \delta_{cluster}$, algorithm $A_{cluster}(n, \boldsymbol{z}, \boldsymbol{z}')$ outputs "same."*
2. *Suppose that $\boldsymbol{x}^1, \boldsymbol{x}^2$ are independent uniform random strings over $\{0,1\}^n$, $\boldsymbol{z} \sim \mathcal{C}_{q,q'}(\boldsymbol{x}^1)$ and $\boldsymbol{z}' \sim \mathcal{C}_{q,q'}(\boldsymbol{x}^2)$. Then with probability at least $1 - \delta_{cluster}$, $A_{cluster}(n, \boldsymbol{z}, \boldsymbol{z}')$ outputs "different."*

4.1 Proof of Theorem 5

For convenience, we consider strings over $\{-1,1\}$ instead of $\{0,1\}$ in the rest of this section. We need the following technical lemma:

▶ **Lemma 6.** *Let $\tau \in (0,1]$ be a constant. Then there exist three positive constants c_1, c_2 and c_3 (that only depend on τ) such that the following property holds. For all positive integers m and m' such that $m' \leq (1-\tau)m$ and m is sufficiently large, letting $\mathbf{X}_1, \ldots, \mathbf{X}_m$ be independent and uniform random variables over $\{-1,1\}$, we have*

$$\boldsymbol{Pr}\left[\left|\mathbf{X}_1 + \cdots + \mathbf{X}_m\right| \geq c_1\sqrt{m}\right] \geq c_2 + c_3 \quad \text{and} \quad \boldsymbol{Pr}\left[\left|\mathbf{X}_1 + \cdots + \mathbf{X}_{m'}\right| \geq c_1\sqrt{m}\right] \leq c_2 - c_3.$$

Proof. The Berry-Esseen theorem (see e.g. [11]) establishes closeness between the cdf of a sum of "well-behaved" independent random variables (such as $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_m$) and the cdf of a Normal distribution with the same mean and variance. By the Berry-Esseen theorem, the probability of $|\mathbf{X}_1 + \cdots + \mathbf{X}_m| \geq c_1\sqrt{m}$ is within an additive $\pm o_m(1)$ of the corresponding probability of $|\boldsymbol{G}_1| \geq c_1\sqrt{m}$, where $\boldsymbol{G}_1 \sim \mathcal{N}(0, m)$.

We first consider the case that m' is not too small compared to m, say $m' > m^{1/3}$. In this case the Berry-Esseen theorem implies that the probability of $|\mathbf{X}_1 + \cdots + \mathbf{X}_{m'}| \geq c_1\sqrt{m}$ is also within an additive $\pm o_m(1)$ of the corresponding probability for Gaussian random variables, which is now $\mathbf{Pr}[|\boldsymbol{G}_2| \geq c_1\sqrt{m}]$ with $\boldsymbol{G}_2 \sim \mathcal{N}(0, m')$. So in this case Lemma 6 is an immediate consequence of an analogous statement for Gaussian random variables,

$$\mathbf{Pr}\left[\left|\boldsymbol{G}_1\right| \geq c_1\sqrt{m}\right] \geq c_2 + c_3 \quad \text{and} \quad \mathbf{Pr}\left[\left|\boldsymbol{G}_2\right| \geq c_1\sqrt{m}\right] \leq c_2 - c_3, \tag{2}$$

where $\boldsymbol{G}_1 \sim \mathcal{N}(0, m)$ and $\boldsymbol{G}_2 \sim \mathcal{N}(0, m')$. The first probability in (2) is the probability that a Gaussian's magnitude exceeds its mean by at least c_1 standard deviations, while the second probability in (2) is the probability that a Gaussian's magnitude exceeds its mean by at least $c_1/\sqrt{1-\tau}$ standard deviations. Given this, for suitable c_1, c_2, c_3 depending only on τ, the inequalities (2) are a straightforward consequence of the following standard bounds on the cdf of a Gaussian $\boldsymbol{G} \sim \mathcal{N}(0, \sigma^2)$ [[11], Section 7.1]:

$$\left(\frac{1}{x} - \frac{1}{x^3}\right) \cdot \frac{e^{-x^2/2}}{\sqrt{2\pi}} \leq \mathbf{Pr}\left[\boldsymbol{G} \geq x\sigma\right] \leq \frac{1}{x} \cdot \frac{e^{-x^2/2}}{\sqrt{2\pi}}, \quad \text{for all } x > 0.$$

Finally we consider the case that m' is very small compared to m, say $m' \leq m^{1/3}$. In this case, by the Berry-Esseen theorem we have that $\mathbf{Pr}[|\mathbf{X}_1 + \cdots + \mathbf{X}_m| \geq c_1\sqrt{m}]$ is $\pm o_m(1)$-close to the probability that a Gaussian's magnitude exceeds its mean by c_1 standard deviations, which is at least some absolute constant, while $\mathbf{Pr}[|\mathbf{X}_1 + \cdots + \mathbf{X}_{m'}| \geq c_1\sqrt{m}]$ is zero for sufficiently large m, because $m' \leq m^{1/3} < c_1\sqrt{m}$ for m sufficiently large. This finishes the proof of Lemma 6. ◀

Recall that constants $q, q' \in [0, 1)$ denote the deletion probability and insertion probability respectively. Let $p, p' \in (0, 1]$ be $p = 1 - q$ and $p' = 1 - q'$. Then the expected length of a string drawn from $\mathcal{C}_{q,q'}(x)$ with $x \in \{-1, 1\}^n$ is $np/p' = \alpha n$, where $\alpha := p/p'$ is a positive constant.

Given a string $x \in \{-1, 1\}^n$, we start by describing an equivalent way of drawing $\boldsymbol{z} \sim \mathcal{C}_{q,q'}(x)$. We say a string r over $[n] \cup \{*\}$ is an *n-pattern* if every $i \in [n]$ appears in r at most once and integers appear in r in ascending order. We write $\mathcal{R}_{n,q,q'}$ to denote the following distribution over n-patterns. To draw $\mathbf{r} \sim \mathcal{R}_{n,q,q'}$ we start with $r^{(0)} = (1, 2, \ldots, n)$. Then for each $j \in [n]$, $\boldsymbol{G}_j(q') - 1$ many $*$'s are inserted before the j-th entry (with value j) of $r^{(0)}$ to obtain $\mathbf{r}^{(1)}$. Finally each entry of $\mathbf{r}^{(1)}$ is independently deleted with probability q to obtain the final string $\mathbf{r}$. Using $\mathcal{R}_{n,q,q'}$, drawing $\boldsymbol{z} \sim \mathcal{C}_{q,q'}(x)$ can be done equivalently as follows:

1. Draw an n-pattern $\mathbf{r} \sim \mathcal{R}_{n,q,q'}$.
2. For each index $i \in [n]$ that appears in $\mathbf{r}$, replace it by x_i in $\mathbf{r}$.
3. Replace each $*$ in $\mathbf{r}$ with an independent and uniform draw from $\{-1, 1\}$.

Next we introduce a number of parameters and constants that will be used in the clustering algorithm A_{cluster}. Two parameters $\tilde{s}$ and t used in the algorithm are

$$t := n^{2/3} \quad \text{and} \quad \tilde{s} = \left\lfloor \frac{\alpha n}{4t} \right\rfloor = \Theta(n^{1/3})$$

so that $2\tilde{s}t \le \alpha n/2$. Three constants β, γ and δ are defined as c_1, c_2 and c_3 in Lemma 6 with τ set to be the following constant in $(0,1)$: $\tau = 0.7pp'$. For each $\ell \in [\tilde{s}]$, we let I_ℓ denote the following set of integers:

$$I_\ell = \big[(2\ell-2)t+1, (2\ell-1)t\big] \cap \mathbb{Z}. \tag{3}$$

Given a string z over $\{-1,1\}$ (or an n-pattern r), we will refer to entries z_i of z (or r_i of r) over $i \in I_\ell$ as the ℓ-th *block* of z (or r). So each block consists of t entries and two consecutive blocks are separated by a gap of t entries. Given an n-pattern r and an integer ℓ, we write $B_\ell(r) \subset [n]$ to denote the set of $i \in [n]$ that appears in the ℓ-th block of r.

The algorithm A_{cluster} is described in Algorithm 1. Before stating the key technical lemma (Lemma 7) and using it to prove Theorem 5, we give some intuition for the algorithm A_{cluster}.

Algorithm 1 Description of the clustering algorithm A_{cluster}.

Algorithm $A_{\text{cluster}}(n, z, z')$

Input: A positive integer n and two strings z and z' over $\{-1,1\}$.

Output: "Same" or "different."

1. For each $\ell \in [\tilde{s}]$, set Z_ℓ to be the sum of z_i over $i \in I_\ell$, with $z_i = 0$ when $i > |z|$.
2. For each $\ell \in [\tilde{s}]$, set Z'_ℓ to be the sum of z'_i over $i \in I_\ell$, with $z'_i = 0$ when $i > |z'|$.
3. Count the number of $\ell \in [\tilde{s}]$ such that $\left|Z_\ell - Z'_\ell\right| \ge \beta\sqrt{2t}$.
4. If the number of such ℓ is at least $\gamma\tilde{s}$, return "different;" otherwise, return "same."

Recall the two cases in Theorem 5. We start with the easier second case, where $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ are drawn from $\{-1,1\}^n$ uniformly and independently, $\boldsymbol{z} \sim \mathcal{C}_{q,q'}(\boldsymbol{x}^1)$ and $\boldsymbol{z}' \sim \mathcal{C}_{q,q'}(\boldsymbol{x}^2)$. First it is easy to show (see property (0) of Lemma 7) that $|\boldsymbol{z}|, |\boldsymbol{z}'| \ge \alpha n/2$ with very high probability. When this happens, $\boldsymbol{Z}_\ell$ is the sum of t independent and uniform random variables over $\{-1,1\}$ and the same holds for $\boldsymbol{Z}'_\ell$. Moreover, $\boldsymbol{Z}_\ell$ and $\boldsymbol{Z}'_\ell$ are independent of each other since $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ are drawn independently and thus, $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ can be equivalently written as the sum of $2t$ independent and uniform variables over $\{-1,1\}$. Furthermore, the $\tilde{s}$ random variables $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ over $\ell \in [\tilde{s}]$ are independent. Thus, it follows from Lemma 6 and our choices of β, γ and δ that the probability of each $|\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell| \ge \beta\sqrt{2t}$ is at least $\gamma + \delta$ and with very high probability, the number of such $\ell \in [\tilde{s}]$ is at least $\gamma\tilde{s}$, in which case the algorithm returns "different" as desired.

In the first case of Theorem 5, we draw $\boldsymbol{x}$ from $\{-1,1\}^n$ uniformly and then draw $\boldsymbol{z}, \boldsymbol{z}'$ from $\mathcal{C}_{q,q'}(\boldsymbol{x})$ independently. Equivalently one can view the process as first drawing two n-patterns $\mathbf{r}$ and $\mathbf{r}'$ independently from $\mathcal{R}_{n,q,q'}$ and $\boldsymbol{x}$ from $\{-1,1\}^n$. The string $\boldsymbol{z}$ (or $\boldsymbol{z}'$) is then obtained by replacing each $i \in [n]$ in $\mathbf{r}$ (or $\mathbf{r}'$) by $\boldsymbol{x}_i$ and each $*$ by an independent draw from $\{-1,1\}$. Again we assume that $|\mathbf{r}|, |\mathbf{r}'| \ge \alpha n/2$, which happens with high probability. When this is the case, each of $\boldsymbol{Z}_\ell$ and $\boldsymbol{Z}'_\ell$ for $\ell \in [\tilde{s}]$ remains the sum of t independent uniform random variables over $\{-1,1\}$. However, when an index $i \in [n]$ appears in the ℓ-th block of both $\mathbf{r}$, $\mathbf{r}'$, then $\boldsymbol{x}_i$ appears in both sums $\boldsymbol{Z}_\ell, \boldsymbol{Z}'_\ell$ and gets cancelled out in their difference $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$.

Our main technical lemma shows that with very high probability over draws $\mathbf{r}$ and $\mathbf{r}'$ from $\mathcal{R}_{n,q,q'}$, the following two properties hold: (1) $|B_\ell(\mathbf{r}) \cap B_\ell(\mathbf{r}')| \ge \tau t$ for every $\ell \in [\tilde{s}]$, i.e., there are at least τt many integers that appear in the ℓ-th block of both $\mathbf{r}$ and $\mathbf{r}'$; and (2) No index $i \in [n]$ appears in two different blocks of $\mathbf{r}$ and $\mathbf{r}'$ (i.e., it cannot be the case that both $i \in B_\ell(\mathbf{r})$ and $i \in B_{\ell'}(\mathbf{r}')$ with $\ell \ne \ell'$; intuitively the reason why we leave a gap of t entries between two consecutive blocks is to achieve this property). Fixing such a pair of n-patterns r and r', property (1) implies that each $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ can be written as the sum of

at most $(1-\tau)2t$ many independent $\{-1,1\}$-variables; given this, it follows directly from Lemma 6 that the probability of each $|\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell| \geq \beta\sqrt{2t}$ is at most $\gamma - \delta$. Furthermore (2) implies that the $\tilde{s}$ variables $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ over $\ell \in [\tilde{s}]$ are independent. This lets us easily infer that the number of ℓ such that $|\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell| \geq \beta\sqrt{2t}$ is less than $\gamma\tilde{s}$ with very high probability, in which case the algorithm returns "same" as desired.

As discussed above, the main technical lemma we require is as follows:

▶ **Lemma 7.** *Let $\mathbf{r}, \mathbf{r}'$ be two n-patterns drawn independently from $\mathcal{R}_{n,q,q'}$. Then with probability at least $1 - \exp(-\Omega(n^{1/3}))$, the following three properties all hold:*
(0): $|\mathbf{r}|, |\mathbf{r}'| \geq \alpha n/2$.
(1): $\left|B_\ell(\mathbf{r}) \cap B_\ell(\mathbf{r}')\right| \geq \tau t$ *for all* $\ell \in [\tilde{s}]$.
(2): *If an* $i \in [n]$ *appears in both* $B_\ell(\mathbf{r})$ *and* $B_{\ell'}(\mathbf{r}')$ *for some* $\ell, \ell' \in [\tilde{s}]$, *then we have* $\ell = \ell'$.

The detailed proof of Lemma 7 is given in Appendix A; here we give some intuition.

To prove Lemma 7, we show that $\mathbf{r}, \mathbf{r}' \sim \mathcal{R}_{n,q,q'}$ satisfy each of the three properties with probability at least $1 - \exp(-\Omega(n^{1/3}))$; the lemma follows from a union bound. Property (0) follows from tail bounds on sums of independent Geometric random variables and from standard Chernoff bounds (see Claim 12) for insertions and deletions, respectively. Indeed property (0) holds with probability $1 - \exp(-\Omega(n))$.

Properties (1) and (2) follow from Lemma 13 in Appendix A. To state the lemma, recall that we write $\mathbf{r}^{(1)}$ to denote the string over $[n] \cup \{*\}$ obtained after insertions during the generation of $\mathbf{r} \sim \mathcal{R}_{n,q,q'}$. For each $i \in [n]$, we use $\boldsymbol{Y}_i$ to denote the number of characters before i in $\mathbf{r}^{(1)}$ that survive deletions; note that $\boldsymbol{Y}_i$ is the number of characters that appear before i in $\mathbf{r}$ if i survives in $\mathbf{r}$, but $\boldsymbol{Y}_i$ is well defined even if i was deleted. By definition, we have $\mathbf{E}[\boldsymbol{Y}_i] = ((i/p') - 1)p$. Lemma 13 shows that for constant $c \in (0,1)$, with probability $1 - \exp(-\Omega(n^{1/3}))$, $|\boldsymbol{Y}_i - \mathbf{E}[\boldsymbol{Y}_i]| \leq ct$. Lemma 13 again follows from tail bounds on sums of independent Geometric random variables and from standard Chernoff bounds. We define $\boldsymbol{Y}'_i$ similarly for $\mathbf{r}'$ and the same statement also holds for $\boldsymbol{Y}'_i$.

Property (2) follows directly from Lemma 13, since for an $i \in [n]$ to appear in two different blocks, it must be the case that $|\boldsymbol{Y}_i - \boldsymbol{Y}'_i| \geq t$ and thus, either $|\boldsymbol{Y}_i - \mathbf{E}[\boldsymbol{Y}_i]| \geq t/2$ or $|\boldsymbol{Y}'_i - \mathbf{E}[\boldsymbol{Y}'_i]| \geq t/2$ (as we have $\mathbf{E}[\boldsymbol{Y}'_i] = \mathbf{E}[\boldsymbol{Y}_i]$), which happens with probability at most $\exp(-\Omega(n^{1/3}))$ by Lemma 13.

To prove property (1) for $\ell \in [\tilde{s}]$, we focus on the following interval of indices in $[n]$:

$$I_\ell^{(0)} := \left[\frac{p'}{p}(2\ell - 1.9)t + 1, \frac{p'}{p}(2\ell - 1.1)t\right] \cap \mathbb{Z},$$

and show that with probability at least $1 - \exp(-\Omega(n^{1/3}))$, we have both
(a) At least $\tau t = 0.7tpp'$ indices in $I_\ell^{(0)}$ survive in both $\mathbf{r}$ and $\mathbf{r}'$; and
(b) Every $i \in I_\ell^{(0)}$ that survives in both $\mathbf{r}$ and $\mathbf{r}'$ lies in both $B_\ell(\mathbf{r})$ and $B_\ell(\mathbf{r}')$.
Item (a) follows from a Chernoff bound: the length of $I_\ell^{(0)}$ is $0.8tp'/p$ and every element survives independently in both strings with probability p^2. Letting i_0, i_1 be the left and right ends of $I_\ell^{(0)}$, item (b) holds when $\boldsymbol{Y}_{i_0}, \boldsymbol{Y}'_{i_0}, \boldsymbol{Y}_{i_1}, \boldsymbol{Y}'_{i_1}$ do not shift too far $(0.1t)$ away from their expectations which happens with probability at least $1 - \exp(-\Omega(n^{1/3}))$ by Lemma 13.

Finally we use Lemma 7 to prove Theorem 5:

Proof of Theorem 5. We start with the second case in which $\boldsymbol{x}^1, \boldsymbol{x}^2$ are independent uniform random strings over $\{0,1\}^n$, $\boldsymbol{z} \sim \mathcal{C}_{q,q'}(\boldsymbol{x}^1)$ and $\boldsymbol{z}' \sim \mathcal{C}_{q,q'}(\boldsymbol{x}^2)$. By our discussion earlier, $\boldsymbol{z}$ and $\boldsymbol{z}'$ can be generated equivalently by first drawing $\boldsymbol{r}, \boldsymbol{r}' \sim \mathcal{R}_{n,q,q'}$, then drawing $\boldsymbol{x}^1, \boldsymbol{x}^2$,

and finally deriving $\boldsymbol{z}$ (or $\boldsymbol{z}'$) from $\boldsymbol{r}$ (or $\boldsymbol{r}'$) using $\boldsymbol{x}^1$ (or $\boldsymbol{x}^2$) as well as independent random bits for the $*$'s. By Lemma 7, $\boldsymbol{r}$ and $\boldsymbol{r}'$ satisfy all three properties with probability at least $1-\exp(-\Omega(n^{1/3}))$. Fixing r and r' that satisfy all three properties (for the first case we only need property (0)), we show that $A_{\text{cluster}}(n, \boldsymbol{z}, \boldsymbol{z}')$ returns "different" with probability at least $1-\exp(-\Omega(n^{1/3}))$ conditioning on $\boldsymbol{r}=r$ and $\boldsymbol{r}'=r'$; the lemma for this case then follows.

To this end, it follows from property (0) that each $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ is a sum of $2t$ independent uniform random variables over $\{-1,1\}$ and thus, each $\ell \in [\tilde{s}]$ satisfies $|\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell| \geq \beta\sqrt{2t}$ with probability at least $\gamma+\delta$. Moreover, the $\tilde{s}$ variables $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ are independent. It follows from a Chernoff bound (and that δ is a positive constant) that A_{cluster} returns "different" with probability $1-\exp(-\Omega(\tilde{s})) = 1-\exp(-\Omega(n^{1/3}))$.

For the second case we can similarly generate $\boldsymbol{z}, \boldsymbol{z}'$ by first drawing $\mathbf{r}, \mathbf{r}' \sim \mathcal{R}_{n,q,q'}$, then drawing $\boldsymbol{x}$, and finally deriving $\boldsymbol{z}, \boldsymbol{z}'$ from $\mathbf{r}, \mathbf{r}'$ using the same $\boldsymbol{x}$ and independent random bits for the $*$'s. Again it follows from Lemma 7 that $\mathbf{r}, \mathbf{r}'$ satisfy all three properties with probability $1-\exp(-\Omega(n^{1/3}))$. Fixing r, r' that satisfy all three properties, we show that $A_{\text{cluster}}(n, \boldsymbol{z}, \boldsymbol{z}')$ returns "same" with probability $1-\exp(-\Omega(n^{1/3}))$, conditioning on $\boldsymbol{r}=r$ and $\boldsymbol{r}'=r'$; the lemma for this case then follows.

For this purpose, properties (0) and (1) imply that each $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ is the sum of at most $(1-\tau)2t$ many independent uniform random variables over $\{-1,1\}$. Lemma 6 implies that the probability of $|\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell| \geq \beta\sqrt{2t}$ is at most $\gamma-\delta$. Moreover, property (2) implies that these $\tilde{s}$ variables $\boldsymbol{Z}_\ell - \boldsymbol{Z}'_\ell$ are independent. It similarly follows from a Chernoff bound that A_{cluster} returns "same" with probability $1-\exp(-\Omega(\tilde{s})) = 1-\exp(-\Omega(n^{1/3}))$. ◀

5 Putting the pieces together: Proof of Theorem 1

In this section we combine the main results from earlier sections, Theorem 3 from Section 3 and Theorem 5 from Section 4, together with standard results on learning discrete distributions, to prove Theorem 1.

5.1 Learning discrete distributions

We recall the following folklore result on learning a discrete distribution from independent samples:

▶ **Theorem 8.** *Fix $\gamma, \kappa > 0, N \in \mathbb{N}$. Let $\mathcal{P}$ be an unknown probability distribution over the discrete set $\{1,\ldots,N\}$, and let $\boldsymbol{S} = \{\boldsymbol{i}_1, \ldots, \boldsymbol{i}_m\}$ be independent draws from $\mathcal{P}$, where $m = O((N/\kappa^2)\cdot\log(1/\gamma))$. Let $\hat{\mathcal{P}}_{\boldsymbol{S}}$ denote the empirical probability distribution over $[N]$ corresponding to $\boldsymbol{S}$. Then with probability at least $1-\gamma$ over the draw of $\boldsymbol{S}$, the variation distance $d_{\text{TV}}(\hat{\mathcal{P}}_{\boldsymbol{S}}, \mathcal{P})$ is at most κ.*

We will need a corollary which says that removing low-frequency elements has only a negligible effect:

▶ **Corollary 9.** *Let $\mathcal{P}, m$ and $\boldsymbol{S}$ be as above. Let $\boldsymbol{S}'$ be the subset of $\boldsymbol{S}$ obtained by removing each element j whose frequency in $\boldsymbol{S}$ is at most $\kappa/(2N)$, and let $\hat{\mathcal{P}}_{\boldsymbol{S}'}$ denote the empirical distribution over $[N]$ corresponding to $\boldsymbol{S}'$. Then with probability at least $1-\gamma$ over the draw of $\boldsymbol{S}'$, $d_{\text{TV}}(\hat{\mathcal{P}}_{\boldsymbol{S}'}, \mathcal{P})$ is at most κ.*

Proof. By Theorem 8, with probability at least $1-\delta$ the hypothesis $\hat{\mathcal{P}}_{\boldsymbol{S}}$ from Theorem 8 is $\kappa/2$-close to $\mathcal{P}$. The corollary follows since the variation distance between $\hat{\mathcal{P}}_{\boldsymbol{S}}$ and $\hat{\mathcal{P}}_{\boldsymbol{S}'}$ is at most $N\cdot\kappa/(2N) = \kappa/2$. ◀

Algorithm 2 Description of the main algorithm A.

Algorithm $A(n, s, \varepsilon, \delta_{\text{hard}}, \delta_{\text{fail}}, \mathcal{C}_{q,q'}(\mathcal{D}))$, with constants $q, q' \in [0, 1)$ as deletion and insertion rates.

Input: String length n, support size $s \leq \exp(\Theta(n^{1/3}))$, accuracy parameter $\varepsilon \geq \exp(-\Theta(n^{1/3}))$, fraction of hard support sets $\delta_{\text{hard}} \geq \exp(-\Theta(n^{1/3}))$, failure probability $\delta_{\text{fail}} \geq \exp(-\Theta(n^{1/3}))$, and access to $\mathcal{C}_{q,q'}(\mathcal{D})$ where $\mathcal{D}$ is a probability distribution over s strings in $\{0,1\}^n$.

Output: Either a probability distribution $\mathcal{D}'$ or "fail."

1. Draw T traces $\boldsymbol{y}^1, \ldots, \boldsymbol{y}^T$ from $\mathcal{C}_{q,q'}(\mathcal{D})$, where
$$T = \frac{s}{\varepsilon^2} \cdot \exp\left(\Theta\left(\left(\log \max\left\{n, \frac{2s}{\delta_{\text{hard}}}\right\}\right)^{1/3}\right)\right) \cdot \log\left(\frac{3s}{\delta_{\text{fail}}}\right).$$
2. For each pair of traces $\boldsymbol{y}^i, \boldsymbol{y}^j$ with $1 \leq i < j \leq T$, run $A_{\text{cluster}}(n, \boldsymbol{y}^i, \boldsymbol{y}^j)$ from Section 4. If the $\binom{T}{2}$-many outcomes of A_{cluster} (corresponding to $\binom{T}{2}$ many answers of "same" or "different") do not correspond to a disjoint union of cliques then halt and output "fail," otherwise continue.
3. Let the resulting clusters / cliques be denoted $C_1, \ldots, C_r$, so $C_1 \sqcup \cdots \sqcup C_r$ is a partition of the set $\{\boldsymbol{y}^1, \ldots, \boldsymbol{y}^T\}$ of traces.[a] Call C_i *large* if it contains at least $T \cdot (\varepsilon/(2s))$ many elements. Let $C'_1, \ldots, C'_{r'}$ denote the large clusters for some $r' \leq r$, and let $C'_{\text{total}} = \sum_i |C'_i|$.
4. For each large multiset C'_i, run $A'_{\text{average-case}}$ from Section 3 using n and strings from C'_i, in which τ is set to $\delta_{\text{hard}}/(2s)$ and δ is set to $\delta_{\text{fail}}/(3s)$. Let z^i be the output of $A'_{\text{average-case}}$ on this input.
5. Distribution $\mathcal{D}'$ that A outputs is supported on $z^1, \ldots, z^{r'}$ and puts weight $|C'_i|/C'_{\text{total}}$ on z^i.

[a] Strictly speaking, each C_i is a multiset.

5.2 Proof of Theorem 1

Algorithm A is given in Algorithm 2. Its proof of correctness is given below.

Proof. Suppose that the true underlying support of $\mathcal{D}$ is $\mathcal{X} = (x^1, \ldots, x^s)$ (as an ordered list). We consider s instances of algorithm $A'_{\text{average-case}}$ from Section 3, where each instance has parameters n, $\tau = \delta_{\text{hard}}/(2s)$ and $\delta = \delta_{\text{fail}}/(3s)$, and the i-th one runs on T^* many traces drawn from $\mathcal{C}_{q,q'}(x^i)$, where

$$T^* = \exp\left(\Theta\left(\left(\log \max\left\{n, \frac{2s}{\delta_{\text{hard}}}\right\}\right)^{1/3}\right)\right) \cdot \log\left(\frac{3s}{\delta_{\text{fail}}}\right)$$

as specified in Section 3 (so we have $T = (s/\varepsilon^2) \cdot T^*$). We say that $\mathcal{X}$ is a *hard support* if either

(a) At least one string x^i, $i \in [s]$, is hard for algorithm $A'_{\text{average-case}}$; or

(b) After drawing T traces from $\mathcal{C}_{q,q'}(x^{(i)})$ for each $i \in [s]$, A_{cluster} fails on one of these $\binom{sT}{2}$ many pairs of traces with probability at least $\delta_{\text{fail}}/3$.

We consider a random support $\boldsymbol{\mathcal{X}} = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^s)$ drawn from $\{0,1\}^n$ independently and uniformly. Theorem 3 says the probability of a uniform random string being hard for $A'_{\text{average-case}}$ is at most $\delta_{\text{hard}}/(2s)$. A union bound says the probability our support

satisfies (a) is at most $\delta_{\text{hard}}/2$. On the other hand, for each support $\mathcal{X}$, we let $\lambda(\mathcal{X})$ denote the probability that A_{cluster} fails on at least one of the $\binom{sT}{2}$ pairs. Theorem 5 implies $\mathbf{E}_{\boldsymbol{\mathcal{X}}}[\lambda(\boldsymbol{\mathcal{X}})] \leq \binom{sT}{2} \cdot \delta_{\text{cluster}} \leq (\delta_{\text{fail}}/3) \cdot \delta_{\text{hard}}/2$, where the last inequality follows by setting the constant hidden in the $\Theta(n^{1/3})$ of upper and lower bounds for $s, \varepsilon, \delta_{\text{hard}}$ and δ_{fail} to be sufficiently small (compared to the constant hidden in δ_{cluster}). By Markov, a random support satisfies (b) with probability at most $\delta_{\text{hard}}/2$. A union bound on (a) and (b) says the probability of a random support being hard is at most δ_{hard}.

If $\mathcal{D}'$ is a probability distribution where $d_{\text{TV}}(\mathcal{D}, \mathcal{D}') \leq \varepsilon$, then we say that $\mathcal{D}'$ is ε-accurate. It suffices to show that for a support $\mathcal{X}$ that is not hard and an arbitrary distribution $\mathcal{D}$ on that support set, the probability that our algorithm A fails to output an ε-accurate distribution $\mathcal{D}'$ is at most δ_{fail}.

Our algorithm has three points of failure. In Step 2, it could fail to cluster the T traces correctly. Given the correct clustering in Step 2, it could fail to learn the underlying string for some cluster in Step 4. Finally, given the correct support, it could fail to output an ε-accurate distribution $\mathcal{D}'$ in Step 5.

By the definition of hard supports we have that Step 2 returns an incorrect clustering with probability at most $\delta_{\text{fail}}/3$. Given a correct clustering in Step 2, each large C_i' will have at least $T \cdot (\varepsilon/2s) = T^*/\varepsilon \geq T^*$ elements. Since no x^i is hard for $A'_{\text{average-case}}$, by Theorem 3 the probability any instance of $A'_{\text{average-case}}$ fails is at most $\delta_{\text{fail}}/(3s)$. By a union bound, the probability of a Step 4 error is at most $\delta_{\text{fail}}/3$.

Since $T \geq \Omega((s/\varepsilon^2) \cdot \log(3/\delta_{\text{fail}}))$ and the large clusters are defined to have size at least a $\varepsilon/2s$ fraction of the number of traces, then by Corollary 9 with $N = s$, $\kappa = \varepsilon$, $\gamma = \delta_{\text{fail}}/3$, and $m = T$, given the correct support the probability that Step 5 fails to output an ε-accurate probability distribution is at most $\delta_{\text{fail}}/3$. By a union bound, the probability of failure on a support that is not hard is at most δ_{fail}.

By Theorem 5 Step 2 takes time $O(nT^2)$. By Theorem 3 Step 4 takes time $\text{poly}(n, s/\delta_{\text{hard}}, \log(1/\delta_{\text{fail}}))$. Step 5 takes time $O(s)$ to compute the weights used in $\mathcal{D}'$. Therefore, the overall running time of the algorithm is $\text{poly}(n, s, 1/\varepsilon, 1/\delta_{\text{hard}}, \log(1/\delta_{\text{fail}}))$. The theorem follows since the sample complexity T is at most $\text{poly}(s, 1/\varepsilon, \exp(\log^{1/3} n), \exp(\log^{1/3}(1/\delta_{\text{hard}})), \log(1/\delta_{\text{fail}}))$. ◀

References

1 Alexandr Andoni, Mark Braverman, and Avinatan Hassidim. Phylogenetic Reconstruction with Insertions and Deletions. Manuscript, 2014.

2 Alexandr Andoni, Constantinos Daskalakis, Avinatan Hassidim, and Sébastien Roch. Global Alignment of Molecular Sequences via Ancestral State Reconstruction. In *ICS*, pages 358–369, 2010.

3 Frank Ban, Xi Chen, Adam Freilich, Rocco A. Servedio, and Sandip Sinha. Beyond trace reconstruction: Population recovery from the deletion channel. *CoRR*, abs/1904.05532, 2019. `arXiv:1904.05532`.

4 T. Batu, S. Kannan, S. Khanna, and A. McGregor. Reconstructing strings from random traces. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004*, pages 910–918, 2004.

5 Zachary Chase. New lower bounds for trace reconstruction. *arXiv preprint*, 2019. `arXiv:1905.03031`.

6 Constantinos Daskalakis and Sébastien Roch. Alignment-Free Phylogenetic Reconstruction. In *RECOMB*, pages 123–137, 2010.

7 A. De, M. Saks, and S. Tang. Noisy population recovery in polynomial time. Technical Report TR-16-026, Electronic Colloquium on Computational Complexity, 2016. To appear in FOCS 2016.

8 Anindya De, Ryan O'Donnell, and Rocco A. Servedio. Optimal mean-based algorithms for trace reconstruction. In *Proceedings of the 49th ACM Symposium on Theory of Computing (STOC)*, pages 1047–1056, 2017.

9 Anindya De, Ryan O'Donnell, and Rocco A. Servedio. Sharp bounds for population recovery. *CoRR*, abs/1703.01474, 2017. `arXiv:1703.01474`.

10 Z. Dvir, A. Rao, A. Wigderson, and A. Yehudayoff. Restriction access. In *Innovations in Theoretical Computer Science*, pages 19–33, 2012.

11 W. Feller. *An introduction to probability theory and its applications*. John Wiley & Sons, 1968.

12 Nina Holden and Russell Lyons. Lower bounds for trace reconstruction. Available at `arXiv:1808.02336`, 2018.

13 Nina Holden, Robin Pemantle, and Yuval Peres. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. *CoRR*, abs/1801.04783, 2018.

14 T. Holenstein, M. Mitzenmacher, R. Panigrahy, and U. Wieder. Trace reconstruction with constant deletion probability and related results. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008*, pages 389–398, 2008.

15 Svante Janson. Tail bounds for sums of geometric and exponential variables. *Statistics & Probability Letters*, 135:1–6, 2018. `doi:10.1016/j.spl.2017.11.017`.

16 Sampath Kannan and Andrew McGregor. More on Reconstructing Strings from Random Traces: Insertions and Deletions. In *IEEE International Symposium on Information Theory*, pages 297–301, 2005.

17 Akshay Krishnamurthy, Arya Mazumdar, Andrew McGregor, and Soumyabrata Pal. Trace Reconstruction: Generalized and Parameterized. *arXiv preprint*, 2019. `arXiv:1904.09618`.

18 S. Lovett and J. Zhang. Improved Noisy Population Recovery, and Reverse Bonami-Beckner Inequality for Sparse Functions. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 137–142, 2015.

19 Andrew McGregor, Eric Price, and Sofya Vorotnikova. Trace Reconstruction Revisited. In *Proceedings of the 22nd Annual European Symposium on Algorithms*, pages 689–700, 2014.

20 Ankur Moitra and Michael E. Saks. A Polynomial Time Algorithm for Lossy Population Recovery. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 110–116, 2013.

21 Fedor Nazarov and Yuval Peres. Trace reconstruction with $\exp(O(n^{1/3}))$ samples. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 1042–1046, 2017.

22 Lee Organick, Siena Dumas Ang, Yuan-Jyue Chen, Randolph Lopez, Sergey Yekhanin, Konstantin Makarychev, Miklos Z Racz, Govinda Kamath, Parikshit Gopalan, Bichlien Nguyen, et al. Random access in large-scale DNA data storage. *Nature biotechnology*, 36(3):242, 2018.

23 Yuval Peres and Alex Zhai. Average-Case Reconstruction for the Deletion Channel: Subpolynomially Many Traces Suffice. In *FOCS*, pages 228–239, 2017.

24 Yury Polyanskiy, Ananda Theertha Suresh, and Yihong Wu. Sample complexity of population recovery. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*, pages 1589–1618, 2017.

25 Krishnamurthy Viswanathan and Ram Swaminathan. Improved string reconstruction over insertion-deletion channels. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 399–408, 2008.

26 A. Wigderson and A. Yehudayoff. Population recovery and partial identification. *Machine Learning*, 102(1):29–56, 2016. Preliminary version in FOCS 2012.

27 S.M. Hossein Tabatabaei Yazdi, Ryan Gabrys, and Olgica Milenkovic. Portable and Error-Free DNA-Based Data Storage. *Scientific Reports*, 7(1):5011, 2017.

A Deferred proof of Lemma 7

We recall Lemma 7:

Lemma 7 (restated). *Let $\mathbf{r}, \mathbf{r}'$ be two n-patterns drawn independently from $\mathcal{R}_{n,q,q'}$. Then with probability at least $1 - \exp(-\Omega(n^{1/3}))$, the following three properties all hold:*
(0): $|\mathbf{r}|, |\mathbf{r}'| \geq \alpha n/2$.
(1): $|B_\ell(\mathbf{r}) \cap B_\ell(\mathbf{r}')| \geq \tau t$ *for all* $\ell \in [\tilde{s}]$.
(2): *If an* $i \in [n]$ *appears in both* $B_\ell(\mathbf{r})$ *and* $B_{\ell'}(\mathbf{r}')$ *for some* $\ell, \ell' \in [\tilde{s}]$, *then we have* $\ell = \ell'$.

We will use the following tail bounds for sums of independent geometric random variables, which are special cases of results proved by [15].

▶ **Theorem 10** (Theorems 2.1 and 3.1 in [15]). *Let $p' \in (0, 1]$, and $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_n$ be independent* Geometric(p') *random variables. Let $\boldsymbol{X} = \sum_{i \in [n]} \boldsymbol{X}_i$ and $\mu = \mathbf{E}[\boldsymbol{X}] = n/p'$. Then the following holds:*

1. *For any $\lambda \geq 1$, we have*

$$\boldsymbol{Pr}\left[X \geq \lambda\mu\right] \leq \exp\left(-p'\mu(\lambda - 1 - \ln\lambda)\right).$$

2. *For any $0 < \lambda \leq 1$, we have*

$$\boldsymbol{Pr}\left[X \leq \lambda\mu\right] \leq \exp\left(-p'\mu(\lambda - 1 - \ln\lambda)\right).$$

Note that $\lambda - 1 - \ln\lambda \geq 0$ for all $\lambda > 0$, with equality only at $\lambda = 1$. We first derive a simpler expression for the tail bounds, using the following claim:

▷ Claim 11. Let $f : (-1, \infty) \to \mathbb{R}$ be defined as $f(x) = x - \ln(1 + x)$. The following properties hold: (i) $f(0) = 0$; (ii) $f(x) > x^2/4$ for all $x \in (-1, 1] \setminus \{0\}$; and (iii) $f(x) \geq x/4$ for all $x \geq 1$.

Proof. The claim follows from elementary calculus. For item (ii) it can be shown that $g(x) = f(x) - x^2/4$ attains its minimum value 0 at $x = 0$ and is strictly convex in $(-1, 1]$. For item (iii) it is easy to verify that $h(x) = f(x) - x/4$ satisfies $h'(x) > 0$ for all $x \geq 1$, and hence its minimum value is $h(1) \geq 0.05$. ◁

Letting $x = \lambda - 1$ in Theorem 10, this claim allows us to replace the $\lambda - 1 - \ln\lambda$ term in the exponent of the tail bounds by either $(\lambda - 1)^2/4$ or $(\lambda - 1)/4$, depending on whether $\lambda < 2$ or $\lambda \geq 2$.

Now, we state and prove a few claims that will be useful for proving Lemma 7. The first claim states that property (0) in Lemma 7 holds with probability at least $1 - \exp(-\Omega(n))$.

▷ Claim 12. With probability at least $1 - \exp(-\Omega(n))$, $\boldsymbol{r} \sim \mathcal{R}_{n,q,q'}$ satisfies that $|\mathbf{r}^{(1)}| \geq \alpha n/2$.

Proof. Let $\mathbf{r}^{(1)}$ be the random string defined earlier in the generation of $\boldsymbol{r} \sim \mathcal{R}_{n,q,q'}$. As $|\mathbf{r}^{(1)}|$ is a sum of n independent Geometric(p') random variables, we have $\mu = \mathbf{E}[|\mathbf{r}^{(1)}|] = n/p'$. Invoking Theorem 10 with $\lambda = 3/4$ and Part (1) of Claim 11 with $x = \lambda - 1$, the probability of $|\mathbf{r}^{(1)}| < 3n/(4p')$ is $\exp(-\Omega(n))$.

Fixing any realization $r^{(1)}$ of $\boldsymbol{r}^{(1)}$ with $|r^{(1)}| \geq 3n/(4p')$, it follows from the standard Chernoff bound that the probability of $|\boldsymbol{r}| < \alpha n/2 \leq (2p)/3 \cdot |r^{(1)}|$ is at most $\exp(-\Omega(n))$. This finishes the proof. ◁

Fix an $i \in [n]$. Let $\boldsymbol{Y}_i^*$ be the random variable denoting the number of characters before i in $\mathbf{r}^{(1)}$ (after insertions). Recall that $\boldsymbol{Y}_i$ denotes the number of characters before i in $\mathbf{r}^{(1)}$ that survive deletions; note that $\boldsymbol{Y}_i$ is well-defined even if i is deleted. Then $\mathbf{E}[\boldsymbol{Y}_i^*] = (i/p') - 1$, and $\mathbf{E}[\boldsymbol{Y}_i] = p \cdot \mathbf{E}[\boldsymbol{Y}_i^*] = ((i/p') - 1)p$.

▶ **Lemma 13.** *For any $i \in [n]$, the probability that $\left|\boldsymbol{Y}_i - \mathbf{E}[\boldsymbol{Y}_i]\right| \geq 0.05t$ is at most $\exp(-\Omega(n^{1/3}))$.*

Proof. Let $\varepsilon = 0.05$ in the proof. We have

$$\left|\boldsymbol{Y}_i - \mathbf{E}[\boldsymbol{Y}_i]\right| \leq \left|\boldsymbol{Y}_i - p\boldsymbol{Y}_i^*\right| + \left|p\boldsymbol{Y}_i^* - p\mathbf{E}[\boldsymbol{Y}_i^*]\right| = \left|\boldsymbol{Y}_i - p\boldsymbol{Y}_i^*\right| + p \cdot \left|\boldsymbol{Y}_i^* - \mathbf{E}[\boldsymbol{Y}_i^*]\right|.$$

We first show that $|\boldsymbol{Y}_i^* - \mathbf{E}[\boldsymbol{Y}_i^*]| \leq \varepsilon t/(2p)$ with probability at least $1 - \exp(-\Omega(n^{1/3}))$. Next conditioning on any fixed realization $r^{(1)}$ of $\mathbf{r}^{(1)}$ with $|Y_i^* - \mathbf{E}[\boldsymbol{Y}_i^*]| \leq \varepsilon t/(2p)$ (in particular this implies that $Y_i^* = O(n)$) we show that $|\boldsymbol{Y}_i - pY_i^*| \leq \varepsilon t/2$ with probability at least $1 - \exp(-\Omega(n^{1/3}))$. The lemma then follows by combining these two steps. Given that the second step follows from the Hoeffding bound (with $Y_i^* = O(n)$ and $t = n^{2/3}$), we focus on the first part in the rest of the proof.

First we analyze the lower tail, i.e., the probability of $\boldsymbol{Y}_i^* - \mathbf{E}[\boldsymbol{Y}_i^*] \leq -\varepsilon t/(2p)$. Because $\boldsymbol{Y}_i^* \geq 0$ we may assume $\mathbf{E}[\boldsymbol{Y}_i^*] > \varepsilon t/(2p)$ (otherwise $\boldsymbol{Y}_i^* \geq \mathbf{E}[\boldsymbol{Y}_i^*] - \varepsilon t/(2p)$ trivially). Let

$$\lambda = 1 - \frac{\varepsilon t}{2p\mathbf{E}[\boldsymbol{Y}_i^*]} \quad \text{and} \quad x = \lambda - 1 = -\frac{\varepsilon t}{2p\mathbf{E}[\boldsymbol{Y}_i^*]},$$

so that $\lambda\mathbf{E}[\boldsymbol{Y}_i^*] = \mathbf{E}[\boldsymbol{Y}_i^*] - \varepsilon t/(2p)$. By Theorem 10 and Part (1) of Claim 11, we have

$$\mathbf{Pr}\left[\boldsymbol{Y}_i^* \leq \mathbf{E}[\boldsymbol{Y}_i^*] - \frac{\varepsilon t}{2p}\right] \leq \exp\left(-\Omega\left(\mathbf{E}[\boldsymbol{Y}_i^*] \cdot \frac{t^2}{\mathbf{E}[\boldsymbol{Y}_i^*]^2}\right)\right) \leq \exp\left(-\Omega\left(\frac{t^2}{n}\right)\right) = \exp\left(-\Omega(n^{1/3})\right).$$

For the second inequality, we used the fact that $\mathbf{E}[\boldsymbol{Y}_i^*] = O(n)$. Similarly, we analyze the upper tail. Let

$$\lambda = 1 + \frac{\varepsilon t}{2p\mathbf{E}[\boldsymbol{Y}_i^*]} \quad \text{and} \quad x = \lambda - 1 = \frac{\varepsilon t}{2p\mathbf{E}[\boldsymbol{Y}_i^*]}.$$

If $\lambda \leq 2$, Theorem 10 and Part (1) of Claim 11 imply that

$$\mathbf{Pr}\left[\boldsymbol{Y}_i^* \geq \mathbf{E}[\boldsymbol{Y}_i^*] + \frac{\varepsilon t}{2p}\right] \leq \exp\left(-\Omega\left(\mathbf{E}[\boldsymbol{Y}_i^*] \cdot \frac{t^2}{\mathbf{E}[\boldsymbol{Y}_i^*]^2}\right)\right) \leq \exp\left(-\Omega\left(\frac{t^2}{n}\right)\right) = \exp\left(-\Omega(n^{1/3})\right).$$

On the other hand, if $\lambda \geq 2$, then $x \geq 1$. By Theorem 10 and Part (2) of Claim 11, we have

$$\mathbf{Pr}\left[\boldsymbol{Y}_i^* \geq \mathbf{E}[\boldsymbol{Y}_i^*] + \frac{\varepsilon t}{2p}\right] \leq \exp\left(-\Omega\left(\mathbf{E}[\boldsymbol{Y}_i^*] \cdot \frac{t}{\mathbf{E}[\boldsymbol{Y}_i^*]}\right)\right) \leq \exp\left(-\Omega(t)\right) = \exp\left(-\Omega(n^{2/3})\right).$$

This finishes the proof of the lemma. ◀

We are ready to prove Lemma 7.

Proof of Lemma 7. We work on the three events separately and apply a union bound at the end.

(0): It follows from Claim 12 that property (0) holds with probability at least $1 - \exp(-\Omega(n))$.

(1): Fix $\ell \in [\tilde{s}]$. Recall that $I_\ell = [(2\ell-2)t+1, (2\ell-1)t] \cap \mathbb{Z}$. Let

$$I_\ell^{(0)} := \left[\frac{p'}{p}(2\ell - 1.9)t + 1, \frac{p'}{p}(2\ell - 1.1)t\right] \cap \mathbb{Z}.$$

Then $I_\ell^{(0)} \subset [n]$. We will show that with probability at least $1 - \exp(-\Omega(n^{1/3}))$, both properties below hold:

- **(a)** At least $0.7tpp'$ elements in $I_\ell^{(0)}$ survive in both $\mathbf{r}$ and $\mathbf{r}'$;
- **(b)** If an element $i \in I_\ell^{(0)}$ survives in both $\mathbf{r}$ and $\mathbf{r}'$, then $i \in B_\ell(\mathbf{r}) \cap B_\ell(\mathbf{r}')$.

Given that (a) and (b) together imply property (1), we have that property (1) holds for $\ell \in [\tilde{s}]$ with probability at least $1 - \exp(-\Omega(n^{1/3}))$. A union bound over all $\ell \in [\tilde{s}]$ implies that property (1) holds for all $\ell \in [\tilde{s}]$ with probability at least $1 - \tilde{s} \cdot \exp(-\Omega(n^{1/3})) = 1 - \exp(-\Omega(n^{1/3}))$.

So it suffices to show that (a) and (b) happen with probability at least $1 - \exp(-\Omega(n^{1/3}))$. For (a), it follows from a standard Chernoff bound that (a) holds with probability at least $1 - \exp(-\Omega(n^{2/3}))$. For (b), let i_0 and i_1 be the left and right endpoints of $I_\ell^{(0)}$, respectively. Let $\boldsymbol{Y}_{i_0}, \boldsymbol{Y}_{i_1}$ ($\boldsymbol{Y}'_{i,0}, \boldsymbol{Y}'_{i_1}$) be as defined earlier with respect to $\mathbf{r}$ ($\mathbf{r}'$). Note that $\mathbf{E}[\boldsymbol{Y}_{i_0}] = ((i_0/p') - 1)p$ and $\mathbf{E}[\boldsymbol{Y}_{i_1}] = ((i_1/p') - 1)p$. Then by Lemma 13 (and a union bound), with probability at least $1 - 4\exp(-\Omega(n^{1/3}))$, we have:

$$\boldsymbol{Y}_{i_0} \geq \mathbf{E}[\boldsymbol{Y}_{i_0}] - 0.05t > (2\ell-2)t \quad \text{and} \quad \boldsymbol{Y}_{i_1} \leq \mathbf{E}[\boldsymbol{Y}_{i_1}] + 0.05t < (2\ell-1)t,$$

and the same holds for $\boldsymbol{Y}'_{i_0}$ and $\boldsymbol{Y}'_{i_1}$. When all these events occur, then clearly all characters in $I_\ell^{(0)}$ that survive in $\mathbf{r}, \mathbf{r}'$ are in I_ℓ in both n-patterns. This finishes the analysis of property (1).

(2): Suppose a character $i \in [n]$ appears in $B_\ell(\mathbf{r})$ and $B_{\ell'}(\mathbf{r}')$ for some $\ell \neq \ell'$. Let $\boldsymbol{Y}_i, \boldsymbol{Y}'_i$ denote the number of characters before i in $\mathbf{r}, \mathbf{r}'$ respectively. Then $\mathbf{E}[\boldsymbol{Y}_i] = \mathbf{E}[\boldsymbol{Y}'_i]$. As any two distinct blocks are separated by at least t positions in the n-patterns, we have $|\boldsymbol{Y}_i - \boldsymbol{Y}'_i| \geq t$. Triangle inequality implies that $|\boldsymbol{Y}_i - \mathbf{E}[\boldsymbol{Y}_i]| \geq t/2$ or $|\boldsymbol{Y}'_i - \mathbf{E}[\boldsymbol{Y}'_i]| \geq t/2$. Assume without loss of generality that $|\boldsymbol{Y}_i - \mathbf{E}[\boldsymbol{Y}_i]| \geq t/2 > 0.05t$. Instantiating Lemma 13, we conclude that this event happens with probability at most $n \cdot \exp(-\Omega(n^{1/3}))$ which remains $\exp(-\Omega(n^{1/3}))$.

The lemma follows from a union bound. ◀

Improved Pseudorandom Generators from Pseudorandom Multi-Switching Lemmas

Rocco A. Servedio
Department of Computer Science, Columbia University, New York, NY, USA
http://www.cs.columbia.edu/~rocco
rocco@cs.columbia.edu

Li-Yang Tan
Department of Computer Science, Stanford University, Palo Alto, CA, USA
liyang@cs.stanford.edu

Abstract

We give the best known pseudorandom generators for two touchstone classes in unconditional derandomization: small-depth circuits and sparse $\mathbb{F}_2$ polynomials. Our main results are an ε-PRG for the class of size-M depth-d AC^0 circuits with seed length $\log(M)^{d+O(1)} \cdot \log(1/\varepsilon)$, and an ε-PRG for the class of S-sparse $\mathbb{F}_2$ polynomials with seed length $2^{O(\sqrt{\log S})} \cdot \log(1/\varepsilon)$. These results bring the state of the art for unconditional derandomization of these classes into sharp alignment with the state of the art for computational hardness for all parameter settings: improving on the seed lengths of either PRG would require breakthrough progress on longstanding and notorious circuit lower bounds.

The key enabling ingredient in our approach is a new *pseudorandom multi-switching lemma*. We derandomize recently-developed *multi*-switching lemmas, which are powerful generalizations of Håstad's switching lemma that deal with *families* of depth-two circuits. Our pseudorandom multi-switching lemma – a randomness-efficient algorithm for sampling restrictions that simultaneously simplify all circuits in a family – achieves the parameters obtained by the (full randomness) multi-switching lemmas of Impagliazzo, Matthews, and Paturi [39] and Håstad [35]. This optimality of our derandomization translates into the optimality (given current circuit lower bounds) of our PRGs for AC^0 and sparse $\mathbb{F}_2$ polynomials.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases pseudorandom generators, switching lemmas, circuit complexity, unconditional derandomization

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.45

Category RANDOM

Related Version A full version of the paper is available at https://arxiv.org/abs/1801.03590.

Funding *Rocco A. Servedio*: Supported by NSF grants CCF-1420349 and CCF-1563155.
Li-Yang Tan: Supported by NSF grant CCF-1563122; part of this research was done during a visit to Columbia University.

Acknowledgements We thank Prahladh Harsha and Srikanth Srinivasan for helpful discussions.

1 Introduction

Switching lemmas. Switching lemmas, first established in a series of breakthrough works in the 1980s [4, 29, 71, 34], are fundamental results stating that depth-two circuits (ORs of ANDs or vice versa) simplify dramatically when they are "hit with a random restriction." They are a powerful technique in circuit complexity, and are responsible for a remarkable suite of hardness results concerning small-depth Boolean circuits (AC^0). Switching lemmas


© Rocco A. Servedio and Li-Yang Tan;
licensed under Creative Commons License CC-BY

are at the heart of several near-optimal bounds on AC^0 circuits, such as essentially optimal correlation bounds against the PARITY function [39, 35] and the worst-case and average-case depth hierarchy theorems of [34, 59, 36]. Indeed, comparably strong results are lacking (and are major open problems) for seemingly small extensions of AC^0, such as AC^0 augmented with parity or mod-p gates, for which switching lemmas do not apply; this gap highlights the importance of switching lemmas as a proof technique.

Switching lemmas are versatile as well as powerful: many results in circuit complexity rely on sophisticated variants and generalizations of the "standard" switching lemmas. Recent examples include the aforementioned correlation bounds and average-case depth hierarchy theorems, as well as powerful lower bounds on the circuit complexity of the CLIQUE problem [12, 57], lower bounds on the small-depth circuit complexity of ST-CONNECTIVITY [25], and lower bounds against AC^0 formulas [58]. Beyond the immediate arena of circuit lower bounds, switching lemmas are also important tools in diverse areas including propositional proof complexity [54, 43, 55, 37], computational learning theory [44], the design of circuit satisfiability algorithms [13, 39], and coding theory [23, 10].

This paper is about the role of switching lemmas in the study of *unconditional pseudorandomness*. Switching lemmas have a long history in this area; indeed, arguably the first work in unconditional derandomization, the seminal paper of Ajtai and Wigderson [6], was based on a *pseudorandom* switching lemma, which they used to give the first non-trivial pseudorandom generator for AC^0. (Interestingly, after many subsequent developments described in detail in Section 2, we come full circle in this paper and use the [6] framework to give a new pseudorandom generator for AC^0 that is essentially best possible without improving longstanding circuit lower bounds.) One key contribution that we make in this paper is to bring together two important generalizations of standard switching lemmas, one quite old and one very new:

- **(i)** *pseudorandom* switching lemmas (originating in [6]), which employ pseudorandom rather than "fully random" restrictions, and
- **(ii)** recently developed *multi-switching lemmas* [39, 35] which simultaneously simplify all of the depth-two circuits in a family of such circuits, rather than a single depth-two circuit as is the case for standard switching lemmas.

Let us discuss each of these generalizations in turn.

Pseudorandom switching lemmas. The (truly) random restrictions that are used in standard switching lemmas make a coordinatewise-independent random choice for each input variable $x_1, \ldots, x_n$ of whether to map it to 0, to 1, or to leave it unassigned (map it to $*$); standard switching lemmas show that a depth-two circuit simplifies dramatically with very high probability when it is hit with such a random restriction. Such "truly random" restrictions are inherently incompatible with unconditional derandomization, which naturally motivates the notion of a *pseudorandom* switching lemma. Such a result defines a much smaller probability space of "pseudorandom" restrictions, and proves that a restriction drawn randomly from this space also has the effect of simplifying a depth-two circuit with high probability. While pseudorandom switching lemmas have been the subject of much research since they were first introduced by Ajtai and Wigderson [6, 5, 24, 3, 32, 39, 31, 65, 30], and have been applied in a range of different ways in unconditional derandomization, they are not yet fully understood.

The designer of a pseudorandom switching lemma faces an inherent tension between achieving strong parameters – intuitively, having a depth-two circuit simplify as much as possible while keeping a large fraction of variables alive – and using as little randomness as

possible. Prior to the work of Trevisan and Xue [65], known pseudorandom switching lemmas fell short of achieving the parameters of Håstad's influential "full randomness" switching lemma [34]. In particular, a parameter of central importance in essentially all applications of switching lemmas is the probability that a given coordinate x_i remains alive under a random (or pseudorandom) restriction; this is often referred to as the "$*$-probability" and denoted by p. A crucial quantitative advantage of Håstad's switching lemma over previous works is that it can be applied even when p is as large as $\Omega(1/\log n)$ for $\mathrm{poly}(n)$-size depth-two circuits – in contrast, the earlier works of [4, 29, 71] required $p = n^{-\Omega(1)}$ – and yields a very strong conclusion, namely that with high probability the restricted circuit collapses to a shallow decision tree[1]. (For example, while the recent pseudorandom switching lemma of [31] is able to achieve a relatively large p, the conclusion of that switching lemma is that the restricted depth-two circuit can w.h.p. be sandwiched by depth-two circuits with small bottom fan-in, which is weaker than the aforementioned decision tree conclusion.)

Trevisan and Xue [65] give a *pseudorandom* switching lemma that is highly randomness efficient and yet achieves the parameters of Håstad's fully random switching lemma (i.e. [65] achieves the same simplification, collapsing to a shallow decision tree, that follows from [34], with the same $*$-parameter p as [34]). The key conceptual ingredient enabling this is a beautiful idea of "fooling the proof" of the Håstad's switching lemma, exploiting its "computational simplicity". Trevisan and Xue leverage their pseudorandom switching lemma to construct a new pseudorandom generator for AC^0, obtaining the first improvement of Nisan's celebrated PRG [52] in over two decades. We elaborate on Trevisan and Xue's ideas and how they obtain their PRG later in Section 2.1.

Multi-switching lemmas. The switching lemma shows that any width-k CNF formula collapses to a shallow decision tree with high probability under a random restriction. Via a simple union bound it is of course possible to extend this result to say that a family of width-k CNF formulas will all collapse to a shallow decision tree with high probability under a random restriction; but this naive approach leads to a quantitative loss in parameters if the argument is iterated, as it typically is, $d-1$ times to analyze a depth-d circuit. (The exact nature of this quantitative loss is important but somewhat subtle; see Section 3 for a detailed explanation.)

Via an ingenious extension of the ideas underlying the original switching lemma, Håstad [35] developed "multi-switching lemmas" that essentially bypass this quantitative loss in parameters that results from iterating a naive union bound (see also the work of Impagliazzo, Matthews, and Paturi [39] for closely related results). Roughly speaking, [35] shows that a *family* of width-k CNF formulas will with high probability have a shallow *common partial decision tree*. Without explaining this structure in detail here (again see Section 3 for a detailed explanation), this makes it possible to iterate the argument and tackle depth-d circuits without incurring a quantitative loss in parameters. The savings thus achieved is the key new ingredient that allowed [39, 35] to achieve essentially optimal correlation bounds for AC^0 against the Parity function, capping off a long line of work [4, 71, 34, 19, 8, 13]. These ideas have also been leveraged to achieve new algorithmic results such as better-than-brute-force satisfiability algorithms and distribution-free PAC learning algorithms for AC^0 [13, 39, 60].

[1] The first published version of the switching lemma with a decision tree conclusion is due to Cai [19]; several authors subsequently noted that Håstad's argument also yields such a conclusion.

A pseudorandom multi-switching lemma. A core technical contribution of this paper is to bring together these two lines of work, on pseudorandom switching lemmas and on multi-switching lemmas. Since the precise statement of our pseudorandom multi-switching lemma, Theorem 14, is somewhat involved we defer it to Section 4 and here merely make some remarks about it. In the spirit of Trevisan and Xue's derandomization of the original switching lemma, to obtain Theorem 14 we "fool the proof" of Håstad's multi-switching lemma [35], exploiting its "computational simplicity". This enables us to achieve optimal parameters in the same sense as [65], namely, that it establishes the same dramatic simplification – now of the family $\mathcal{F}$ of depth-two circuits – as [35], and while only requiring the same $*$-probability p as [35]. Our pseudorandom switching lemma is highly efficient in its use of randomness; this randomness efficiency is crucial in the constructions of our pseudorandom generators for AC^0 circuits and sparse $\mathbb{F}_2$ polynomials using Theorem 14, which we now describe in the next section.[2]

2 PRGs for AC^0 and sparse $\mathbb{F}_2$ polynomials

We employ our pseudorandom multi-switching lemma to give the best known pseudorandom generators for two canonical classes in unconditional derandomization: AC^0 circuits and sparse $\mathbb{F}_2$ polynomials. As we describe in this section, our results bring the state of the art for unconditional derandomization of these classes into sharp alignment with the state of the art for computational hardness: improving on the seed lengths of either PRG would require breakthrough progress on longstanding and notorious circuit lower bounds. In this sense, our results are in the same spirit as those of Imagliazzo, Meka, and Zuckerman [40], which gave optimal (assuming current circuit lower bounds) pseudorandom generators for various classes of Boolean formulas and branching programs; however, our techniques are very different from those of [40].

2.1 PRGs for AC^0 circuits

The class of small-depth Boolean circuits (AC^0) is a class of central interest in unconditional derandomization, and has been the subject of intensive research in this area over the past 30 years [6, 45, 52, 53, 50, 49, 41, 64, 66, 11, 56, 18, 42, 26, 2, 1, 62, 47, 28, 32, 31, 65, 30, 63, 33, 21]. This highly successful line of work on derandomizing AC^0 has generated a wealth of ideas and techniques that have become mainstays in the field of pseudorandomness. A prominent example is Nisan's celebrated PRG for AC^0 circuits [52], which introduced ideas that enriched the surprising connections between pseudorandomness and computational hardness [14, 70, 53]. The *hardness-versus-randomness paradigm* asserts, qualitatively, that strong explicit PRGs exist if and only if strong explicit circuit lower bounds exist. In the context of unconditional derandomization (the subject of this work), this strongly motivates the goal of constructing, for every circuit class $\mathcal{C}$, unconditional PRGs for $\mathcal{C}$ that are best possible given the current best lower bounds for $\mathcal{C}$. In other words, this is the goal of achieving a *quantitatively optimal hardness to randomness conversion* for $\mathcal{C}$, converting "all the hardnesss" in our lower bounds for $\mathcal{C}$ into pseudorandomness for $\mathcal{C}$.

[2] While our focus in this work is on unconditional derandomization, we briefly mention that recent work of Ball et al. [10] establishes a new connection between pseudorandom switching lemmas and *non-malleable codes* in coding theory [27]. Using this connection, [10] are able to leverage the randomness efficiency of [65]'s pseudorandom switching lemma in their design of new non-malleable codes for small-depth circuits. We leave the possibility of applying our techniques to obtain further-improved non-malleable codes as an interesting avenue for future work.

For $\mathcal{C}$ being the class of n-variable size-M depth-d AC^0 circuits this amounts to constructing PRGs with seed length $\log^{d-1}(Mn)\log(1/\varepsilon)$: such seed length is best possible without improving longstanding AC^0 lower bounds that date back to the 1980s [34]. (More precisely, it is well known, see e.g. [65], that achieving seed length say $\log^{d-1.01}(Mn)\log(1/\varepsilon)$ would yield $\exp(\omega(n^{1/(d-1)})$ size lower bounds against depth-d AC^0 circuits, which is a barrier that has stood for over 30 years even in the $d = 3$ case.) We give the first construction of a PRG that achieves this seed length up to an *additive* absolute constant in the exponent of $\log(Mn)$:

▶ **Theorem 1** (PRG for AC^0 circuits)**.** *For every $d \geq 2$, $M \in \mathbb{N}$ and $\varepsilon > 0$, there is an ε-PRG for the class of n-variable size-M depth-d circuits with seed length $\log^{d+O(1)}(Mn)\log(1/\varepsilon)$.*

2.1.1 Background and prior PRGs for AC^0 circuits

As noted above there has been a significant body of work on PRGs for AC^0 circuits, spanning over 30 years. In this section we give a brief overview of the history and prior state-of-the-art for this touchstone problem in unconditional derandomization.

Ajtai–Wigderson and Nisan. Ajtai and Wigderson, in their seminal work [6] pioneering the study of unconditional derandomization, constructed the first non-trivial PRG for AC^0 circuits with an $n^{o(1)}$ seed length; we will discuss their techniques in detail later. [6]'s seed length was improved significantly in the celebrated work of Nisan [52], using what is now known as the Nisan–Wigderson framework [53], which provides a generic template for converting correlation bounds against a circuit class to PRGs for a closely related class (in the case of AC^0 these two classes essentially coincide). Via this approach Nisan showed how correlation bounds for AC^0 against the Parity function [34] yield a PRG with seed length $\log^{2d+O(1)}(Mn/\varepsilon)$.

We remark that the generality of the Nisan–Wigderson framework comes at a quantitative price: it is straightforward to verify that a seed length of $(\log^d(Mn) + \log(1/\varepsilon))^2$ is the best that can be achieved via this framework given current AC^0 circuit lower bounds (see e.g. [65, 33]). This is roughly quadratically worse than the sought-for $\log^{d-1}(Mn)\log(1/\varepsilon)$, the best that can be achieved assuming *only* current AC^0 circuit lower bounds.

Bounded independence fools AC^0. Nisan's seed length for AC^0 circuits stood unmatched for more than two decades. However, in this interim period there was significant progress on showing that distributions with bounded independence fool AC^0, a well-known conjecture posed by Linial and Nisan [45]. Braverman's breakthrough result [18] showed that polylog(n)-wise independence fools AC^0, which (along with standard constructions of k-wise independent distributions) gave a PRG with seed length $\log^{O(d^2)}(Mn/\varepsilon)$; this was subsequently sharpened to $\log^{3d+O(1)}(Mn/\varepsilon)$ by Tal [63]. Recently, Harsha and Srinivasan [33] further improved the seed length of Braverman's generator to $\log^{3d+O(1)}(Mn)\log(1/\varepsilon)$, which is notable for its optimal dependence on the error parameter ε.

The work of Trevisan and Xue. Recent work of Trevisan and Xue [65] makes a significant advance towards achieving seed length $\log^{d-1}(Mn)\log(1/\varepsilon)$: their work circumvents the "quadratic loss" associated with the Nisan–Wigderson framework with a PRG of seed length $\log^{d+O(1)}(Mn/\varepsilon)$. This is the first PRG to achieve a $\log^{d+O(1)}(Mn)$ dependence, an exponent that is within an *additive* absolute constant of the sought-for $\log^{d-1}(Mn)$, and is also the first strict improvement on Nisan's seed length in more than two decades. (Note however, that like Nisan's PRG the dependence on ε is suboptimal: $\log^{d+O(1)}(1/\varepsilon)$ instead of $\log(1/\varepsilon)$.)

Rather than going through the Nisan–Wigderson framework – which, as noted above, carries with it an associated quantitative loss in parameters – Trevisan and Xue construct their PRG by *derandomizing the proof* of AC^0 lower bounds, "opening up the black-box" of AC^0 lower bounds, so to speak. At a high level, [65] adopts the strategy employed in the early work of Ajtai and Wigderson [6]. We describe this strategy in detail in the full version of this paper, but roughly speaking, Ajtai and Wigderson introduced a powerful and generic framework for constructing PRGs from pseudorandom switching lemmas. In [6], they instantiated this framework with a derandomization of Ajtai's switching lemma [4] – which underlies his proof of the first superpolynomial lower bounds against AC^0 – to obtain the first non-trivial PRG for AC^0. Trevisan and Xue obtain their PRG by revisiting this early framework of [6], instantiating it with their derandomization of Håstad's switching lemma [34]. (And as we will soon discuss, in this work we obtain our PRG by instantiating the [6] framework with our derandomization of the [35] multi-switching lemmas.)

PRGs via polarizing random walks. Finally, in recent exciting work Chattopadhyay, Hatami, Hosseini, and Lovett [21] have introduced an elegant new framework for obtaining pseudorandom generators which has consequences for fooling AC^0. Their framework is based on a notion of "fractional" pseudorandom generators, which are used as steps in a random walk which ultimately yields a (standard) pseudorandom generator. [21] show that if a class $\mathcal{C}$ is closed under restrictions and has sufficiently strong Fourier concentration on low-degree coefficients, then almost k-wise independence suffice to yield a fractional PRG, which their random walk approach can then convert into a standard PRG against $\mathcal{C}$. Using Tal's sharp bounds [63] on the Fourier concentration of AC^0, they obtain a seed length of $O(\log(n/\varepsilon)(\log(\log(n)/\varepsilon))\log^{2d-2} M)$ for size-M depth-d circuits.

2.1.2 Our PRG and approach

To summarize, prior to our work there were three incomparable best known PRGs for AC^0, achieving three different tradeoffs in the overall dependence on M, d and $1/\varepsilon$. These were the PRG of Trevisan and Xue [65], which has seed length $\log^{d+O(1)}(Mn/\varepsilon)$; Harsha and Srinivasan's improvement of Braverman's generator [33], which has seed length $\log^{3d+O(1)}(Mn)\log(1/\varepsilon)$; and the [21] PRG, which has seed length $O(\log(n/\varepsilon)(\log(\log(n)/\varepsilon))\cdot \log^{2d-2} M)$, i.e. essentially $\log^{2d-1}(Mn)\log^2(1/\varepsilon)$.

Theorem 1 unifies and improves these three incomparable seed lengths. Our PRG achieves an essentially optimal hardness to randomness conversion for AC^0: our seed length of $\log^{d+O(1)}(Mn)\log(1/\varepsilon)$ comes very close to $\log^{d-1}(Mn)\log(1/\varepsilon)$, which is best possible without improving longstanding AC^0 circuit lower bounds that date back to the 1980s.

Table 1 provides a comparison of the seed length of our PRG (and the techniques that underlie our construction) and those of previous work.

Our approach. Our approach draws on and unifies ideas in the works of [6, 65, 33] discussed above, which we use in conjunction with our derandomization of the [35] multi-switching lemma to obtain our PRG.

At a high level, we adopt the overall conceptual strategy of Ajtai and Wigderson [6] and Trevisan and Xue [65], and obtain our PRG by derandomizing the proof of AC^0 lower bounds. The key technical ingredient in our PRG construction is our pseudorandom multi-switching lemma, a derandomization of the multi-switching lemmas which underlie the [39, 35] optimal correlation bounds for AC^0 against Parity. Our pseudorandom multi-switching lemma improves both the pseudorandom switching lemma of [65] (a derandomization of Håstad's

Table 1 PRGs for ε-fooling n-variable size-M depth-d AC^0 circuits.

Reference	Seed length	Techniques
[6]	$n^{o(1)}$ for $M = \mathrm{poly}(n)$	derandomize [4] switching lemma
[52]	$\log^{2d+O(1)}(Mn/\varepsilon)$	[53] framework, [34] correlation bounds
[18]	$\log^{O(d^2)}(Mn/\varepsilon)$	bounded independence
[65]	$\log^{d+O(1)}(Mn/\varepsilon)$	[6] framework, derandomize [34] switching lemma
[63]	$\log^{3d+O(1)}(Mn/\varepsilon)$	bounded independence
[33]	$\log^{3d+O(1)}(Mn)\log(1/\varepsilon)$	bounded independence
[21]	(essentially) $\log^{2d-1}(Mn)\log^2(1/\varepsilon)$	almost bounded independence, fractional PRGs, polarizing random walks
This work	$\log^{d+O(1)}(Mn)\log(1/\varepsilon)$	[6] framework, derandomize [35] multi-switching lemma, bounded independence

switching lemma [34] which underlies his exponential lower bounds against AC^0) and the pseudorandom switching lemma of [6] (a derandomization of Ajtai's switching lemma [4] which underlies his superpolynomial lower bounds against AC^0).

Our derandomization of the [35] multi-switching lemma is largely influenced by Trevisan and Xue's derandomization of the Håstad's original switching lemma [34]. We describe our approach in detail in Section 4, but highlight here the simple but ingenious new idea underlying [65]'s argument. Very roughly speaking, they derandomize the [34] switching lemma by "fooling its proof": showing that Håstad's proof of his switching lemma "cannot δ-distinguish" between truly random restrictions and pseudorandom restrictions drawn from polylog(n)-wise independent distributions. Since Håstad's switching lemma holds for truly random restrictions, it thus follows that it also holds for pseudorandom restrictions drawn from polylog(n)-wise independent distributions (up to a δ additive loss in the failure probability).

To accomplish this, Trevisan and Xue exploit the fact that Håstad's proof of the switching lemma is "computationally simple": for a fixed k-CNF F, there is a small depth-3 circuit that takes as input an encoding of a restriction ρ, and outputs 1 iff ρ is a bad restriction for the desired conclusion of Håstad's switching lemma, contributing to its failure probability (more precisely, the failure event is that the "canonical decision tree" for $F \restriction \rho$ has large depth). In similar spirit, our derandomization of the [35] multi-switching lemma also exploits the "computational simplicity" of its proof. In our case, for a fixed family $\mathcal{F}$ of k-CNF formulas we construct a small depth-4 circuit for recognizing bad restrictions (the one additional layer of depth reflects the fact that multi-switching lemmas are, roughly speaking, "one quantifier more complex" than switching lemmas). To obtain optimal parameters in our PRG constructions, we use the $d = 3$ case of Harsha and Srinivasan's strengthening of Braverman's generator [33] to fool this depth-4 circuit, and hence show that [35]'s proofs of the multi-switching lemmas "cannot distinguish" between truly random and pseudorandom restrictions. The fact that [33] achieves an optimal $\log(1/\varepsilon)$ seed length dependence plays a crucial role in enabling the optimal $\log(1/\varepsilon)$ seed length dependence of our PRG.

2.2 PRGs for sparse $\mathbb{F}_2$ polynomials

Our second main result deals with the class of sparse $\mathbb{F}_2$ polynomials. Like AC^0 circuits, sparse $\mathbb{F}_2$ polynomials and low-degree $\mathbb{F}_2$ polynomials have been extensively studied in unconditional derandomization [51, 7, 50, 15, 66, 46, 68, 16, 47, 48, 22].

Via the hardness-versus-randomness paradigm, the problem of derandomizing $\mathbb{F}_2$ polynomials is intimately related to that of proving correlation bounds for $\mathbb{F}_2$ polynomials. A prominent open problem in the latter context – arguably the current flagship challenge in this area – is that of obtaining superpolynomially small correlation bounds against $\mathbb{F}_2$ polynomials of degree $\log n$. Degree $\log n$ represents the fundamental limit of our current suite of powerful techniques for proving $\mathbb{F}_2$ correlation bounds [9, 17, 20, 69], and breaking this "degree $\log n$ barrier" would constitute a significant technical breakthrough[3]. See Open Question 1 of Viola's excellent survey [67] for a detailed discussion of this important open problem and its relationship with other central challenges in complexity theory.

As a second application of our pseudorandom multi-switching lemma, we give an ε-PRG for S-sparse $\mathbb{F}_2$ polynomials with seed length $2^{O(\sqrt{\log S})}\log(1/\varepsilon)$, which is best possible without breaking the aforementioned "degree $\log n$ barrier" for $\mathbb{F}_2$ correlation bounds:

▶ **Theorem 2** (PRG for sparse $\mathbb{F}_2$ polynomials)**.** *For every $S = 2^{\omega(\log\log n)^2}$ and $\varepsilon > 0$ there is a PRG with seed length $2^{O(\sqrt{\log S})}\log(1/\varepsilon)$ that ε-fools the class of n-variable S-sparse $\mathbb{F}_2$ polynomials.*

Background and prior PRGs for $\mathbb{F}_2$ polynomials. The first unconditional PRGs for $\mathbb{F}_2$ polynomials were given in early influential work of Luby, Veličković, and Wigderson [50], who constructed a PRG that ε-fools size-S $\mathsf{SYM} \circ \mathsf{AND}$ circuits – including S-sparse $\mathbb{F}_2$ polynomials as an important special case – with seed length $2^{O(\sqrt{\log(S/\varepsilon)})}$. To obtain their PRG, Luby et al. employed the Nisan–Wigderson framework [53] together with multi-party number-on-the-forehead (NOF) communication complexity lower bounds from the seminal work of Babai, Nisan, and Szegedy [9]. Viola [66] subsequently extended this $2^{O(\sqrt{\log(S/\varepsilon)})}$ seed length to the broader class of $\mathsf{SYM} \circ \mathsf{AC}^0$ circuits with a more modular proof. In recent work [61], the authors have improved the seed length dependence on ε of [50, 66] to $2^{O(\sqrt{\log(S)})} + \mathrm{polylog}(1/\varepsilon)$. We discuss the relation between our techniques and those of [61] in more detail below.

In a related line of work, PRGs for *low-degree* $\mathbb{F}_2$ polynomials have also been intensively studied. Starting with the fundamental results of Naor and Naor [51] on ε-biased distributions (which resolved the degree-1 case), this research continued through an exciting line of work on the degree $k \geq 2$ case [15, 16] and culminated in the breakthroughs of Lovett [46] and Viola [68] which are described in more detail below. It is interesting to note that prior to our work, the underlying techniques used for the sparse case (multi-party communication complexity) are completely different from the techniques used for the low-degree case (Fourier analysis).

Our PRG and approach. Theorem 2 gives an exponential and optimal improvement of the PRG of [50] in terms of its dependence on the error parameter ε. Our PRG achieves an optimal hardness to randomness conversion for $\mathbb{F}_2$ polynomials: since every $\log(n)$-degree $\mathbb{F}_2$ polynomial has at most $n^{\log n}$ monomials, it can be shown (using the simple Proposition 3.1 of [68]) that a PRG with seed length $2^{o(\sqrt{\log S})}\log(1/\varepsilon)$ would break the degree $\log n$ barrier.

[3] Breaking this "degree $\log n$ barrier" is also well-known (via a simple and beautiful observation of Håstad and Goldmann [38]) to be a prerequisite for breaking the notorious "$\log n$ party barrier" in multi-party communication complexity [9], a longstanding open problem that has resisted attack for over two decades.

Our techniques for Theorem 2 are substantially different from the techniques of [61, 66]. As summarized in Table 2, the basic approach of [61], like [66] and [50], is via the Nisan-Wigderson paradigm using multi-party communication complexity bounds; the main point of departure between [61] and [66] is that [61] leverages Håstad's multi-switching lemma from [35] in place of his earlier [34] switching lemma which was used in [66]. (We note that similar to the situation for AC^0 circuits, it is straightforward to verify that our optimal $\log(1/\varepsilon)$ dependence is not achievable via the Nisan–Wigderson framework without dramatic breakthroughs in correlation bounds for $\mathbb{F}_2$ polynomials, going well beyond breaking the degree $\log n$ barrier.) In contrast, we do not use the Nisan–Wigderson framework or multi-party communication complexity lower bounds; instead, as for AC^0, our approach is based on the [6] framework and our *derandomization* of the [35] multi-switching lemma. Indeed, our approach to obtaining Theorem 2 bridges the two previously disparate lines of work on pseudorandomness for sparse and low degree polynomials: roughly speaking, it can be viewed as a reduction from PRGs for S-sparse polynomials to PRGs for degree-$\sqrt{\log S}$ polynomials. This allows us to leverage the result of Viola [68] (building on the work of Lovett [46]), which gives PRGs for n-variable degree-k $\mathbb{F}_2$ polynomials with seed length

$$O(k \log n + k2^k \log(1/\varepsilon)).$$

More precisely, at the heart of our reduction is a new pseudorandom switching lemma for sparse $\mathbb{F}_2$ polynomials, showing that such a polynomial is very likely to collapse to a *small-depth decision tree with low-degree $\mathbb{F}_2$ polynomials at its leaves* under a suitable pseudorandom restriction. This is essentially a special case of our pseudorandom multi-switching lemma. With this reduction in hand, we then exploit the strength and generality of Viola's result – roughly speaking, that the sum of k independent copies of a sufficiently strong ε-biased distribution fools degree-k polynomials – to show that his PRG extends to fool not only low-degree polynomials, but also small-depth decision trees with low-degree polynomials at their leaves.

Table 2 provides a comparison of the seed length of our PRG (and the techniques that underlie our construction) and those of previous work.

Table 2 PRGs for ε-fooling $\mathbb{F}_2$ polynomials.

Reference/ Class	Seed length	Techniques
[50] S sparse	$2^{O(\sqrt{\log(S/\varepsilon)})}$	[53] framework, [9] multi-party NOF communication complexity
[61] S sparse	$2^{O(\sqrt{\log S})} + (\log(1/\varepsilon))^{4.01}$	[53] framework, [9] multi-party NOF communication complexity, [35] multi-switching lemma
[46] degree k	$O(2^k \log n + 4^k \log(1/\varepsilon))$	Fourier analysis
[68] degree k	$O(k \log n + k2^k \log(1/\varepsilon))$	Fourier analysis
This work S sparse	$2^{O(\sqrt{\log S})} \log(1/\varepsilon)$	[6] framework, derandomize [35] multi-switching lemma, Fourier analysis, bounded independence

2.3 Organization

Section 2.4 recalls some basic preliminaries from unconditional pseudorandomness. We describe and contrast the original Håstad switching lemma [34] versus the [35] multi-switching lemma in Section 3. Section 3.1 establishes some infrastructure towards derandomizing the [35] switching lemma, and the actual derandomization result (the pseudorandom multi-switching lemma, Theorem 14) is stated in Section 4 and proved in Appendix A. In the full version we describe a general framework for constructing pseudorandom generators that is implicit in the work of Ajtai and Wigderson [6], and explain how our derandomized multi-switching lemma from Section 4 can be used (along with other ingredients) within this framework to establish the PRGs for AC^0 and for sparse $\mathbb{F}_2$ polynomials that are our main PRG results.

2.4 Preliminaries

For $r < n$, we say that a distribution $\mathcal{D}$ over $\{0,1\}^n$ can be *sampled efficiently with r random bits* if (i) $\mathcal{D}$ is the uniform distribution over a multiset $z^{(1)}, \ldots, z^{(s)}$ of strings from $\{0,1\}^n$ where $s \in [\frac{1}{\mathrm{poly}(n)} \cdot 2^r, 2^r]$ and (ii) there is a deterministic algorithm $\mathrm{Gen}_{\mathcal{D}}$ which, given as input a uniform random element of $[s]$, runs in time $\mathrm{poly}(n, s)$ and outputs a string drawn from $\mathcal{D}$.

For $\delta > 0$ and a class $\mathcal{C}$ of functions from $\{0,1\}^n$ to $\{0,1\}$, we say that a distribution $\mathcal{D}$ over $\{0,1\}^n$ *δ-fools $\mathcal{C}$ with seed length r* if (a) $\mathcal{D}$ can be sampled efficiently with r random bits via algorithm $\mathrm{Gen}_{\mathcal{D}}$, and (b) for every function $f \in \mathcal{C}$, we have

$$\left| \mathop{\mathbf{E}}_{\boldsymbol{s} \leftarrow \{0,1\}^r} \left[f(\mathrm{Gen}_{\mathcal{D}}(\boldsymbol{s})) \right] - \mathop{\mathbf{E}}_{\boldsymbol{x} \leftarrow \{0,1\}^n} \left[f(\boldsymbol{x}) \right] \right| \leq \delta.$$

Equivalently, we say that $\mathrm{Gen}_{\mathcal{D}}$ is a *δ-PRG for $\mathcal{C}$ with seed length r*.

Two kinds of distributions which are extremely useful in derandomization are *δ-biased* and *k-wise independent* distributions. We say that a distribution $\mathcal{D}$ over $\{0,1\}^n$ is *δ-biased* if it δ-fools the class of all 2^n parity functions $\{\textsc{Parity}_S\}_{S \subseteq [n]}$, where $\textsc{Parity}_S : \{0,1\}^n \to \{0,1\}$ is defined by $\textsc{Parity}_S(x) = \sum_{i \in S} x_i \mod 2$. We say that a distribution $\mathcal{D}$ over $\{0,1\}^n$ is *k-wise independent with parameter p* if for every $1 \leq i_1 < \cdots < i_k \leq n$ and every $(b_1, \ldots, b_k) \in \{0,1\}^k$, we have

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x} \leftarrow \mathcal{D}} \left[\boldsymbol{x}_{i_1} = b_1 \text{ and } \cdots \text{ and } \boldsymbol{x}_{i_k} = b_k \right] = p^{\sum_{j=1}^k b_j} \cdot (1-p)^{k - \sum_{j=1}^k b_j},$$

i.e. every subset of k coordinates is distributed identically to a product distribution with parameter p.

A *restriction* ρ of variables $x_1, \ldots, x_n$ is an element of $\{0, 1, *\}^n$. We write $\mathrm{supp}(\rho)$ to denote the set of coordinates that are fixed to 0 or 1 by ρ. Given a function $f(x_1, \ldots, x_n)$ and a restriction ρ, we write $f \restriction \rho$ to denote the function obtained by fixing x_i to $\rho(i)$ if $\rho(i) \in \{0,1\}$ and leaving x_i unset if $\rho(i) = *$. For two restrictions $\rho, \rho' \in \{0,1,*\}^n$, their *composition*, denoted $\rho\rho' \in \{0,1,*\}^n$, is the restriction defined by

$$(\rho\rho')_i = \begin{cases} \rho_i & \text{if } \rho_i \in \{0,1\} \\ \rho'_i & \text{otherwise.} \end{cases}$$

Given a collection $\mathcal{F} = \{f_1, \ldots, f_M\}$ of functions and a restriction ρ we write $\mathcal{F} \restriction \rho$ to denote the family $\{f_1 \restriction \rho, \ldots, f_M \restriction \rho\}$.

Given an AC^0 circuit, we define its size to include the input variables (along with the number of gates in the circuit). We adopt this convention for notational convenience, since we may then always assume that the size M of an n-variable circuit is always at least n. (We do *not* adopt this convention for $\mathbb{F}_2$ polynomials: as is standard, we define the sparsity of an $\mathbb{F}_2$ polynomial to be the number of monomials in its support.)

Finally, if g is a Boolean function and $\mathcal{C}$ is a class of circuits, we say that g is *computed by a* $(t, \mathcal{C})$*-decision tree* if g is computed by a decision tree of depth t (with single Boolean variables x_i at internal nodes as usual) in which each leaf is labeled by a function from $\mathcal{C}$.

3 Multi-switching lemmas

At the heart of almost all applications of Håstad's original switching lemma [34] is a powerful structural fact about AC^0 circuits: every AC^0 circuit "collapses" (i.e. simplifies dramatically) to a depth-t decision tree with high probability, at least $1 - \varepsilon$, under a random restriction that randomly fixes a $(1-p)$-fraction of coordinates. In the precise quantitative statement of this fact, both t and p depend on ε: as the desired failure probability ε tends to 0, the $*$-probability p tends to 0 (more coordinates are fixed) and t tends to n (the resulting decision tree is of larger depth). It is easy to see that this dependence is inherent given the statement of the [34] switching lemma, and indeed this will be clear from the discussion later in this section.

The recent multi-switching lemma of Håstad [35] (see also [39]) achieves a remarkable strengthening of the above: essentially the same structural fact about AC^0 holds (in terms of the quantitative relation between the decision tree depth t and the failure probability ε) *with the* $*$*-probability* p *being independent of* ε. This is the key qualitative difference underlying the optimal AC^0 correlation bounds for Parity obtained in [39, 35]; likewise, in this work, this is the key qualitative difference underlying the optimal ε-dependence in the seed lengths of our PRGs for AC^0 circuits and sparse $\mathbb{F}_2$ polynomials.

Let $\mathcal{R}_p$ denote the random restriction which independently sets each variable x_i to 0 with probability $(1-p)/2$, to 1 with probability $(1-p)/2$, and to $*$ with probability p. We first recall the original switching lemma from [34]:

▶ **Theorem 3** (Håstad's switching lemma). *Let F be a k-CNF. Then for all $t \geq 1$, we have that*

$$\Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_p} [F \restriction \boldsymbol{\rho} \text{ does not have a decision tree of depth } t] \leq (5pk)^t.$$

In the context of AC^0 circuits the switching lemma is used to achieve *depth reduction* under random restrictions: we apply Theorem 3 separately to each of the bottom-layer depth-2 subcircuits, choosing t appropriately so that all of them "switch" to depth-t decision trees with high probability. The following corollary is what is typically used:

▶ **Corollary 4** (AC^0 depth reduction via Theorem 3). *Let $\mathcal{C}$ be a size-M depth-d AC^0 circuit with bottom fan-in k, and let $p = 1/(10k)$. Then for all $\varepsilon > 0$,*

$$\Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_p} \left[\mathcal{C} \restriction \boldsymbol{\rho} \text{ is not computed by a depth-}(d-1) \text{ circuit with bottom fan-in } \log(M/\varepsilon)\right] \leq \varepsilon.$$

Proof. This follows from applying Theorem 3 with $t = \log(M/\varepsilon)$ to each of the bottom-layer depth-2 subcircuits of $\mathcal{C}$ (at most M of them), along with the basic fact that a depth-t decision tree can be expressed as both a t-DNF as well as a t-CNF. ◀

The same argument is then repeated again on the ($k = \log(M/\varepsilon)$)-DNFs at the bottom two layers of the new circuit (applying the dual form of the switching lemma for k-DNFs rather than k-CNFs) to further reduce the depth to $d-2$. However, observe that in this second application of the switching lemma (and in later applications as well), in order to use Corollary 4, the parameter p of the random restriction must now depend on ε, since we must now take $p < 1/(5k) = 1/(5\log(M/\varepsilon))$ in order to get a nontrivial bound in Theorem 3. This is why standard applications of the [34] switching lemma (involving $d-1$ iterative applications of Corollary 4) show that every size-M depth-d AC^0 circuit collapses to depth-($t = \log(M/\varepsilon)$) decision tree with high probability, at least $1-\varepsilon$, under a random restriction with $*$-probability $p = \Theta(1/\log^{d-1}(M/\varepsilon))$. Note that t and p both depend on ε.

As alluded to above, the recent multi-switching lemma of [35] shows, remarkably, that essentially the same simplification holds under a random restriction with $*$-probability $p = \Theta(1/\log^{d-1}(M))$, independent of ε. Let us establish some terminology and notation to present these results.

▶ **Definition 5** (Common partial decision tree). *Let $\mathcal{F} = \{F_1, \dots, F_M\}$ be a collection of Boolean functions. We say that a decision tree T is a* common ℓ-partial decision tree for $\mathcal{F}$ *if every $F_i \in \mathcal{F}$ can be expressed as T with depth-ℓ decision trees at its its leaves. (Equivalently, for every $F_i \in \mathcal{F}$ and root-to-leaf path π in T, we have that $F_i \restriction \pi$ is computed by a depth-ℓ decision tree.)*

The multi-switching lemma of [35] is as follows:

▶ **Theorem 6** (Multi-switching lemma, Lemma 3.8 of [35]). *Let $\mathcal{F} = \{F_1, \dots, F_M\}$ be a collection of k-CNFs and $\ell := \log(2M)$. Then for all $t \geq 1$,*

$$\Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_p}[\mathcal{F} \restriction \boldsymbol{\rho} \text{ does not have a common } \ell := \log(2M)\text{-partial DT of depth } t] \leq M(24pk)^t.$$

The following corollary should be contrasted with Corollary 4:

▶ **Corollary 7** (AC^0 depth reduction via Theorem 6; c.f. Corollary 4). *Let $\mathcal{C}$ be a size-M depth-d AC^0 circuit with bottom fan-in k, and let $p = 1/(48k)$. Then for all $\varepsilon > 0$, the probability (over $\boldsymbol{\rho} \leftarrow \mathcal{R}_p$) that $\mathcal{C} \restriction \boldsymbol{\rho}$ is not computed by a $((\log(M/\varepsilon), \mathsf{AC}^0(\text{depth } d-1, \text{ bottom fan-in } \log(2M))$-decision tree is at most ε.*

Proof. This follows by applying Theorem 6 with $\mathcal{F}$ being the bottom-layer depth-2 subcircuits of $\mathcal{C}$ and $t = \log(M/\varepsilon)$, along with the fact that a depth-ℓ decision tree can be expressed as both a ℓ-DNF and an ℓ-CNF. ◀

We highlight a crucial qualitative aspect of Corollary 7: while the depth $t = \log(M/\varepsilon)$ of the decision tree whose existence it asserts does depend on ε, the depth-$(d-1)$ AC^0 circuits at its leaves have bottom fan-in $k = \log(2M)$ which does *not* depend on ε. This means that in successive application of Corollary 7, the values of $p = 1/(48k) = \Theta(1/\log M)$ will remain independent of ε. This leads to much better quantitative bounds than can be obtained through repeated applications of Corollary 4: $d-1$ iterative applications of Corollary 7 imply that every size-M depth-d AC^0 circuit collapses to a depth-$O(2^d \log(M/\varepsilon))$ decision tree with high probability, at least $1-\varepsilon$, under a random restriction with $*$-probability $p = \Theta(1/\log^{d-1} M)$. Note that the overall $*$-probability p is independent of ε.

Multi-switching lemmas and sparse $\mathbb{F}_2$ polynomials. The qualitative advantage of multi-switching lemmas – in particular, the crucial role of a common partial decision tree – can also be seen within the context of $\mathbb{F}_2$ polynomials.

Let P be an S-sparse $\mathbb{F}_2$ polynomial. It is an easy observation that P becomes a low-degree polynomial with high probability when hit with a random restriction: for all $\varepsilon, p \in (0,1)$ and $k \in \mathbb{N}$,

$$\mathop{\mathbf{Pr}}_{\boldsymbol{\rho} \leftarrow \mathcal{R}_{\frac{p}{2}}} [\, P \restriction \boldsymbol{\rho} \text{ is not a degree-}k \text{ polynomial} \,] \leq \frac{\varepsilon}{2} + S\binom{w}{k} p^k \quad \text{where } w = \Theta(\log(S/\varepsilon)). \quad (1)$$

(The proof follows by considering each monomial of P individually and taking a union bound over all S of them. For a fixed monomial, the probability that more than $\Omega(\log(S/\varepsilon))$ variables survive a random restriction from $\mathcal{R}_{\frac{1}{2}}$ is at most $\varepsilon/(2S)$; next, the probability that at least k variables in a width-w monomial survive a random restriction from $\mathcal{R}_p$ is at most $\binom{w}{k}p^k$.) The failure probability of (1) can be made at most ε by choosing p and k appropriately, but note that at least one of p (the $*$-probability) or k (the degree of the resulting polynomial) must depend on ε.

Using a slight extension of the ideas in the multi-switching lemmas of [35], we can instead bound the probability that $P \restriction \boldsymbol{\rho}$ becomes a *depth-t decision tree with degree-k polynomials at its leaves*. While this provides weaker structural information than the simple observation above (cf. Corollary 4 vs. Corollary 7 in the context of AC^0), the crucial win will come from the fact that p and k can *both* be taken to be independent of the failure probability ε (and only t will depend on ε).

3.1 Canonical common ℓ-partial decision trees

An important concept in the proof of Theorem 6 is that of a *canonical* common ℓ-partial decision tree for an ordered collection $\mathcal{F}$ of k-CNFs, which we define in this section.

Given a k-CNF formula F (which we view as an ordered sequence of width-k clauses $C_1 \wedge C_2 \wedge \cdots$), we recall the notion of the *canonical decision tree* for F, denoted $\mathrm{CDT}(F)$. This is a decision tree which computes F and is obtained as follows:

- If any clause C_i is identically-0, then the tree is the constant 0.
- If every clause C_i is identically-1, then the tree is the constant 1.
- Otherwise, let C_{i_1} be the first clause that is not identically-1, and let $\kappa \in [k]$ be the number of variables in C_{i_1}. The first κ levels of $\mathrm{CDT}(F)$ exhaustively query these κ variables. At each of the 2^κ resulting leaves of the tree (each one corresponding to some restriction $\eta \in \{0,1\}^\kappa$ fixing those κ variables), recursively put down the canonical decision tree $\mathrm{CDT}(F \restriction \eta)$.

We observe that the tree $\mathrm{CDT}(F)$ is unique given a fixed ordering $C_1, C_2, \ldots$ of the clauses in F.

Håstad's proof of his original switching lemma (Theorem 3) actually shows that if F is a k-CNF, then the canonical decision tree $\mathrm{CDT}(F \restriction \boldsymbol{\rho})$ is shallow w.h.p. over $\boldsymbol{\rho} \leftarrow \mathcal{R}_p$. This is crucially important for the arguments of Trevisan and Xue [65], who give a *derandomized* version of Håstad's original switching lemma: they construct a pseudorandom distribution over restrictions to take the place of $\mathcal{R}_p$, and show that with high probability a restriction drawn from this pseudorandom distribution causes a k-CNF to collapse to a small-depth decision tree. Their argument uses the structure of a canonical decision tree in an essential way.

Turning to Håstad's multi-switching lemma [35], we observe that analogous to his original switching lemma, the proof of Theorem 6 given in [35] implicitly establishes a stronger statement: $\mathcal{F} \restriction \boldsymbol{\rho}$ has a small-depth *canonical* common ℓ-partial decision tree w.h.p. over $\boldsymbol{\rho} \leftarrow \mathcal{R}_p$. In fact, we will use the fact that it actually establishes an even stronger statement: w.h.p. over $\boldsymbol{\rho} \leftarrow \mathcal{R}_p$, *every* canonical common ℓ-partial decision tree for $\mathcal{F} \restriction \boldsymbol{\rho}$ is shallow – as we explain below, there is more than one canonical common ℓ-partial decision tree for a sequence $\mathcal{F}$ of CNFs.

Let us explain what a canonical common ℓ-partial decision tree for a sequence of CNFs $\mathcal{F}$ is. We will see that there is a set of canonical common ℓ-partial decision trees for a given $\mathcal{F}$ rather than just one tree; note that this is the case even though we assume a fixed ordering $F_1, F_2, \ldots$ on the elements of $\mathcal{F}$ as well as on the clauses within each CNF. (Observe the contrast with the case of a canonical decision tree for a single formula F, where we assume a fixed ordering on the clauses of F; in that setting, as explained above there is a single canonical decision tree $\mathrm{CDT}(F)$.)

We need a preliminary definition to handle a technical issue related to the final segment of paths through a canonical decision tree.

▶ **Definition 8** (Full paths in the CDT). *Let $F = C_1 \wedge C_2 \wedge \cdots$ be a k-CNF and consider the canonical decision tree $\mathrm{CDT}(F)$ for F. Every path η in $\mathrm{CDT}(F)$ can be written as the the disjoint union of segments $\eta = \eta^{(1)} \circ \eta^{(2)} \circ \cdots \circ \eta^{(u)}$, where for all $j \in [u]$, the segment $\eta^{(j)}$ is an assignment to the surviving variables in the restricted clause $C_{i_j} \restriction \eta^{(1)} \circ \cdots \circ \eta^{(j-1)}$, and C_{i_j} is the first clause in $F \restriction \eta^{(1)} \circ \cdots \circ \eta^{(j-1)}$ that is not identically-1.*

Furthermore, note that for $j \in [u-1]$, the segment $\eta^{(j)}$ is in fact an assignment fixing all *the surviving variables in $C_{i_j} \restriction \eta^{(1)} \circ \cdots \circ \eta^{(j-1)}$. We say that η is* full *if this is also the case for the final segment: η is* full *if $\eta^{(u)}$ is an assignment fixing all the surviving variables in $C_{i_u} \restriction \eta^{(1)} \circ \cdots \circ \eta^{(u-1)}$.*

▶ **Observation 9.** *Let F be a k-CNF and suppose $\mathrm{depth}(\mathrm{CDT}(F)) > \ell$. Then there is a full path η of length $|\eta| \in \{\ell+1, \ldots, \ell+k\}$ in $\mathrm{CDT}(F)$.*

To help minimize confusion, we will reserve "η" for paths or segments of paths in CDTs, and "π" for paths (or segments of paths) in CCDTs.

We are now ready to define the set of canonical common ℓ-partial decision trees:

▶ **Definition 10** (Canonical common ℓ-partial DT). *Let $\mathcal{F} = (F_1, \ldots, F_M)$ be an ordered collection of k-CNFs. The set of all* canonical *common ℓ-partial decision trees for $\mathcal{F}$, which we denote $\mathrm{CCDT}_\ell(\mathcal{F})$, is defined inductively as follows:*

0. *If $M = 0$ (i.e. $\mathcal{F}$ is an empty collection of k-CNFs) then $\mathrm{CCDT}_\ell(\mathcal{F})$ contains a single tree, the empty tree with no nodes. (Note that otherwise $M \geq 1$, so there is some first formula F_1 in $\mathcal{F}$.)*

1. *If $\mathrm{CDT}(F_1) \leq \ell$, then $\mathrm{CCDT}_\ell(\mathcal{F})$ is simply $\mathrm{CCDT}_\ell(\mathcal{F}')$, where $\mathcal{F}' = (F_2, \ldots, F_M)$. (Note that in this case, since inductively each tree in $\mathrm{CCDT}_\ell(\mathcal{F}')$ is a common ℓ-partial DT for $\mathcal{F}'$, each such tree is also a common ℓ-partial DT for $\mathcal{F}$.)*

2. *Otherwise, since $\mathrm{CDT}(F_1) > \ell$ there must be a witnessing* full *path η of length between $\ell+1$ and $\ell+k$ in $\mathrm{CDT}(F_1)$, and there are at most $2^{\ell+k}$ such witnessing full paths. Let P be the set of all such witnessing full paths. For each path $\eta \in P$, let T_η be the tree of depth $|\eta|$ obtained by exhaustively querying all the variables in η in the first $|\eta|$ levels. Recurse at the end of each path in T_η: for each path π in T_η, attach a tree T' from $\mathrm{CCDT}_\ell(\mathcal{F} \restriction \pi)$ at the end of the path. So in this case $\mathrm{CCDT}_\ell(\mathcal{F})$ is the set of all trees that can be obtained in this way (across all possible choices of $\eta \in P$ and all possible choices of a tree $T' \in \mathrm{CCDT}_\ell(\mathcal{F} \restriction \pi)$ for each path $\pi \in T_\eta$).*

We write $\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F}))$ to denote the maximum depth of any tree in the set $\mathrm{CCDT}_\ell(\mathcal{F})$.

The following slight variant of Theorem 6 can be extracted, with some effort, from a slight modification of the proof given in [35], which we provide in the full version:

▶ **Theorem 11** (Slight variant of Håstad's multi-switching lemma. Theorem 6). *Let $\mathcal{F} = (F_1, \ldots, F_M)$ be an ordered collection of k-CNFs. Then for all $\ell, t \geq 1$,*

$$\Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_p}[\,\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \boldsymbol{\rho})) \geq t\,] \leq M^{\lceil t/\ell \rceil}(32pk)^t.$$

A comparison of Theorem 6 (Håstad's multi-switching lemma) and Theorem 11 (our variant of it). We emphasize that the differences are technical in nature, and all the ideas in our proof of Theorem 11 are from [35]. First, we observe that ℓ is now a free parameter rather than being fixed to $\log(2M)$; this flexibility will be necessary in our PRG construction for sparse $\mathbb{F}_2$ polynomials (where we take $\ell = \Theta(\sqrt{\log M})$). Second, our notion of a canonical common partial decision tree differs slightly from the one that is implicit in [35]: in case 2 of Definition 10, we query a witnessing full path of length between $\ell+1$ and $\ell+k$, whereas [35] queries any witnessing path of length greater than ℓ.

4 A pseudorandom multi-switching lemma

As suggested earlier, the crux of our PRG construction is a *derandomization* of the multi-switching lemma of Theorem 11: we devise a suitable *pseudorandom* distribution over random restrictions in place of $\mathcal{R}_p$ (the truly random distribution over restrictions) and show that a random restriction $\boldsymbol{\rho}$ drawn from this pseudorandom distribution satisfies a similar guarantee to Theorem 11.

Our derandomization of Theorem 11 is largely influenced by Trevisan and Xue's [65] ingenious derandomization of Håstad's original switching lemma (Theorem 3). Roughly speaking, we will derandomize the multi-switching lemma of Theorem 11 by "fooling its proof": we will show that the proof of Theorem 11 (given in the full version, which we again emphasize is only a slight technical modification of Håstad's proof of his multi-switching lemma, Theorem 6) "cannot δ-distinguish" between truly random restrictions and pseudorandom restrictions drawn from polylog(n)-wise independent distributions. Since Theorem 11 holds for truly random restrictions, it thus follows that it also holds for pseudorandom restrictions drawn from polylog(n)-wise independent distributions (up to a δ additive loss in the failure probability).

To accomplish this, we exploit the "computational simplicity" of Theorem 11's proof: for a fixed family $\mathcal{F}$ of k-CNF formulas, we will show that there is a small AC^0 circuit that takes as input an encoding of a restriction ρ, and outputs 1 iff ρ is a bad restriction for the desired conclusion of Theorem 11, contributing to its failure probability (i.e. iff $\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho)) > t$). As alluded to in Section 3.1, this relies on the fact that Theorem 11 does not simply bound the depth of the *optimal* common ℓ-partial decision tree for $\mathcal{F} \restriction \boldsymbol{\rho}$, but instead the depth of any *canonical* common ℓ-partial decision tree for $\mathcal{F} \restriction \boldsymbol{\rho}$. Indeed, this "constructive" aspect of the proof is crucial for our derandomization strategy: it is not at all clear that there is a small circuit for checking if the *optimal* common ℓ-partial decision tree for $\mathcal{F} \restriction \rho$ has depth greater than t.

It will be convenient for us to represent restrictions $\rho \in \{0,1,*\}^n$ as bitstrings $(\varrho, y) \in \{0,1\}^{n\times q} \times \{0,1\}^n := \{0,1\}^{Y_q}$, where $q \in \mathbb{N}$ is a parameter.

▶ **Definition 12** (Representing restrictions as bitstrings). *We associate with each string $(\varrho, y) \in \{0,1\}^{Y_q}$ the restriction $\rho(\varrho, y) \in \{0,1,*\}^n$ defined as follows:*

$$\rho(\varrho, y)_i = \begin{cases} * & \text{if } \varrho_{i,1} = \cdots = \varrho_{i,q} = 1 \\ y_i & \text{otherwise.} \end{cases}$$

The following observation explains the role of q:

▶ **Observation 13.** *Let $(\boldsymbol{\varrho}, \boldsymbol{y})$ be drawn from the uniform distribution over $\{0,1\}^{Y_q}$. Then the random restriction $\rho(\boldsymbol{\varrho}, \boldsymbol{y}) \in \{0,1,*\}^n$ is distributed according to $\mathcal{R}_p$ where $p = 2^{-q}$.*

Now we are ready to state our pseudorandom multi-switching lemma:

▶ **Theorem 14** (Derandomized version of Theorem 11). *Let $\mathcal{F} = (F_1, \ldots, F_M)$ be an ordered list of Q-clause k-CNFs. Let $\delta, p \in (0,1)$ and define $q = \log(1/p)$. Let $\mathcal{D}$ be any distribution over $\{0,1\}^{Y_q}$ that $(\delta/(M^{\lceil t/\ell \rceil} n^{O(t)}))$-fools the class of depth-3 circuits of size $M(n^{O(\ell)} + Q2^{O(kq)})$. Then for all $\ell \geq k$ and all $t \in \mathbb{N}$,*

$$\Pr_{(\boldsymbol{\eta},\boldsymbol{z}) \leftarrow \mathcal{D}} \left[\operatorname{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\boldsymbol{\eta}, \boldsymbol{z}))) \geq t \right] \leq 16^{t+\ell} M^{\lceil t/\ell \rceil} (32pk)^t + \delta.$$

In the full version of the paper we prove this lemma and show how it, along with other ingredients, yields our circuit complexity derandomization results.

References

1 Scott Aaronson. A Counterexample to the Generalized Linial–Nisan Conjecture. *Electronic Colloquium on Computational Complexity*, 17:109, 2010.

2 Scott Aaronson. BQP and the polynomial hierarchy. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 141–150, 2010.

3 Manindra Agrawal, Eric Allender, Russell Impagliazzo, Toniann Pitassi, and Steven Rudich. Reducing the complexity of reductions. *Comput. Complexity*, 10(2):117–138, 2001.

4 Miklós Ajtai. Σ^1_1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.

5 Miklós Ajtai. Geometric properties of sets defined by constant depth circuits. In *Combinatorics, Paul Erdős is eighty, Vol. 1*, Bolyai Soc. Math. Stud., pages 19–31. János Bolyai Math. Soc., Budapest, 1993.

6 Miklós Ajtai and Avi Wigderson. Deterministic Simulation of Probabilistic Constant Depth Circuits. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 11–19, 1985.

7 Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992.

8 László Babai. Random oracles separate PSPACE from the polynomial-time hierarchy. *Information Processing Letters*, 26(1):51–53, 1987.

9 László Babai, Noam Nisan, and Márió Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. System Sci.*, 45(2):204–232, 1992. doi:10.1016/0022-0000(92)90047-M.

10 Marshall Ball, Dana Dachman-Soled, Siyao Guo, Tal Malkin, and Li-Yang Tan. Non-malleable codes for small-depth circuits. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2018. To appear.

11 Louay Bazzi. Polylogarithmic independence can fool DNF formulas. *SIAM Journal on Computing*, 38(6):2220–2272, 2009.

12 Paul Beame. Lower bounds for recognizing small cliques on CRCW PRAM's. *Discrete Applied Mathematics*, 29(1):3–20, 1990.

13 Paul Beame, Russell Impagliazzo, and Srikanth Srinivasan. Approximating AC^0 by Small Height Decision Trees and a Deterministic Algorithm for $\#\mathsf{AC}^0$-SAT. In *Proceedings of the 27th IEEE Conference on Computational Complexity (CCC)*, pages 117–125, 2012.

14 Manuel Blum and Silvio Micali. How to Generate Cryptographically Strong Sequences of Pseudo Random Bits. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 112–117, 1982.

15 Andrej Bogdanov. Pseudorandom generators for low degree polynomials. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–30. SIAM, 2005. doi:10.1145/1060590.1060594.

16 Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM J. Comput.*, 39(6):2464–2486, 2010. doi:10.1137/070712109.

17 Jean Bourgain. Estimation of certain exponential sums arising in complexity theory. *Comptes Rendus Mathematique*, 340(9):627–631, 2005. doi:10.1016/j.crma.2005.03.008.

18 Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *Journal of the ACM*, 57(5):28, 2010.

19 Jin-Yi Cai. With Probability One, a Random Oracle Separates PSPACE from the Polynomial-Time Hierarchy. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–29, 1986.

20 Arkadev Chattopadhyay. Discrepancy and the power of bottom fan-in in depth-three circuits. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 449–458, 2007.

21 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom Generators from Polarizing Random Walks. In *33rd Computational Complexity Conference, CCC*, pages 1:1–1:21, 2018.

22 Eshan Chattopadhyay, Pooya Hatami, Shachar Lovett, and Avishay Tal. Pseudorandom Generators from the Second Fourier Level and Applications to AC^0 with Parity Gates. In *10th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 22:1–22:15, 2019.

23 Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1171–1184. ACM, 2017.

24 Shiva Chaudhuri and Jaikumar Radhakrishnan. Deterministic restrictions in circuit complexity. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 30–36, 1996.

25 Xi Chen, Igor Carboni Oliveira, Rocco A. Servedio, and Li-Yang Tan. Near-optimal small-depth lower bounds for small distance connectivity. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, pages 612–625, 2016.

26 Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved pseudorandom generators for depth 2 circuits. In *Proceedings of the 13th International Workshop on Randomization and Computation (RANDOM)*, pages 504–517, 2010.

27 Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-Malleable Codes. *Journal of the ACM (JACM)*, 65(4):20, 2018.

28 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS)*, pages 468–483. ACM, 2012.

29 Merrick Furst, James Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

30 Oded Goldreich and Avi Widgerson. On derandomizing algorithms that err extremely rarely. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 109–118, 2014.

31 Parikshit Gopalan, Raghu Meka, and Omer Reingold. DNF sparsification and a faster deterministic counting algorithm. *Comput. Complexity*, 22(2):275–310, 2013. doi:10.1007/s00037-013-0068-6.

32 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Better Pseudorandom Generators from Milder Pseudorandom Restrictions. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 120–129, 2012.

33 Prahladh Harsha and Srikanth Srinivasan. On Polynomial Approximations to AC^0. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2016*, pages 32:1–32:14, 2016.

34 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing (STOC)*, pages 6–20, 1986.

35 Johan Håstad. On the Correlation of Parity and Small-Depth Circuits. *SIAM Journal on Computing*, 43(5):1699–1708, 2014.

36 Johan Håstad. An Average-Case Depth Hierarchy Theorem for Higher Depths. In *Proceedings of the 57th Annual Symposium on Foundations of Computer Science (FOCS)*, 2016.

37 Johan Håstad. On small-depth Frege proofs for Tseitin for grids. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 97–108. IEEE Computer Society, 2017.

38 Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Comput. Complexity*, 1(2):113–129, 1991. doi:10.1007/BF01272517.

39 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In *Proceedings of the 23rd Annual Symposium on Discrete Algorithms (SODA)*, pages 961–972, 2012.

40 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *Proceedings of the 53rd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 111–119. IEEE Computer Society, 2012.

41 Adam Klivans. On the Derandomization of Constant Depth Circuits. In *Proceedings of 5th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 249–260, 2001.

42 Adam Klivans, Homin Lee, and Andrew Wan. Mansour's Conjecture is True for Random DNF Formulas. In *Proceedings of the 23rd Conference on Learning Theory (COLT)*, pages 368–380, 2010.

43 Jan Krajíček, Pavel Pudlák, and Alan Woods. An exponential lower bound to the size of bounded depth Frege proofs of the pigeonhole principle. *Random Structures & Algorithms*, 7(1):15–39, 1995.

44 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

45 Nathan Linial and Noam Nisan. Approximate inclusion-exclusion. *Combinatorica*, 10(4):349–365, 1990.

46 Shachar Lovett. Unconditional pseudorandom generators for low-degree polynomials. *Theory Comput.*, 5:69–82, 2009. doi:10.4086/toc.2009.v005a003.

47 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size AC^0 circuits with $n^{1-o(1)}$ symmetric gates. In *Approximation, randomization, and combinatorial optimization*, volume 6845 of *Lecture Notes in Comput. Sci.*, pages 640–651. Springer, Heidelberg, 2011. doi:10.1007/978-3-642-22935-0_54.

48 Chi-Jen Lu. Hitting set generators for sparse polynomials over any finite fields. In *Proceedings of the 27th IEEE Conference on Computational Complexity (CCC)*, pages 280–286, 2012. doi:10.1109/CCC.2012.20.

49 Michael Luby and Boban Veličković. On deterministic approximation of DNF. *Algorithmica*, 16(4-5):415–433, 1996. doi:10.1007/s004539900054.

50 Michael Luby, Boban Veličković, and Avi Wigderson. Deterministic approximate counting of depth-2 circuits. In *Proceedings of the 2nd ISTCS*, pages 18–24, 1993.

51 Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993. doi:10.1137/0222053.

52 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

53 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *J. Comput. System Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

54 Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Computational complexity*, 3(2):97–140, 1993.

55 Toniann Pitassi, Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. Poly-logarithmic Frege depth lower bounds via an expander switching lemma. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, pages 644–657, 2016.

56 Alexander Razborov. A simple proof of Bazzi's theorem. *ACM Transactions on Computation Theory*, 1(1):3, 2009.

57 Benjamin Rossman. On the constant-depth complexity of k-clique. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 721–730, 2008.

58 Benjamin Rossman. The Average Sensitivity of Bounded-Depth Formulas. In *Proceedings of the 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 424–430, 2015.

59 Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An Average-Case Depth Hierarchy Theorem for Boolean Circuits. In *Proceedings of the 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1030–1048, 2015.

60 Rocco A. Servedio and Li-Yang Tan. What circuit classes can be learned with nontrivial savings? In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS)*, 2017.

61 Rocco A. Servedio and Li-Yang Tan. Luby–Veličković–Wigderson revisited: Improved correlation bounds and pseudorandom generators for depth-two circuits. In *Proceedings of the 22nd International Workshop on Randomization and Computation (RANDOM)*, pages 56:1–56:20, 2018.

62 Jirí Síma and Stanislav Zák. A Polynomial Time Construction of a Hitting Set for Read-Once Branching Programs of Width 3. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:88, 2010.

63 Avishay Tal. Tight Bounds on the Fourier Spectrum of AC^0. In *Proceedings of the 32nd Computational Complexity Conference (CCC)*, pages 15:1–15:31, 2017. doi:10.4230/LIPIcs.CCC.2017.15.

64 Luca Trevisan. A note on approximate counting for k-DNF. In *Proceedings of the 8th International Workshop on Randomization and Computation (RANDOM)*, pages 417–426, 2004.

65 Luca Trevisan and Tongke Xue. A derandomized switching lemma and an improved derandomization of AC^0 . In *Proceedings of the 28th IEEE Conference on Computational Complexity (CCC)*, pages 242–247, 2013.

66 Emanuele Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM J. Comput.*, 36(5):1387–1403, 2007. doi:10.1137/050640941.

67 Emanuele Viola. *On the power of small-depth computation*. Now Publishers Inc, 2009.

68 Emanuele Viola. The sum of d small-bias generators fools polynomials of degree d. *Comput. Complexity*, 18(2):209–217, 2009. doi:10.1007/s00037-009-0273-5.

69 Emanuele Viola and Avi Wigderson. Norms, XOR Lemmas, and Lower Bounds for Polynomials and Protocols. *Theory of Computing*, 4(7):137–168, 2008. doi:10.4086/toc.2008.v004a007.

70 Andrew Yao. Theory and applications of trapdoor functions. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982.

71 Andrew Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1–10, 1985.

A Proof of Theorem 14

A.1 Bad restrictions and the structure of witnessing paths

Fix $\mathcal{F} = (F_1, \ldots, F_M)$. We say that a restriction $\rho \in \{0, 1, *\}^n$ is *bad* if

$$\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho)) \geq t.$$

Fix ρ to be a bad restriction. Recalling our definition of the set of canonical common partial decision trees (Definition 10), there exists a tree $T \in \mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho)$ and a path Π of length exactly t through T. Furthermore, we have that

1. There exist indices $1 \le i_1 \le i_2 \le \cdots \le i_u \le M$ where $u \le \lceil t/\ell \rceil$, and
2. $\Pi = \pi^{(1)} \circ \cdots \circ \pi^{(u)}$, where for all $j \in [u]$, we have that $\mathrm{supp}(\pi^{(j)}) = \mathrm{supp}(\eta^{(j)})$ where $\eta^{(j)}$ is a path through the canonical decision tree

$$\mathrm{CDT}(F_{i_j} \restriction \rho \circ \pi^{(1)} \circ \cdots \circ \pi^{(j-1)}).$$

Furthermore, for every $j \in [u-1]$ we have that $\eta^{(j)}$ is a full path of length between $\ell+1$ and $\ell+k$ through the CDT, and $\eta^{(u)}$ is a path of length exactly $t - \sum_{j=1}^{u-1} |\mathrm{supp}(\eta^{(j)})|$. (Note that $\eta^{(u)}$ is not necessarily a full path.)

(Note that by (2), these subpaths $\pi^{(j)}$ of Π are supported on mutually disjoint sets of coordinates.) With this structure of Π in mind, we make the following definition:

▶ **Definition 15** ($\mathcal{F}$-traversal). *Let $\mathcal{F} = (F_1, \ldots, F_M)$ be an ordered list of CNFs. An ℓ-segmented $\mathcal{F}$-traversal of length t is a tuple $P = (\mathcal{I}, \{S_1, \ldots, S_u\}, \Pi, \mathrm{H})$ comprising:*

1. *An ordered list of indices $\mathcal{I} = (i_1, \ldots, i_u)$ where $1 \le i_1 \le \cdots \le i_u \le M$ and $u \le \lceil t/\ell \rceil$,*
2. *For each index $i_j \in \mathcal{I}$, a subset $S_j \subseteq [n]$ such that*
   a. *These sets are mutually disjoint: $S_j \cap S_{j'} = \emptyset$ for all $j \ne j'$.*
   b. *For $1 \le j \le u-1$, each S_j has size between $\ell+1$ and $\ell+k$, and S_u has size exactly $t - \sum_{j=1}^{u-1} |\mathrm{supp}(\eta^{(j)})|$.*

   (Consequently $|S_1 \cup \cdots \cup S_u| = t$.)
3. *An assignment $\Pi = \pi^{(1)} \circ \cdots \circ \pi^{(u)}$ to the variables in $S_1 \cup \cdots \cup S_u$, where*

$$\pi^{(j)} : \{0,1\}^{S_j} \to \{0,1\} \qquad \text{for } 1 \le j \le u.$$

4. *An assignment $\mathrm{H} = \eta^{(1)} \circ \cdots \circ \eta^{(u)}$ to the variables in $S_1 \cup \cdots \cup S_u$, where again*

$$\eta^{(j)} : \{0,1\}^{S_j} \to \{0,1\} \qquad \text{for } 1 \le j \le u.$$

By our discussion above, for any restriction $\rho \in \{0,1,*\}^n$ and any tree $T \in \mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho)$, every path Π of length t through $\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho)$ uniquely induces an ℓ-segmented $\mathcal{F}$-traversal P of length t. We say that P *occurs in* $\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho)$ if it is induced by some path Π of length t through T for some $T \in \mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho)$.

Definition 15 immediately yields the following:

▶ **Proposition 16** (Number of $\mathcal{F}$-traversals). *Fix an ordered list $\mathcal{F} = (F_1, \ldots, F_M)$ of k-CNFs, and let $\mathcal{P}_{\mathcal{F},\ell,t}$ denote the collection of all ℓ-segmented $\mathcal{F}$-traversals of length t. Then*

$$|\mathcal{P}_{\mathcal{F},\ell,t}| \le M^{\lceil t/\ell \rceil} n^{O(t)}.$$

A.2 A small AC^0 circuit for recognizing bad restrictions

We begin by showing that for every $\mathcal{F}$-traversal $P = (\mathcal{I}, \{S_1, \ldots, S_u\}, \Pi, \mathrm{H})$, there is a small circuit $\mathcal{C}_P$ over $\{0,1\}^{Y_q}$ that outputs 1 on input $(\varrho, y) \in \{0,1\}^{Y_q}$ iff P occurs in $\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\varrho, y))$. Since

$$\begin{aligned}\rho(\varrho, y) \text{ is bad} &\iff \mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\varrho, y))) \ge t \\ &\iff \exists\ \ell\text{-segmented } \mathcal{F}\text{-traversal } P \text{ of length } t \text{ occurring in } \mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\varrho, y)),\end{aligned}$$

by considering

$$\mathcal{C}_{\mathcal{F},\ell,t}(\varrho, y) := \bigvee_{P \in \mathcal{P}_{\mathcal{F},\ell,t}} \mathcal{C}_P(\varrho, y) \tag{2}$$

we have that

$$\rho(\varrho, y) \text{ is bad} \iff \mathcal{C}_{\mathcal{F},\ell,t}(\varrho, y) = 1.$$

▷ Claim 17 (Circuit for a single $\mathcal{F}$-traversal). Let $P = (\mathrm{I}, \{S_1, \ldots, S_u\}, \Pi, \mathrm{H})$ be an ℓ-segmented $\mathcal{F}$-traversal of length t. There is a depth-3 AND-OR-AND circuit $\mathcal{C}_P : \{0,1\}^{Y_q} \to \{0,1\}$ of size $M(n^{O(\ell)} + Q2^{O(kq)})$ such that

$$\forall\, (\varrho, y) \in \{0,1\}^{Y_q} : \quad \mathcal{C}_P(\varrho, y) = 1 \iff P \text{ occurs in } \mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\varrho, y))$$

Proof. Our circuit $\mathcal{C}_P$ will be the AND of M many depth-3 subcircuits of size $n^{O(\ell)}$, one for each k-CNF $F \in \mathcal{F}$. As we will explain later, each of these subcircuits is one of two types. We first describe these two types of "candidate subcircuits", and then explain precisely which M subcircuits of each type are AND-ed together to give $\mathcal{C}_P$. (Both these types of circuits are implicit in the work of [65].)

1. **First type: Circuits checking that a particular restriction η is a path in a particular CDT.** We claim that for any Q-clause k-CNF $F' = C_1 \wedge \cdots \wedge C_Q$ and restriction η, there is a $Q2^{O(kq)}$-clause $O(kq)$-CNF G over $\{0,1\}^{Y_q}$ that outputs 1 on input (ϱ, y) iff η is a path in $\mathrm{CDT}(F' \restriction \rho(\varrho, y))$.
   For each $i \in [Q]$, we write Fixed_i to denote the set

   $$\{j \in [n] : j \in \eta^{-1}(\{0,1\}) \text{ and } x_j \text{ occurs in } C_i\}$$

   of all variables that are fixed by η and occur in C_i. We write $\sigma^{(i)} \in \{0,1\}^{\mathrm{Fixed}_i}$ to denote η restricted to the coordinates in Fixed_i. It is straightforward to verify that η is a path in $\mathrm{CDT}(F' \restriction \rho(\varrho, y))$ iff for all $i \in [Q]$ such that $\mathrm{Fixed}_1 \cup \cdots \cup \mathrm{Fixed}_{i-1} \subsetneq \mathrm{supp}(\eta)$,
   - **a.** If $\mathrm{Fixed}_i \setminus (\mathrm{Fixed}_1 \cup \cdots \cup \mathrm{Fixed}_{i-1}) = \emptyset$ then the clause C_i is satisfied by $\rho(\varrho, y) \circ \sigma^{(1)} \circ \cdots \circ \sigma^{(i-1)}$. (Hence this clause does not contribute to $\mathrm{CDT}(F' \restriction \rho(\varrho, y))$; it is "skipped" in the canonical decision tree construction process.)
   - **b.** Otherwise, writing $\mathrm{Fixed}'_i := \mathrm{Fixed}_i \setminus (\mathrm{Fixed}_1 \cup \cdots \cup \mathrm{Fixed}_{i-1})$,
     - **i.** $\rho(\varrho, y)_j = *$ for all $j \in \mathrm{Fixed}'_i$, and
     - **ii.** $\rho(\varrho, y) \circ \sigma_1 \circ \cdots \circ \sigma_{i-1}$ falsifies all the remaining literals in C_i and are not in Fixed'_i.

     In other words, the clause

     $$C_i \restriction \rho(\varrho, y) \circ \sigma^{(1)} \circ \cdots \circ \sigma^{(i-1)}$$

     is not satisfied and its surviving variables are precisely those in Fixed'_i. (Hence the variables in Fixed'_i are exactly those queried by the canonical decision tree construction process when it reaches C_i.)

   Since both conditions (a) and (b) depend only on the coordinates of $\rho(\varrho, y)$ that occur in C_i (at most k such coordinates since C_i has width at most k), and hence at most $k(q+1)$ coordinates of $(\varrho, y) \in \{0,1\}^{Y_q}$, it is clear that both conditions can be checked by a $2^{O(kq)}$-clause $O(kq)$-CNF over $\{0,1\}^{Y_q}$. The overall CNF G is simply the AND of all Q many of these CNFs, one for each clause C_i of F', and hence G is itself a $Q2^{O(kq)}$-clause $O(kq)$-width CNF.
2. **Second type: Circuits checking that a particular CDT has depth at most ℓ.** Next, we claim that for every Q-clause k-CNF F', there is a depth-3 AND-OR-AND circuit with fan-in sequence $((2n)^{\ell+1}, Q2^{O(kq)}, O(kq))$ that outputs 1 on input (ϱ, y) iff $\mathrm{depth}(\mathrm{CDT}(F' \restriction \rho(\varrho, y))) \leq \ell$.
   We establish this by showing that there is a depth-3 OR-AND-OR circuit Σ with the claimed fan-in sequence that outputs 1 on input (ϱ, y) if $\mathrm{depth}(\mathrm{CDT}(F' \restriction \rho(\varrho, y))) > \ell$; given such a circuit Σ, the desired AND-OR-AND circuit is obtained by negating Σ and using de Morgan's law. Certainly $\mathrm{depth}(\mathrm{CDT}(F' \restriction \rho(\varrho, y))) > \ell$ iff there is a path η of

length $\ell+1$ in $\mathrm{CDT}(F' \restriction \rho(\varrho, y))$. There are at most $(2n)^{\ell+1}$ many possible paths of length $\ell+1$ (every path is simply an ordered list of literals), and as argued in (1) above, for every path η there is a $Q2^{O(kq)}$-clause, $O(kq)$-CNF over $\{0,1\}^{Y_q}$ that checks if η is a path in $\mathrm{CDT}(F' \restriction \rho(\varrho, y))$. The overall circuit Σ is simply the OR of at most $(2n)^{\ell+1}$ such circuits, one for each path η.

With these two types of circuits in hand the overall circuit $\mathcal{C}_P$ is now easy to describe. $\mathcal{C}_P$ is the AND of M many depth-3 subcircuits, one for each k-CNF $F \in \mathcal{F}$:

- For each of the u indices $i_j \in \mathcal{I}$, a circuit of the first type that checks that $\eta^{(j)}$ is a path in $\mathrm{CDT}(F_{i_j} \restriction \rho(\varrho, y) \circ \pi^{(1)} \circ \cdots \circ \pi^{(j-1)})$ (recall from Definition 15 that $\eta^{(j)}$ is H restricted to the variables in S_j);
- For all $M-u$ other indices $i \in [M] \setminus \mathcal{I}$, a circuit of the second type that checks that $\mathrm{depth}(\mathrm{CDT}(F_i \restriction \rho(\varrho, y) \circ \pi^{(1)} \circ \cdots \circ \pi^{(i^-)})) \le \ell$, where $i^- = \max\{j \in [u] \colon i_j < i\}$.

The bound on the size of this overall circuit follows from a union bound over the sizes of the subcircuits given in (1) and (2) above. ◁

A.3 Putting the pieces together: Proof of Theorem 14

Recalling the definition (2) of $\mathcal{C}_{\mathcal{F},\ell,t}$,

$$\mathcal{C}_{\mathcal{F},\ell,t}(\varrho, y) := \bigvee_{P \in \mathcal{P}_{\mathcal{F},\ell,t}} \mathcal{C}_P(\varrho, y),$$

Proposition 16 giving a bound on its top fan-in, and Claim 17 giving a bound on the size of its subcircuits, we have shown the following:

▷ Claim 18 (Circuit for recognizing bad restrictions). Let $\mathcal{F} = (F_1, \ldots, F_M)$ be an ordered list of Q-clause k-CNFs, and let $\ell, t \ge 1$. There is a depth-4 circuit $\mathcal{C}_{\mathcal{F},\ell,t}$ over $\{0,1\}^{Y_q}$ such that

$$\mathcal{C}_{\mathcal{F},\ell,t}(\varrho, y) = 1 \iff \mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\varrho, y))) \ge t.$$

This circuit $\mathcal{C}_{\mathcal{F},\ell,t}$ is the OR of $M^u n^{O(t)}$ many depth-3 circuits of size $M(n^{O(\ell)} + Q2^{O(kq)})$.

The following observation will be useful for us:

▶ **Observation 19.** *Let $\mathcal{F} = (F_1, \ldots, F_M)$ be an ordered collection of k-CNFs. For $\ell \ge k$, the total number of paths Π such that Π is a path of length exactly t in some tree $T \in \mathrm{CCDT}_\ell(\mathcal{F})$ is at most $(2^{\ell+k} \cdot 2^{\ell+k})^{\lceil t/\ell \rceil} \le 16^{t+\ell}$. Consequently, if $(\varrho, y) \in \{0,1\}^{Y_q}$ is such that $\mathcal{C}_{\mathcal{F},\ell,t}(\varrho, y) = 1$, then $\mathcal{C}_P(\varrho, y) = 1$ for (at least one) and at most $16^{t+\ell}$ many ℓ-segmented $\mathcal{F}$-traversals P of length t.*

Proof. This follows by inspection of the recursive construction of the set $\mathrm{CCDT}_\ell(\mathcal{F})$ of canonical common ℓ-partial decision trees for $\mathcal{F}$. Each time case (2) of the definition is reached, the set P of witnessing full paths has size at most $2^{\ell+k}$, and for each path in P there are at most $2^{\ell+k}$ possible assignments to the variables on the path. Finally, there are at most $\lceil t/\ell \rceil$ levels of recursive calls. ◀

With Claim 18 and Observation 19 in hand, we are now ready to prove our main result of this section (Theorem 14), a derandomized version of the multi-switching lemma (Theorem 11). We restate Theorem 14 here for the reader's convenience:

▶ **Theorem 14.** *Let $\mathcal{F} = (F_1, \ldots, F_M)$ be an ordered list of Q-clause k-CNFs. Let $\delta, p \in (0,1)$ and define $q = \log(1/p)$. Let $\mathcal{D}$ be any distribution over $\{0,1\}^{Y_q}$ that $(\delta/(M^{\lceil t/\ell \rceil} n^{O(t)}))$-fools the class of depth-3 circuits of size $M(n^{O(\ell)} + Q2^{O(kq)})$. Then for all $\ell \geq k$ and all $t \in \mathbb{N}$,*

$$\Pr_{(\boldsymbol{\eta},\boldsymbol{z})\leftarrow\mathcal{D}}\left[\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\boldsymbol{\eta},\boldsymbol{z}))) \geq t\right] \leq 16^{t+\ell} M^{\lceil t/\ell \rceil} (32pk)^t + \delta.$$

Proof.

$$\begin{aligned}
&\Pr_{(\boldsymbol{\eta},\boldsymbol{z})\leftarrow\mathcal{D}}\left[\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\boldsymbol{\eta},\boldsymbol{z}))) \geq t\right] \\
&= \mathop{\mathbf{E}}_{(\boldsymbol{\eta},\boldsymbol{z})\leftarrow\mathcal{D}}\left[\mathcal{C}_{\mathcal{F},\ell,t}(\boldsymbol{\eta},\boldsymbol{z})\right] && \text{(Claim 18)} \\
&\leq \sum_{P\in\mathcal{P}_{\mathcal{F},\ell,t}} \mathop{\mathbf{E}}_{(\boldsymbol{\eta},\boldsymbol{z})\leftarrow\mathcal{D}}\left[\mathcal{C}_P(\boldsymbol{\eta},\boldsymbol{z})\right] && \text{(union bound)} \\
&\leq \sum_{P\in\mathcal{P}_{\mathcal{F},\ell,t}} \left(\mathop{\mathbf{E}}_{(\boldsymbol{\varrho},\boldsymbol{y})\leftarrow\mathcal{U}}[\mathcal{C}_P(\boldsymbol{\varrho},\boldsymbol{y})] + \frac{\delta}{M^{\lceil t/\ell \rceil} n^{O(t)}} \right) && (\mathcal{D}\ (\delta/(M^{\lceil t/\ell \rceil} n^{O(t)}))\text{-fools } \mathcal{C}_P) \\
&\leq \delta + \mathop{\mathbf{E}}_{(\boldsymbol{\varrho},\boldsymbol{y})\leftarrow\mathcal{U}}\left[\sum_{P\in\mathcal{P}_{\mathcal{F},\ell,t}} \mathcal{C}_P(\boldsymbol{\varrho},\boldsymbol{y})\right] && \text{(Proposition 16)} \\
&\leq \delta + 16^{t+\ell} \mathop{\mathbf{E}}_{(\boldsymbol{\varrho},\boldsymbol{y})\leftarrow\mathcal{U}}[\mathcal{C}_{\mathcal{F},\ell,t}(\boldsymbol{\varrho},\boldsymbol{y})] && \text{(Observation 19)} \\
&= \delta + 16^{t+\ell} \Pr_{(\boldsymbol{\varrho},\boldsymbol{y})\leftarrow\mathcal{U}}\left[\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \rho(\boldsymbol{\varrho},\boldsymbol{y}))) \geq t\right] && \text{(Claim 18)} \\
&= \delta + 16^{t+\ell} \Pr_{\boldsymbol{\rho}\leftarrow\mathcal{R}_p}\left[\mathrm{depth}(\mathrm{CCDT}_\ell(\mathcal{F} \restriction \boldsymbol{\rho})) \geq t\right] && \text{(Observation 13)} \\
&\leq \delta + 16^{t+\ell} M^{\lceil t/\ell \rceil} (32pk)^t. && \text{(Theorem 11)}
\end{aligned}$$

◀

Unconstraining Graph-Constrained Group Testing

Bruce Spang
Stanford University, CA, USA
bspang@stanford.edu

Mary Wootters
Stanford University, CA, USA
marykw@stanford.edu

Abstract

In network tomography, one goal is to identify a small set of failed links in a network using as little information as possible. One way of setting up this problem is called *graph-constrained group testing*. Graph-constrained group testing is a variant of the classical combinatorial group testing problem, where the tests that one is allowed are additionally constrained by a graph. In this case, the graph is given by the underlying network topology.

The main contribution of this work is to show that for most graphs, the constraints imposed by the graph are no constraint at all. That is, the number of tests required to identify the failed links in graph-constrained group testing is near-optimal even for the corresponding group testing problem *with no graph constraints*. Our approach is based on a simple randomized construction of tests. To analyze our construction, we prove new results about the size of giant components in randomly sparsified graphs.

Finally, we provide empirical results which suggest that our connected-subgraph tests perform better not just in theory but also in practice, and in particular perform better on a real-world network topology.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis

Keywords and phrases Group testing, network tomography, random graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.46

Category RANDOM

Related Version A full version of the paper is available at `https://arxiv.org/abs/1809.03589`.

Acknowledgements We thank Clément Canonne, Nick McKeown and the anonymous reviewers for helpful comments.

1 Introduction

Suppose you run a network with n switches and m links between the switches. Occasionally links will fail, and it is your goal to find and fix them. One common approach is to have each switch send a test packet on its neighboring links and report the results to a central monitoring system. However, in large networks these monitoring systems – and the volume of data that they produce – can become hard to manage. In light of this, the problem (sometimes called *network tomography* [5]) is: how little information does this central system need to find failing links quickly?

In the version of this problem that we focus on, suppose that some set of at most d links fail. Instead of observing these failures directly, we may send test packets along any connected walks in the network, and we observe whether or not each packet reaches its destination.[1] The goal is to identify any set of up to d failed links while sending as few packets as possible.

[1] We explain a bit more about how this, or tests equivalent to this, might be implemented in Section 2.3.1.


© Bruce Spang and Mary Wootters;
licensed under Creative Commons License CC-BY

We focus on the *non-adaptive* setting, where the packet walks must be fixed ahead of time. Non-adaptive tests are faster since they allow packets to be sent in parallel, and are easier to implement since these walks can be hard-coded once into the switches.

As observed by [6, 18], this problem is a variant of a well-studied problem called *combinatorial group testing*. Combinatorial group testing, originally motivated by the problem of cheaply testing for disease [10], has been studied since the 1940's and has applications from computational biology to wireless networks. We refer the reader to [11] for a survey. In the combinatorial group testing problem there are m items, at most d of which are "defective." A single test reveals whether or not there are any defectives in a subset T of items. The goal is to identify the defective items, by observing the output of a few tests.

The connection to network tomography is as follows: each link is defective if it fails, and each test T corresponds to a set of links. In the network tomography setting, there is one additional requirement: a test $T \subseteq [m]$ must correspond to a walk that a packet could take through the network. Because of this connection, [6] called this problem "graph-constrained group testing."

Our Question. A natural question is whether graph-constrained group testing is more difficult than classical group testing. That is, whether the additional constraints of graph-constrained group testing lead to significantly more tests. For the unconstrained group testing problem, the state-of-the-art construction uses $O(d^2 \log(m/d))$ tests [25], which nearly matches the lower bound of $\Omega(d^2 \log_d m)$ tests [13, 27]. Thus, our question is as follows:

▶ **Question 1.** *For what graphs G can we solve the graph-constrained group testing problem using $O(d^2 \log(m/d))$ tests?*

Previous work [18] has shown that certain graphs, such as a line, require $\Omega(m)$ tests, far more than the $O(d^2 \log(m/d))$ we would need for the unconstrained problem. However, it is also known that for sufficiently "well-connected" graphs (for example, those with large minimum cuts, many disjoint spanning trees, or constant mixing time), a sublinear number of tests suffice [6, 18]. These works have proposed using large subtrees [18] or random walks [6] as the tests. However, both of these approaches stop short (by polylogarithmic factors or more) of obtaining an $O(d^2 \log(m/d))$ bound for most graphs.

Our contributions. We improve upon the results of [6,18] to show that $O(d^2 \log(m/d))$ tests are sufficient for a wide collection of graphs, including many of the graphs already considered in prior work. Our construction – which is randomized – is quite simple: we sparsify the graph by choosing edges at random, and use the resulting large connected subgraphs. This is similar in flavor to earlier work – for example, [6] considered random walks – but our tests lead to stronger theorems, and also appear to perform better in practice. Moreover, our approach is quite general and works for other variants of the group-testing problem. To illustrate this, we show that it also gives near-optimal results in a model of random link failures. Concretely, our contributions are as follows:

- **$O(d^2 \log(m/d))$ tests suffice for (β, α)-edge expanders.** Our main result, Theorem 8, applies to graphs which are (β, α)-edge expanders, meaning that every set $S \subseteq V$ of size at most βn has at least $\alpha|S|$ edges coming out of it. We show that $O(d^2 \log(m/d))$ tests suffice when β is constant and $\alpha \gtrsim d$. Moreover, if β is sub-constant, then the number of tests required degrades gracefully with β.
  (β, α)-edge expansion is a general notion, and our results imply improved graph-constrained group-testing schemes for several natural classes of graphs like Erdös-Rényi graphs and constant-degree expanders. Moreover, our results are even optimal when applied to certain "counter-example" graphs like the barbell graph which foil earlier work.

Table 1 Summary of general results using connected-subgraph tests to identify any d defective edges for "well-connected" graphs with n vertices and m edges, for various notions of "well-connected." See discussion in Section 3 for slightly more general statements of results in previous work.

Source	Graph	Max. defective edges	Number of Tests
[18]	G has min-cut K	$d \leq \lceil \frac{K-1}{2} \rceil - 1$	$O(d^3 \log(m))$
[6]	G is D-regular, with mixing time τ	$d \leq d_0$ for some $d_0 = \Omega(D/\tau^2)$	$O(\tau^2 d^2 \log(m/d))$
Proposition 2	G has min-cut K	$d \leq \frac{K}{5 \log n}$	$O(d^2 \log(m/d))$
Theorem 8	G is a (β, α)-edge expander	$d \leq 1.99\alpha$	$O\left(\frac{d^2 \log(m/d)}{\beta}\right)$

Our general theorem (Theorem 8) is compared to existing general theorems in Table 1. The results for a few specific families of graphs are shown in Table 2. These results are presented in more detail in Section 4.

- **New results about large connected components of random graphs.** While our construction is quite simple, the analysis requires some delicacy. In order to show that our tests work, we prove new results about giant components in randomly sparsified graphs.

  More precisely, our main technical theorem (Theorem 10) establishes the following. Suppose that $G = (V, E)$ is a (β, α)-edge expander, and let $G(p) = G(V, E')$ be the graph where $E' \subseteq E$ is a random subset where each edge is kept independently with probability p. We show that if $p \geq (1 + \varepsilon)/\alpha$, then for any edge $e \in E$, with probability $\Omega(p\varepsilon)$ then not only does e survive, but also e's connected component in $G(p)$ has size at least βn. This is of a similar flavor to previous work on giant components of randomly sparsified graphs, but with two important differences: first, our result works even if β is small (so the components are "big" but not "giant") and second, we require that any edge e be contained in a large component with decent probability after sparsification. Theorem 10 is stated and proved in Section 5.

- **A general algorithm for graph-constrained group testing problems.** Our approach is to simulate a uniformly random (unconstrained) approach in the constrained setting. Because a uniformly random approach is near-optimal for many variants on the group testing problem, our results extend to variants of the graph-constrained group testing setting. We illustrate this in the full version of this paper, where we show that our algorithm also achieves near-optimal results in the stochastic model (where the link failures are random) [28].

- **Empirical results.** Finally, we present empirical results which suggest that our approach significantly out-performs the random-walk method of [6] and in many cases, nearly matches the performance of unconstrained random tests. On complete graphs and hypercubes it uses less than half the tests of the random-walk method. On a family of graphs often used in datacenter networks (the "fat-tree" topology), our approach is able to find defectives using a nontrivial number of tests while the random-walk method is not.

Table 2 Summary of work on the number of connected-subgraph tests required to identify any $d \leq d_0$ failures, for specific families of graphs. All graphs have n vertices and m edges. Results which meet the near-optimal $O(d^2 \log(m/d))$ tests or which have only the asymptotically optimal restriction $d \leq d_0$ for some $d_0 = \Omega(\text{degree})$ are highlighted in grey.

Graph	**Source**	**Number of tests**	**Limit d_0 so that recovery of $d \leq d_0$ failures is possible**
Complete Graphs	[6]	$O(d^2 \log(m/d))$	$d_0 = \Omega(n)$
	This work	$O(d^2 \log(m/d))$	$d_0 = \Omega(n)$
D-Regular Expanders ($O(1)$ spectral gap)	[18]	$O(d^3 \log m)$	$d_0 = \Omega(D)$
	[6]	$O(d^2 \log^3(m))$	$d_0 = \Omega(D/\log^2(n))$
	This work	$O(d^2 \log(m/d))$	$d_0 = \Omega(D)$
Erdös-Rényi Graphs $G(n, D/n)$	[6]	$O(d^2 \log^3(m))$	$d_0 = \Omega(D/\log^2(n))$
	This work	$O(d^2 \log(m/d))$	$d_0 = \Omega(D)$
Barbells	[18]	$O(d^3 \log m)$	$d_0 = 1$
	[6]	m (see discussion)	$d_0 = n$
	This work	$O(d^2 \log(m/d))$	$d_0 = \Omega(n)$
Fat-Trees	[18]	$O(d^3 \log m)$	$d_0 = 1$
	This work	$O(d^2 \log(m/d))$	$d_0 = \Omega(D/\log D)$

1.1 Overview of approach

Our approach is quite simple: we just choose random edges and take the large connected components. Intuitively, the reason that this works is because:

1. If we just chose random edges, we would be back in the traditional group testing setting, where random tests are nearly optimal by Proposition 6.
2. We will show that if G is a (β, α)-edge expander, then the graph $G(p)$ formed by choosing random edges has mostly large components, of size at least βn. Intuitively this means that throwing away the few disconnected parts should not matter much.

There are several challenges in making the above intuition rigorous. First, once we throw away the edges that are not in a large connected component, the edges that remain, conditional on remaining, are no longer independent. Thus, the intuition from point 1 above does not quite hold. However, this ends up being reasonably straightforward to deal with.

The second and more interesting challenge is that, while there is a great deal of work on when random sparsifications of graphs have giant components, we need a different result that to the best of our knowledge does not appear in the literature. The first difference between our setting and existing work is that we work with (β, α)-edge expanders; this means that we need to show that there are "decently big" connected components rather than "giant" components if β is small. The second difference is that we must show that each edge e is still contained in a test with high probability. That is, when we pass from G to $G(p) = (V, E')$,

the probability that $e \in E'$ is p; for the analysis in the random case to still work, we need the probability that e is in a large connected component of E' to also be proportional to p.

To address this second challenge, we reduce the question to one about random walks. Taking inspiration from [22] who study random sparsifications of the complete graph, we introduce a process to generate the connected component of a particular edge v, and argue that this process generates a large connected component if and only if an appropriately chosen random walk diverges with decent probability; the we prove it does converge.

The details of the approach are given in Section 5. However, as a warm-up to show why the intuition presented above is believable, we first prove an easy statement of the same flavor where neither of these challenges arise. Proposition 2 below shows that if we replace (β, α)-expansion with the property of having a large min-cut, $O(d^2 \log(m/d))$ tests suffice.

▶ **Proposition 2** (Informal). *There is a constant $C > 0$ so that the following holds. Suppose that $G = (V, E)$ is a graph with $|V| = n$, $|E| = m$. Let $d \geq 1$ be an integer. Suppose that the minimum cut K of G satisfies $K \geq 5(d+1)\log(n)$. Let $\mathcal{T}$ be a set of tests $T \subseteq E$ generated according to the following process:*

- *Initialize $\mathcal{T} = \emptyset$.*
- *For $t = 1, 2, \ldots, Cd^2 \log(m/d)$:*
  - *Let $T \subset E$ be a random set where each edge is included with probability $1/(d+1)$.*
  - *Add T to $\mathcal{T}$.*

Then $\mathcal{T}$ can identify d failed edges, and with high probability each test is connected.

Proof sketch. As we will see in Proposition 6, $\mathcal{T}$ is able to identify d failed edges, and so it suffices to show that a random set T is connected with high probability. Fortunately, this is true:

▶ **Theorem 3** ([21]). *Let K be the minimum cut of G. If $p > \min\left(\frac{5\log n}{K}, 1\right)$, $G(p)$ is connected with probability at least $1 - \frac{1}{n}$.*

By Theorem 3 and a union bound, the probability any test is disconnected is at most $|\mathcal{T}|/n = O\left(\frac{d^2 \log(m/d)}{n}\right)$. ◀

Proposition 2 is already enough to establish order-optimal results for some of the examples shown in Table 2 (for example the complete graph and the fat tree), but for the others we will need Theorem 8 about (α, β)-expanders.

Organization. In Section 2, we formally set up the problem. In Section 3, we survey related work, and we state our theoretical results in Section 4. The proofs of these results follow in Section 5. Finally, we present our empirical results in Section 6.

2 Setup and Preliminaries

We begin with some basic notation and definitions.

2.1 Graph-theoretic preliminaries

Throughout, we will be working with undirected, unweighted graphs $G = (V, E)$ with $|V| = n$, $|E| = m$. For a set of vertices $A \subseteq V$, the *boundary* of A is $\partial A = \{\{u, v\} \in E : u \in A, v \notin A\}$. For a set of edges $B \subseteq E$, we use the notation $N(B)$ to denote the set of vertices v that are endpoints of an edge in B: $N(B) = \{v \in V : \exists u, \{u, v\} \in B\}$.

The minimum cut K of a graph G is defined by $K = \min_{A \subseteq V} |\partial A|$. Our main theorem is about *edge expanders*. We give a slightly more general definition than the usual notion (which would have $\beta = 1/2$ below), so that we can state a more general theorem.

▶ **Definition 4.** *A graph $G = (V, E)$ is a (β, α)-edge expander if for all sets $A \subseteq V$ with $|A| \leq \beta|V|$, $|\partial A| \geq \alpha|A|$.*

We will consider random sparsifications of graphs. For a graph $G = (V, E)$ and $p \in (0, 1)$, $G(p) = (V, E')$ denotes the random graph where $E' \subseteq E$ is generated by including each edge of E in E' independently with probability p. We use $G(n, p) = K_n(p)$ to denote the Erdös-Rényi graph where each edge is included independently with probability p. (Here, K_n denotes the complete graph on n vertices).

2.2 Group testing preliminaries

The combinatorial group testing problem is set up as follows (using slightly non-standard notation in order to be consistent with the graph-constrained set-up below). Let E be a set of size m, and suppose that $B \subseteq E$ is a set of at most d special or "defective" items in E. A *test* $T \subseteq E$ is a collection of items, and we say that the *outcome* of the test T is TRUE if $T \cap B \neq \emptyset$ and FALSE otherwise. We say that a collection of tests $\mathcal{T} \subseteq 2^E$ (here, 2^E denotes the power set of E, consisting of all subsets of E) *can identify up to d defective items in E* if for any $B \subseteq E$ with $|B| \leq d$, B is uniquely determined from the outcomes of the tests $T \in \mathcal{T}$. The goal is to design a collection of tests $\mathcal{T} \subseteq 2^E$ which can identify up to d defective items, so that $|\mathcal{T}|$ is as small as possible. A useful notion in the group testing literature is *disjunctness*, which is a sufficient condition for recovery.

▶ **Definition 5.** *Let E be a set and $\mathcal{T} \subseteq 2^E$. We say that $\mathcal{T}$ is* d-disjunct *if for all $e \in E$, for all $B \subseteq E$ where $|B| \leq d$ and $e \notin B$, there exists a test $T \in \mathcal{T}$ so that $e \in T$ and $B \cap T = \emptyset$.*

If $\mathcal{T}$ is d-disjunct, then $\mathcal{T}$ can identify up to d defective items in E. More precisely, it is not hard to see that the following algorithm will do the job: for each item $e \in E$, declare $e \in B$ if and only if all the tests $T \in \mathcal{T}$ with $e \in T$ had outcome TRUE.

Choosing tests completely at random is a good way to obtain d-disjunct sets.

▶ **Proposition 6** (See e.g. [11] Theorem 8.1.3)**.** *Let $d \geq 1$. Let E be a set. Consider a random test $T \subseteq E$ such that each $e \in E$ is included in T independently with probability $p = \frac{1}{d+1}$. Let $\mathcal{T} = \{T_1, \ldots, T_\tau\}$, where each $T_i \in \mathcal{T}$ is chosen independently from the above distribution. Then there is a value $\tau = O(d^2 \log(m/d))$ so that $\mathcal{T}$ is d-disjunct with probability at least $1 - 1/m$.*

This is nearly optimal, up to a factor of $O(1/\log d)$:

▶ **Theorem 7** ([13])**.** *Let E be a set of size m and $\mathcal{T} \subseteq 2^E$. If $\mathcal{T}$ is d-disjunct, then $|\mathcal{T}| = \Omega(d^2 \log_d m)$*

2.3 Graph-constrained group testing

Given a graph $G = (V, E)$, the graph-constrained group testing problem on G is the same as the standard group testing problem on a set of items E, with the additional constraint that each test $T \subseteq E$ be a connected subgraph of G. That is, in the network tomography application, a packet must be able to traverse a test T. We say that a *connected-subgraph test* is a set of edges $T \subseteq E$ so that the graph $(N(T), T)$ is connected.

We note that the definition of disjunctness directly applies to the graph-constrained setting, and our goal in this work will be to design d-disjunct collections $\mathcal{T}$ of connected-subgraph tests, such that $\mathcal{T}$ is as small as possible.

2.3.1 Why connected-subgraph tests?

Our work, like existing work on graph-constrained group testing [6, 18, 20], uses connected-subgraph tests. Connected-subgraph tests can be implemented in an actual network in a number of ways:

- The test can be converted into a path on the graph by solving an instance of the Chinese Postman Problem [14]. A packet can then be sent along this path, for instance by using source routing or by adding a rule at each switch on the path matching the packet's source IP, destination IP, and time-to-live.
- We can compute a spanning tree of subgraph. Each switch in the subgraph checks the health of its links, and forwards a bit to its parent in the tree: 1 if its adjacent links are healthy, and 0 otherwise. If the link from a switch to its parent in the tree is down, then it cannot send anything. If the root node receives all 1's, it knows that all the links in the subgraph are healthy; otherwise at least one of the links is broken.

One could also imagine restricting the tests to be, for example, simple paths or trees. Some restrictions on tests do have some advantages in implementation (in particular, tests which are *shortest* paths may be easier to implement than general connected-subgraph tests: for example many networks support equal-cost multipath routing (ECMP) which splits traffic across all the shortest paths between a pair of hosts). However, connected subgraph tests are strictly more powerful than either simple paths or trees for the constrained group-testing problem. We provide some examples which demonstrate this in the full version of this paper [28].

3 Related Work

Boolean network tomography. Most of the work on *boolean network tomography* (that is, the problem of identifying failures in a graph using end-to-end traffic) has a much harsher set-up than the one we consider here, in that both the graph and the tests are taken to be worst-case, or at least very constrained. For example, all the tests may be required to be simple paths starting from a particular vertex. The reason for the harsh set-up is that historically, networks have been quite inflexible, which severely limits both the graph topologies and the sorts of tests that are used. Since identifying failures uniquely is often impossible in these settings, this work has focused on doing as well as possible given the circumstances, for example by finding *any* set of failures that will explain the test outcomes [3, 8] or by finding the most likely set of failures given some underlying distribution [12, 24]. When the input graph and set of allowed tests is worst-case, these problems are hard, and the usual approach is to reduce to some NP-hard problem and use a heuristic or approximation algorithm.

Graph-constrained group testing. More recently, the field of networking has shifted towards flexible datacenter networks, where the tomography problem is still interesting [26, 30]. Modern datacenter networks, however, fundamentally change the constraints of the tomography problem. Datacenters are good expanders [9, 29], so the worst-case assumptions about the graphs can be relaxed. Modern networks are programmable [4]: instead of the network defining what can and cannot be done, operators program networks to do what they want. Thus, the set of allowed tests $\mathcal{T}$ need not be worst-case. This leads to graph-constrained group testing, where we can design the tests, and make assumptions about the connectivity of the underlying network.

We are not the first to investigate group testing with graph constraints (although to the best of our knowledge our work is the first to consider graph-constrained group testing with random failures). Du and Hwang discuss two different group-testing problems on graphs in Chapter 12 of [11], although neither are exactly the same as the setup we consider here. The connection between boolean network tomography and group testing was first observed by [18]. They give results for specific families of graphs including line graphs, grids, and binary trees. Their most general result is that if a graph has d edge-disjoint spanning trees $T_1, \ldots, T_d$ with δ being the maximum diameter of the trees, then $O(d^3 \log m + d \min(\delta + \log^2 n, \delta \log n))$ tests are sufficient to identify at most d failures. (As a corollary, this implies that any graph with minimum cut K can identify $d \leq \lceil \frac{K-1}{2} \rceil - 1$ failed edges, which is what is stated in Table 1). Finally, [20] considers the adaptive version of graph-constrained group testing. They present an adaptive algorithm which, on any graph, uses a number of tests that is within a constant factor of the optimal number.

The work closest to our is that of Cheraghchi et al. [6], who give a randomized construction of connected-subgraph tests via random walks. They show that $O(d^2 \log(m/d))$ tests suffice for *very* well-connected graphs (those with constant mixing time), and $O(d^2 \log^3(m))$ tests suffice for certain expanders and Erdös-Réyni graphs. Their most general result is that for graphs with mixing time τ and where there exists some $c > 0$ such that the degree D_v of each vertex $v \in V$ lies in between $6c^2 d\tau^2 \leq D_v \leq 6c^3 d\tau^2$, at most $O(c^4 \tau^2 d^2 \log(m/d))$ tests are sufficient. We summarize the general results for related work in non-adaptive graph-constrained group testing in Table 1.

Giant components in random graphs. Finally, we mention some related work on the size of giant components of randomly sparsified graphs, since our main technical theorem (Theorem 10) is related to this. This question is well-studied, but we need a slightly different result. One difference is that we work with (β, α)-edge expansion, and in particular our "giant" components need not be so giant if β is small. A second difference is that we require that every edge be contained in a large connected component with constant probability; to the best of our knowledge, existing work does not explicitly give such a guarantee.

The study of giant components in $G(n, p)$ (aka, a randomly sparsified complete graph) was initiated by Erdös and Rényi in [15]. This was extended to sparsifications of the hypercube [1] and sufficiently good expander graphs [16] and [7]. Our approach to Theorem 10 is based on that of Krivelevich and Sudakov [22] who give a simpler argument for existing results on giant components.

Most of these results show that as long as $p \leq \frac{1+\varepsilon}{D}$, where D is the degree of the graph G, then $G(p)$ contains a giant component. We show a similar result: if $p \leq \frac{1+\varepsilon}{\alpha}$, where G is a (β, α)-edge expander, then there exists a connected component of size at least βn. (And moreover, any edge is contained in such a component with decent probability).

4 Results

4.1 Main result

Our group testing scheme is quite simple: the idea is just to choose random edges of the graph, and keep any large-enough connected components. Recall the notation that for a graph $G = (E, V)$, $G(p) = (V, E')$ is the graph where each edge in E is included in E' independently with probability p. Then the randomized algorithm for constructing the tests is given in Algorithm 1.

Algorithm 1 Make-Tests.

input : Graph $G = (V, E)$; number of failed edges d; parameters $\frac{2}{d} \leq \delta \leq 1/3$, $\beta \in (0, \frac{1}{2}]$, and $\tau \in \mathbb{N}$
output : A collection of tests $\mathcal{T} \subseteq 2^E$

```
T ← ∅;
p ← 1/(δd);
for t = 1, ..., τ do
    Draw G' ~ G(p) independently from all the other rounds;
    Find the connected components A_1, A_2, ..., A_r of G(p);
    for each A_i so that |A_i| ≥ βn do
        Add the test A_i to T;
    end
end
return T
```

Our main theorem implies that any (β, α)-edge expander with large enough α admits a group testing scheme with $O(d^2 \log(m/d)/\beta)$ tests.

▶ **Theorem 8.** *There are constants $c, C > 0$ so that the following holds. Suppose that $G = (V, E)$ is a graph with $|V| = n, |E| = m$. Let $d \geq 1$ be an integer, and $\frac{2}{d} \leq \delta \leq \frac{1}{3}$. Suppose that G is a (β, α)-edge-expander with $\beta \in (0, 1/2]$ and such that $\alpha \geq 1$ satisfies*

$$\alpha \geq d\left(\frac{1}{2} + \delta\right). \tag{1}$$

Let $\mathcal{T}$ be the set of tests returned by Algorithm 1 run with parameters δ, β, and

$$\tau = Cd^2 \log(m/d) e^{(1+\delta)/\delta}.$$

Then with probability at least $1 - m^{-cd}$, $\mathcal{T}$ is d-disjunct. Further,

$$|\mathcal{T}| \leq \frac{Cd^2 \log(m/d) e^{(1+\delta)/\delta}}{\beta}.$$

If β, δ are constant, then the number of tests required is $O(d^2 \log(m/d))$, which is nearly optimal even for the unconstrained group testing problem (that is, it nearly matches Theorem 7). Moreover, the requirement on the expansion factor α is nearly tight. That is, in (1), we may take $\alpha = d\left(\frac{1}{2} + \delta\right) d$ for any constant $\delta > 0$, while still maintaining the near-optimal $O(d^2 \log(m/d))$ test complexity. On the other hand, such a statement could not hold for α much smaller than $d/2$. More precisely, we show below in Proposition 9 that the degree D of the graph G must be at least $d/2$ to obtain any nontrivial bound on $|\mathcal{T}|$. Since for any $\gamma > 0$, there exist D-regular (β, α)-edge expanders with $\alpha = (1 - \gamma)D$ for small $\beta \sim \gamma$, this implies that the cut-off for α of $d/2$ in Theorem 8 cannot be improved.

▶ **Proposition 9.** *Let $G = (V, E)$ be a D-regular graph with $|E| = m$. If $\mathcal{T} \subseteq 2^E$ is a collection of d-disjunct connected-subgraph tests, for $d \geq 2D - 2$, then $|\mathcal{T}| \geq m$.*

Proof. Suppose that $\mathcal{T} \subseteq 2^E$ with $|\mathcal{T}| < m$. Then there is some edge $e = \{u, v\}$ so $\{e\} \notin \mathcal{T}$. Let $B = \partial(\{u, v\})$ be the set of edges adjacent to e, so $|B| = 2D - 2 \leq d$. Then the only connected-subgraph test $T \subset E$ so that $e \in T$ but $T \cap B = \emptyset$ is $\{e\}$, which by assumption is not in $\mathcal{T}$. Thus, $\mathcal{T}$ is not d-disjunct. (See Figure 1). ◀

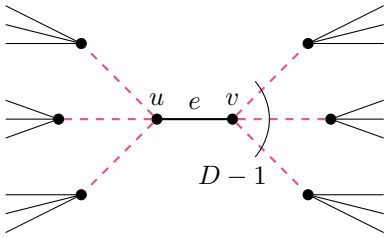


Figure 1 Proof of Proposition 9. Suppose that the number of edges that may fail is $d \geq 2D-2$ where D is the degree of the graph. If the set of tests $\mathcal{T}$ does not contain the singleton $\{e\}$ for an edge $e = \{u, v\}$, then the set B of dashed edges – all of the neighbors of e – provides an counter-example to d-disjunctness.

We instantiate Theorem 8 for several diffrent families of graphs in Appendix A; the results are summarized in Table 2. We note that our results also extend to a model with random link failures; we defer this discussion to the full version of this paper [28].

5 Proofs

In this section, we prove Theorem 8. Our proof is based on the following theorem, which implies that any edge e is reasonably likely to be contained in a large connected component of $G(p)$.

▶ **Theorem 10.** *Let $\beta \in (0, 1/2)$ and $\alpha \geq 1$, and let $G = (V, E)$ be a graph with $|V| = n, |E| = m$ so that, for any set $A \subset V$ of size $2 \leq |A| \leq \beta n$, we have $|\partial A| \geq \alpha |A|$. For an edge $e \in E$, let C_e denote the connected component of $G(p)$ containing e, or $\emptyset$ if there is no such connected component (that is, if $e \notin G(p)$ was deleted), and let $|C_e|$ denote the number of vertices in C_e.*

Choose any $\varepsilon \in (0, 1/3)$, and suppose that $p \geq \frac{1+\varepsilon}{\alpha}$. Then for all edges $e \in E$,

$$\mathbb{P}\left(|C_e| \geq \beta n\right) \geq \frac{p\varepsilon}{8}. \tag{2}$$

Before we prove Theorem 10, we discuss how it can be used to prove Theorem 8.

Let $G = (V, E)$, and let $G(p) = (V, E')$ be the random sparsification. Fix $B \subseteq E$ with $|B| = d$ and $e \in E$. The following lemma shows that with high probability, at least one of the tests in $\mathcal{T}$ will separate e from B. Then one can union bound over all choices for e and B to conclude that $\mathcal{T}$ is d-disjunct. We defer the details to the full version of this paper [28].

▶ **Lemma 11.** *Consider one draw of $G(p)$ in Make-Tests, and let $T_1, T_2, \ldots$ be the connected components of $G(p)$ which have size at least βn. Fix $B \subseteq E$ with $|B| = d$ and $e \in E$. Then*

$$\mathbb{P}\left(\exists i \text{ s.t. } e \in T_i \text{ and } B \cap T_i = \emptyset\right) \geq \frac{\delta p}{8} \cdot (1-p)^d.$$

Proof. We have

$$\begin{aligned}
\mathbb{P}\left(\exists i \text{ s.t. } e \in T_i \text{ and } B \cap T_i = \emptyset\right) &= \mathbb{P}\left(|C_e| \geq \beta n \text{ and } B \cap C_e = \emptyset\right) \\
&\leq \mathbb{P}\left(|C_e| \geq \beta n \text{ and } B \cap E' = \emptyset\right) \\
&= \mathbb{P}\left(|C_e| \geq \beta n \mid B \cap E' = \emptyset\right) \cdot \mathbb{P}\left(B \cap E' = \emptyset\right).
\end{aligned}$$

We have $\mathbb{P}\left(B \cap E' = \emptyset\right) = (1-p)^d$, since this is just the probability that all the edges in B survive in $G(p)$.

For our fixed e, B, let $\bar{G} = (V, E \setminus B)$ be the graph with all the edges in B removed. Consider the distribution of $G(p)$ conditioned on the event that $B \cap E' = \emptyset$. This is the same as the distribution of $\bar{G}(p)$. To see this, notice that for $A \subseteq E \setminus B$, the random sets of $A \cap E'$ and $B \cap E'$ are independent. Let $\overline{C_e}$ be the connected component containing e in $\bar{G}(p)$. Then

$$\mathbb{P}\left(|C_e| \geq \beta n \mid B \cap E' = \emptyset\right) = \mathbb{P}\left(|\overline{C_e}| \geq \beta n\right).$$

Notice that since G is a (β, α)-edge expander with $\alpha \geq d\left(\frac{1}{2} + \delta\right)$, then for any set $A \subset V$ of size $2 \leq |A| \leq \beta n$, we have $|\partial A| \geq \alpha|A| - d \geq \left(\alpha - \frac{d}{2}\right)|A|$. Thus, we may apply Theorem 10 to $\bar{G}$. Choose $p = \frac{1+\delta}{d\delta}$, and apply Theorem 10. Our assumption (1) that $\alpha \geq d\left(\frac{1}{2} + \delta\right)$ and the choice of p implies that $p \geq \frac{1+\delta}{\alpha - d/2}$, we conclude that $\mathbb{P}\left(|\overline{C_e}| \geq \beta n\right) \geq \frac{\delta p}{8}$. Putting things together proves the lemma. ◀

Proof of Theorem 10. As in the theorem statement, suppose $p \geq (1+\varepsilon)/\alpha$ for some $\varepsilon \in (0, 1/3)$, and let $G = (V, E)$ be a (β, α)-edge expander. Write $G(p) = (V, E')$, so that $E' \subseteq E$. Choose $p \geq (1+\varepsilon)/\alpha$.

Consider the probability (u, v) is in a large component,

$$\mathbb{P}\left(C_{(u,v)} \geq \beta n\right) = \mathbb{P}\left((u,v) \in E'\right) \cdot \mathbb{P}\left(|C_{(u,v)}| \geq \beta n \mid (u,v) \in E'\right).$$

Condition on the event that $(u, v) \in E'$, and imagine building $C_{(u,v)}$ by starting with $\{(u, v)\}$ and building the set outwards, one at a time. More precisely, consider the following randomized process:

- $S_0 \leftarrow \{(u, v)\}$, $B_0 \leftarrow \emptyset$
- For $t = 0, 1, 2, \ldots$:
  1. Let $N(S_t)$ be the set of vertices in V adjacent to S_t.
  2. Let $U_t = \partial(N(S_t)) \setminus B_t$ be the set of unvisited edges that lie on the boundary of $N(S_t)$.
  3. If $|U_t| = 0$, **break**.
  4. $S_{t+1} \leftarrow S_t$, $B_{t+1} \leftarrow B_t$.
  5. Choose an edge $e \in U_t$ arbitrarily.
  6. With probability p, declare that e has survived and add it to S_{t+1}.
  7. Otherwise (with probability $1 - p$) add e to B_{t+1}.

This process is illustrated in Figure 2.

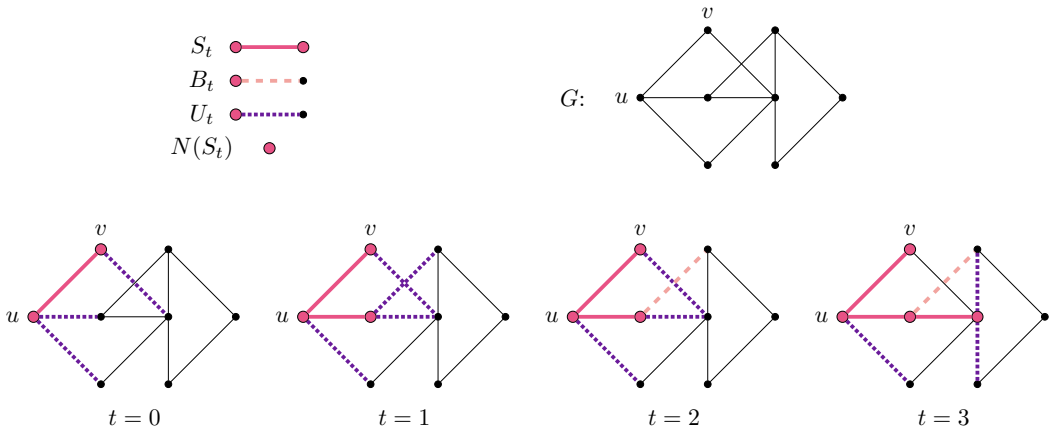


Figure 2 First few steps of the process to build $C_{(u,v)}$ in the proof of Theorem 10. The edge from U_t which we chose ended up being in E' in steps $t = 1$ and $t = 3$, but not $t = 2$.

It is not hard to see that this process terminates at the first time $t_{\max}$ so that $|U_{t_{\max}}| = 0$, and when it does, $N(S_{t_{\max}})$ is distributed identically to the set of vertices in $C_{(u,v)}$. Thus, to bound $|C_{(u,v)}|$ with high probability, we can bound the set $|N(S_t)|$ with high probability. Notice that S_t is a tree; thus, $|S_t| = |N(S_t)|-1$, and so it suffices to show that $|S_{t_{\max}}| > \beta n-1$ with high probability. To that end, we will show that, as long as $|S_t| \leq \beta n-1$, the probability that $|U_t| = 0$ is very small.

At each step t, we either add an edge to B_t or to S_t, so $|S_t| + |B_t| = t$. Let X_t be the random variable which is 1 if we added an edge to S_t in step t; thus $X_t \sim \text{Ber}(p)$ and $|S_t| = 1 + \sum_{i=1}^{t} X_i$.

Suppose that S_t is nonempty and $|S_t| \leq \beta n - 1$, so $2 \leq |N(S_t)| \leq \beta n$. By our expansion assumption, we have $|\partial N(S_t)| \geq \alpha |S_t|$, and so

$$\begin{aligned}|U_t| &= |\partial(N(S_t)) \setminus B_t| \geq \alpha|S_t| - |B_t| = \alpha|S_t| - (t - |S_t|) = (1+\alpha)|S_t| - t \\ &= (1+\alpha)\left(1 + \sum_{i=1}^{t} X_i\right) - t \geq \sum_{i=1}^{t}((1+\alpha)X_i - 1) + \alpha.\end{aligned}$$

Thus,

$$\begin{aligned}\mathbb{P}\left(\exists t > 0 \quad \text{s.t. } |S_t| \leq \beta n - 1 \text{ and } |U_t| = 0\right) &= \mathbb{P}\left(\exists t > 0 \quad \text{s.t. } |S_t| \leq \beta n - 1 \text{ and } |U_t| \leq 0\right) \\ &\leq \mathbb{P}\left(\exists t > 0 \quad \text{s.t.} \quad \sum_{i=1}^{t}((1+\alpha)X_i - 1) \leq -\alpha\right).\end{aligned}$$

Let $Y_i = (\alpha+1)X_i - 1$, so that $Y_i = \alpha$ with probability p, and $Y_i = -1$ with probability $1-p$. Let $Z_t = \sum_{i=1}^{t} Y_i$ be a random walk. The above shows that

$$\mathbb{P}\left(\exists t > 0, |S_t| \leq \beta n - 1 \text{ and } |U_t| = 0\right) \leq \mathbb{P}\left(\exists t > 0, Z_t \leq -\alpha\right). \tag{3}$$

Let $\tau_k(Y_i)$ denote the smallest t so that $\left|\sum_{i=1}^{t} Y_i\right| \geq |k|$ and $\text{sign}(\sum_{i=1}^{t} Y_i) = \text{sign}(k)$. We claim that $\mathbb{P}\left(\exists t > 0, Z_t \leq -\alpha\right) \leq 1 - \inf_{M>0} \mathbb{P}\left(\tau_M(Y_i) \leq \tau_{-\alpha}(Y_i)\right)$. Indeed, suppose that there is some $t > 0$ so that $Z_t \leq -\alpha$. Then there is some sufficiently large $M > \alpha t$ so that Z_t could not have reached M in t steps, and so the random walk does not arrive at M before arriving at $-\alpha$. Thus

$$\begin{aligned}\mathbb{P}\left(\exists t > 0, Z_t \leq -\alpha\right) &\leq \mathbb{P}\left(\exists M, \tau_M(Y_i) > \tau_{-\alpha}(Y_i)\right) \\ &= \sup_M \mathbb{P}\left(\tau_M(Y_i) > \tau_{-\alpha}(Y_i)\right) = 1 - \inf_{M>0} \mathbb{P}\left(\tau_M(Y_i) \leq \tau_{-\alpha}(Y_i)\right).\end{aligned}$$

Thus our goal is to show that $\mathbb{P}\left(\tau_M(Y_i) \leq \tau_{-\alpha}(Y_i)\right)$ is bounded away from zero for all M; then there will be some constant probability that the edge-exploration process will make progress as long as $|S_t| \leq \beta n - 1$.

▷ Claim 12. Using the notation above, for all $M > 0$,

$$\mathbb{P}\left(\tau_M(Y_i) \leq \tau_{-\alpha}(Y_i)\right) \geq \frac{\varepsilon}{8}.$$

The proof of the claim can be found in the full version of this paper [28]. The main idea is as follows: suppose for simplicity that $\alpha \in \mathbb{Z}$. Then we replace the walk defined with the Y_i's by one whose steps W_i are ± 1, where the probability of 1 is chosen so that the the probability that W_i reaches α before -1 is at most p. Analyzing this walk is then an instance of the Asymmetric Gambler's Ruin problem (see, for example, [17]).

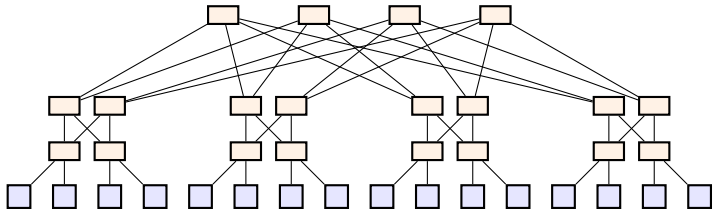


Figure 3 A small example of the fat tree topology with $n = 36, m = 48$. In our experiments, we consider larger versions ($n = 80, m = 256$ and $n = 45, m = 108$).

Using the claim and the discussion following (3), we have that

$$\mathbb{P}\left(\exists t > 0 \text{ s.t. } |S_t| \leq \beta n - 1, |U_t| = 0\right) \leq 1 - \varepsilon/8.$$

Finally,

$$\begin{aligned}\mathbb{P}\left(|C_{(u,v)}| \geq \beta n\right) &= \mathbb{P}\left(|C_{(u,v)}| \geq \beta n \mid (u,v) \in E'\right) \cdot \mathbb{P}\left((u,v) \in E'\right) \\ &= p \cdot \left(1 - \mathbb{P}\left(\exists t \text{ s.t. } |S_t| \leq \beta n - 1, |U_t| = 0\right)\right) \geq \frac{p\varepsilon}{8}.\end{aligned}$$

This completes the proof. ◀

6 Empirical Results

In this section, we numerically compare Algorithm 1 to existing work. We compare to the random walk based approach of [6] and to the randomized group testing *without* graph constraints of Proposition 6. We find that the random subgraph tests perform nearly as well as the unconstrained versions in most settings, and often perform significantly better than the random walk approach.

Below, we test the following randomized constructions of tests:

- Our approach, Algorithm 1 (called "Subgraph" in the figures). We use $p = 1/(d+1)$ and include only the largest connected component of $G(p)$.
- The random walk approach of [6] (called "Random walks" in the figures). We empirically estimate the mixing time τ by picking a node at random and finding the first time that the total variation distance to the equilibrium distribution is less than $1/(2cn)^2$, as per the definition in [6], where c is defined so that the graph $G = (V, E)$ has $D \leq \deg(v) \leq cD$ for each $v \in V$. We fix a constant $\ell > 0$ and run each random walk for $\lceil \frac{\ell n D}{c^3 d \tau} \rceil$ steps. We tried a few of values of ℓ and present the best results for each graph: for the complete graph we chose $\ell = 1$ and for all other graphs we chose $\ell = 4$.
- Unconstrained random approach (called "Random" in the figures). We include each edge in a test with probability $p = 1/(d+1)$, and ignore the graph constraints.

We consider four types of graphs. The first three – random regular graphs, complete graphs, and hypercubes – are idealized graphs that may or may not capture real networks. For our last graph, we choose the "Fat-Tree" graph [23], originally designed for use in supercomputers and which is now widely used in datacenter networks [2]. As the name suggests, this is a "fattened" tree, where the fatness (number of links) near the top of the tree is greater than the fatness near the leaves. (See Figure 3).

We perform two types of experiments:

1. In the first type of experiment, we compare the probability of obtaining a d-disjunct matrix from any of these three randomized approaches. Unfortunately, it is computationally intense to determine whether or not a given collection $\mathcal{T}$ of tests is d-disjunct, and so we are only able to do this for small d ($d = 1$ and $d = 2$).
2. In the second type of experiment, we are trying to understand the performance of our method for larger d. Since determining d-disjunctness is computationally infeasible for large d, instead we choose d random defectives and estimate the probability of success under each of the three methods. We note that, as shown in the full version of this paper [28], our algorithm is order-optimal for random defectives.

6.1 Number of tests required for d-disjunctness

First, we estimate the number of tests required for d-disjunctness for Algorithm 1 and compare it to [6] and randomized group testing without graph constraints. We find that for many graphs, our approach requires roughly the same number of tests as group testing without constraints.

Figure 4 (resp. Figure 5) shows the probability that a randomly generated test matrix is 1-disjunct (resp. 2-disjunct) for various graphs, algorithms, and numbers of tests. Each point is the empirical mean of 200 independent trials, and we plot error bars of width $1/\sqrt{200} \approx 0.07$. (Notice that by Hoeffding's inequality, the probability that the true average lies outside the error bars is at most $1/e^2$).

Algorithm 1 performs similarly to the nearly optimal randomized group testing procedure of Proposition 6. Notably, for the Fat-Tree graph, Algorithm 1 significantly outperforms the approach of [6].

6.2 Number of tests required for d random failures

As mentioned above, determining d-disjunctness is computationally infeasible for larger d, and so to assess larger d we consider performance on random failures, which we address in the full version [28].

For our experiments with random failures, we focus on the fat-tree topology. The main reasons for this are (a) that the "Fat-Tree with random failures" set-up is perhaps the most relevant for real-life applications, and (b) the other topologies yield graphs that look similar, but the differences between the three approaches are less pronounced.

We find that Algorithm 1 significantly out-performs the random walk approach of [6], but performs less well as d grows. In this graph once d becomes much larger than 5, even the random group testing construction without graph constraints requires at least m tests.

Figure 6 shows the probability that a set of tests correctly identifies d random failures, where the probability is taken over both the tests and the failures. Each point is the empirical mean of 200 independent trials, and we plot error bars of width $1/\sqrt{200} \approx 0.07$. As above, by a Hoeffding bound the probability that the true mean lies outside the error bars is at most e^{-2}.

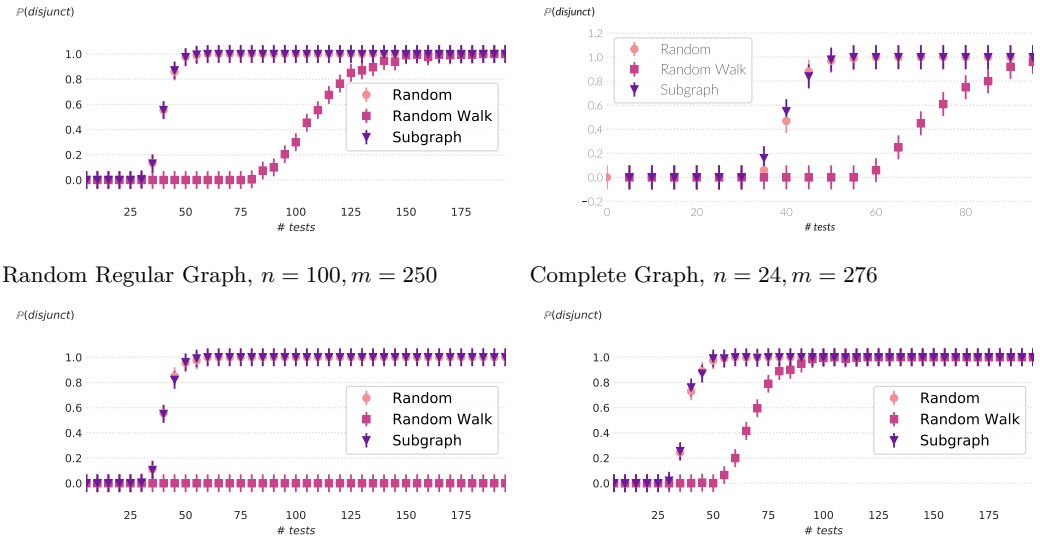
7 Conclusion

We have given a simple randomized construction which shows that for many graphs, graph-constrained group testing is possible with a near-optimal number of tests. Our results – which are proved by analyzing a particular random walk – improve over previous work, and also apply to a wider range of graphs. However, many open questions remain, and we conclude with a few of these here.

1. Both our approach and the approach of [6] give randomized constructions. Derandomizing these constructions remains a fascinating open question. Such a derandomization would be especially useful if it allowed a node to extremely efficiently determine which of its neighbors a test packet should be sent to next, using only minimal information stored in the packet.

2. While (β, α)-expansion is reasonably general, it is not completely general. For example, hypercubes are not very good (β, α)-expanders, but the result of [18] implies that (since they have many disjoint spanning trees) hypercubes are reasonably good for the graph-constrained group-testing problem: $O(d^3 \log m)$ tests suffice to identify d defectives for $d \lesssim \log(n)$. It seems possible that one could modify our analysis using the approach of [1] – which shows that random sparsifications of hypercubes have large connected components with high probability – to obtain a good result for hypercubes as well. Thus, it is an open question to see how well our approach works for hypercubes, but more generally if there is some quantity (more general than (β, α)-edge expansion) which precisely captures when our approach works and when it does not.

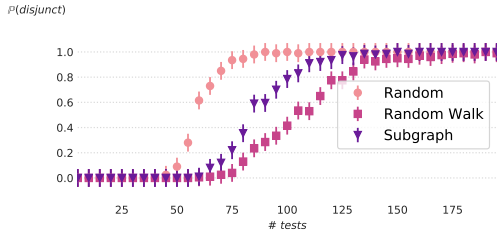
3. In the full version [28], we show that simple paths are not as powerful as connected-subgraph tests for graph-constrained group testing. However, in practice it is often the case that simple paths (and especially shortest paths) are easier to implement. It would be interesting to characterize the limitations of graph-constrained group testing when the tests are restricted to (shortest) simple paths.



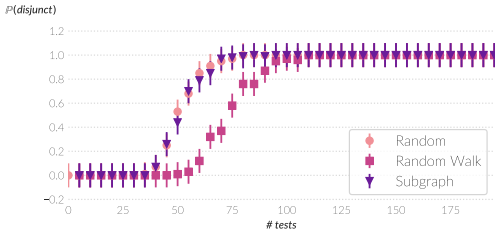
Random Regular Graph, $n = 100, m = 250$ | Complete Graph, $n = 24, m = 276$

Fat Tree, $n = 80, m = 256$ | Hypercube, $n = 64, m = 192$

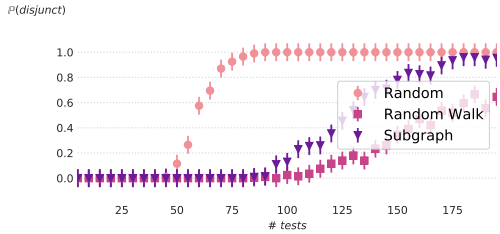
Figure 4 Probability that a randomly generated test matrix with a certain number of tests is 1-disjunct for various graphs. Each point is the mean of 200 trials, with error bars of $1/\sqrt{200}$. "Subgraph" is our approach, "Random Walk" the approach of [6], "Random" the nearly-optimal randomized construction for unconstrained group testing.



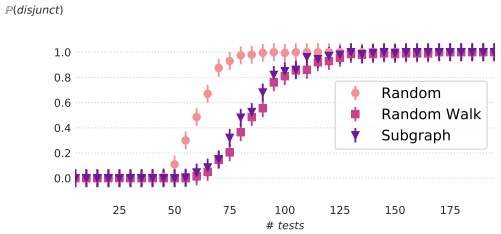
Random Regular Graph, $n = 16, m = 32$



Complete Graph, $n = 9, m = 36$

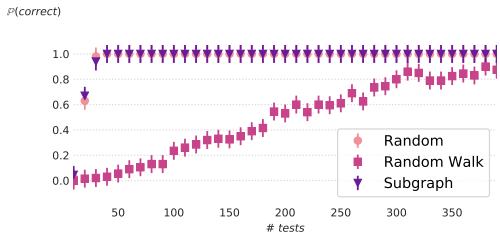


Fat Tree, $n = 20, m = 32$

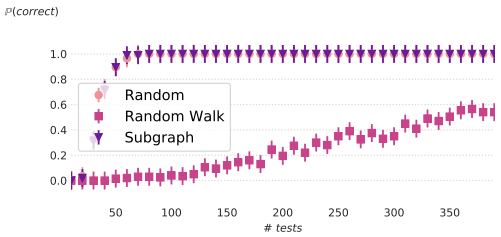


Hypercube, $n = 16, m = 32$

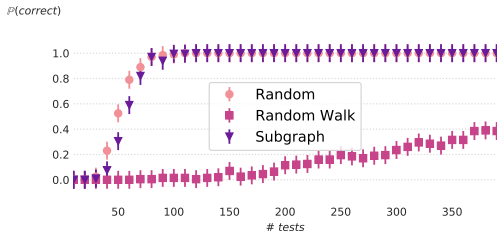
Figure 5 Probability that a randomly generated test matrix with a certain number of tests is 2-disjunct for various graphs. Each point is the mean of 200 trials, and with error bars of $1/\sqrt{200}$. "Subgraph" is our approach, "Random Walk" the approach of [6], "Random" the nearly-optimal randomized construction for unconstrained group-testing.



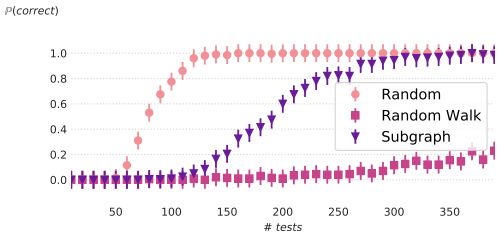
Fat Tree, $n = 80, m = 256, d = 1$



Fat Tree, $n = 80, m = 256, d = 2$



Fat Tree, $n = 80, m = 256, d = 3$



Fat Tree, $n = 80, m = 256, d = 5$

Figure 6 The probability that tests generated from various schemes with a certain number of tests correctly identifies d random failed edges. "Subgraph" is our approach, "Random Walk" the approach of [6], "Random" the nearly-optimal randomized construction for unconstrained group testing.

References

1 Miklós Ajtai, János Komlós, and Endre Szemerédi. Largest random component of a k-cube. *Combinatorica*, 2(1):1–7, 1982. doi:10.1007/BF02579276.

2 Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A scalable, commodity data center network architecture. *SIGCOMM*, 2008. URL: http://dblp.org/rec/conf/sigcomm/Al-FaresLV08.

3 Yigal Bejerano and Rajeev Rastogi. Robust Monitoring of Link Delays and Faults in IP Networks. *INFOCOM*, 1:134–144, 2003. doi:10.1109/INFCOM.2003.1208666.

4 Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, and David Walker. P4 - programming protocol-independent packet processors. *Computer Communication Review*, 44(3):87–95, 2014. doi:10.1145/2656877.2656890.

5 Rui Castro, Mark Coates, Gang Liang, Robert Nowak, and Bin Yu. Network Tomography: Recent Developments. *Statistical Science*, 19(3):499–517, August 2004. doi:10.1214/088342304000000422.

6 Mahdi Cheraghchi, Amin Karbasi, Soheil Mohajer, and Venkatesh Saligrama. Graph-Constrained Group Testing. *CoRR*, 2010. URL: http://dblp.org/rec/journals/corr/abs-1001-1445.

7 Fan Chung and Linyuan Lu. The Volume of the Giant Component of a Random Graph with Given Expected Degrees. *SIAM J. Discrete Math.*, 20(2):395–411, January 2006. doi:10.1137/050630106.

8 Amogh Dhamdhere, Renata Teixeira, Constantine Dovrolis, and Christophe Diot. NetDiagnoser - troubleshooting network unreachabilities using end-to-end probes and routing data. *CoNEXT*, page 1, 2007. doi:10.1145/1364654.1364677.

9 Michael Dinitz, Michael Schapira, and Gal Shahaf. Large low-diameter graphs are good expanders. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, pages 71:1–71:15, 2018. doi:10.4230/LIPIcs.ESA.2018.71.

10 Robert Dorfman. The detection of defective members of large populations. *The Annals of Mathematical Statistics*, 14(4):436–440, 1943.

11 Ding-Zhu Du and Frank K Hwang. *Combinatorial Group Testing and Its Applications*, volume 12 of *Series on Applied Mathematics*. World Scientific Publishing Co. Pte. Ltd., 2 edition, 1999. doi:10.1142/9789812798107.

12 Nick G Duffield. Network Tomography of Binary Network Performance Characteristics. *IEEE Trans. Information Theory*, 2006. URL: https://dblp.org/rec/journals/tit/Duffield06.

13 Arkadii Georgievich D'yachkov and Vladimir Vasil'evich Rykov. Bounds on the length of disjunctive codes. *Problemy Peredachi Informatsii*, 18(3):7–13, 1982.

14 Jack Edmonds and Ellis L Johnson. Matching, Euler tours and the Chinese postman. *Mathematical Programming*, 5(1):88–124, 1973. doi:10.1007/BF01580113.

15 Paul Erdös and Alfréd Rényi. On random graphs I. *Publicationes Mathematicae Debrecen*, 6:290–298, 1959.

16 Alan M Frieze, Michael Krivelevich, and Ryan R Martin. The emergence of a giant component in random subgraphs of pseudo-random graphs. *Random Struct. Algorithms*, 24(1):42–50, 2004. doi:10.1002/rsa.10100.

17 Charles Miller Grinstead and James Laurie Snell. *Introduction to probability*. American Mathematical Soc., 2012.

18 Nicholas J A Harvey, Mihai Patrascu, Yonggang Wen, Sergey Yekhanin, and Vincent W S Chan. Non-Adaptive Fault Diagnosis for All-Optical Networks via Combinatorial Group Testing on Graphs. *INFOCOM*, pages 697–705, 2007. doi:10.1109/INFCOM.2007.87.

19 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(04):439–562, October 2006. doi:10.1090/S0273-0979-06-01126-8.

20 Amin Karbasi and Morteza Zadimoghaddam. Sequential group testing with graph constraints. In *2012 IEEE Information Theory Workshop*, pages 292–296. IEEE, 2012. doi:10.1109/itw.2012.6404678.

21 David R Karger. Using Randomized Sparsification to Approximate Minimum Cuts. *SODA*, 1994. URL: https://dblp.org/rec/conf/soda/Karger94.

22 Michael Krivelevich and Benny Sudakov. The phase transition in random graphs - A simple proof. *Random Struct. Algorithms*, 2013. URL: https://dblp.org/rec/journals/rsa/KrivelevichS13.

23 Charles E Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE transactions on Computers*, 100(10):892–901, 1985.

24 Hung Xuan Nguyen and Patrick Thiran. The Boolean Solution to the Congested IP Link Location Problem - Theory and Practice. *INFOCOM*, pages 2117–2125, 2007. doi:10.1109/INFCOM.2007.245.

25 Ely Porat and Amir Rothschild. Explicit Non-Adaptive Combinatorial Group Testing Schemes. *WINE*, cs.DS, 2007. arXiv:0712.3876v5.

26 Arjun Roy, Hongyi Zeng, Jasmeet Bagga, and Alex C Snoeren. Passive Realtime Datacenter Fault Detection and Localization. *NSDI*, 2017. URL: https://dblp.org/rec/conf/nsdi/RoyZBS17.

27 Miklós Ruszinkó. On the Upper Bound of the Size of the r-Cover-Free Families. *Journal of Combinatorial Theory*, pages 302–310, 1994. URL: https://dblp.org/rec/journals/jct/Ruszinko94.

28 Bruce Spang and Mary Wootters. Unconstraining graph-constrained group testing. *arXiv preprint*, 2018. arXiv:1809.03589.

29 Asaf Valadarsky, Gal Shahaf, Michael Dinitz, and Michael Schapira. Xpander - Towards Optimal-Performance Datacenters. *CoNEXT*, 2016. URL: http://dblp.org/rec/conf/conext/ValadarskySDS16.

30 Hongyi Zeng, Peyman Kazemian, George Varghese, and Nick McKeown. Automatic Test Packet Generation. *IEEE/ACM Transactions on Networking*, 22(2):554–566, April 2013. doi:10.1109/TNET.2013.2253121.

A Instantiations of Theorem 8

In this appendix, we instantiate Theorem 8 for several families of graphs and compare them to existing results. We remark that some of these families (D-regular expanders, or $G(n,p)$) are natural candidates, while others (like a barbell graph) are concocted to show the difference between our theorem and that of previous work. A summary is shown in Table 2, and we go into the details below.

A.1 Complete Graphs

The mixing time for a complete graph is constant, so [6] gives an optimal construction requiring $O(d^2 \log(m/d))$ tests. Theorem 8 gives the same result.

A.2 D-Regular Expander Graphs with Constant Spectral Gap

D-regular expander graphs are D-regular graphs which are very "well-connected." One way of measuring this is the *spectral gap*, that is the difference between the largest eigenvalue D of the adjacency matrix A_G and the second-largest eigenvalue, λ. (We refer the reader to the excellent survey [19] for more background on expander graphs.) We consider families of D-regular graphs G whose second largest eigenvalue λ is bounded away from D by a constant: $\lambda \leq D(1-c)$ for some constant $c \in (0,1)$ independent of n.

For larger $D = \Omega(\log^2 n)$, [6] show that $O(d^2 \log^3(m))$ tests are sufficient, which is optimal up to logarithmic factors. For smaller D, in particular when D is a constant, the best previously known result guarantees $O(d^3 \log(m/d))$ tests [18].

As we will see below, Theorem 8 guarantees $O(d^2 \log(m/d))$ tests are sufficient for all D. In order to apply Theorem 8, we relate the second largest eigenvalue to edge-expansion as follows:

▶ **Theorem 13** (See [19] Theorem 4.11). *Let $G = (V, E)$ be a finite, connected, D-regular graph and let λ be its second eigenvalue. Then G is $(1/2, \alpha)$-edge expander with*

$$\frac{D-\lambda}{2} \leq \alpha \leq \sqrt{2D(D-\lambda)}.$$

By plugging in Theorem 13 to Theorem 8 (with constant and sufficiently small $\delta > 0$) we obtain the following corollary:

▶ **Corollary 14.** *Let G be a D-regular expander with second largest eigenvalue $\lambda \leq D(1-c)$ for some constant $c > 0$. Then for any $d < cD/2$, there is a collection $\mathcal{T}$ of connected subgraph tests so that $\mathcal{T}$ is d-disjunct and $|\mathcal{T}| = O(d^2 \log(m/d))$.*

Notice that by Proposition 9, the restriction that $d \leq d_0$ for some $d_0 = O(D)$ is necessary.

A.3 Erdös-Rényi Graphs

Like the D-regular expanders above, an Erdös-Rényi random graph $G(n, p)$ on n nodes with parameter $0 \leq p \leq 1$ is well-connected, and has good spectral properties with high probability. However, these graphs are not D-regular so we consider them separately.

For larger $p = \Omega(\log^2 n/n)$, [6] show that $O(d^2 \log^3(m))$ tests are sufficient to guarantee d-disjunctness. Theorem 8 can improve this to $O(d^2 \log(m/d))$, with only the restriction $p \geq p_0$ for some $p_0 = \Omega(1/n)$.

In order to apply Theorem 8, we use the following lemma from [6], which implies that $G(n, D/n)$ for $D = \Omega(d \log n)$ has $(1/2, \alpha)$ edge expansion for $\alpha \geq (2+\varepsilon)d$. Theorem 8 immediately implies that $O(d^2 \log(m/d))$ tests are sufficient.

▶ **Lemma 15** ([6] Lemma 32). *For every $\phi < 1/2$ there is an $\alpha > 0$ such that a random graph $G = G(n, p)$ with $p \geq \alpha \ln n/n$ has edge expansion $\alpha \geq \phi D$ with probability $1 - o(1)$.*

A.4 Barbells

One of the advantages of our result over previous work is that the notion of (β, α)-edge expansion captures a more general notion of "well-connected" than is captured by minimum cuts or mixing times. As an extreme example of this, consider a barbell graph G, which we define as two copies of the complete graph $K_{n/2}$ on $n/2$ vertices, connected by one edge e (Figure 7).

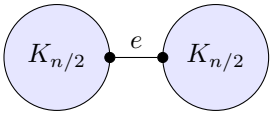


Figure 7 A barbell graph.

This graph is great for graph-constrained group testing: we test the connecting edge e on its own, then then use $O(d^2 \log(m/d))$ tests to identify up to d failures in each of the two copies of $K_{n/2}$. Thus, $O(d^2 \log(m/d))$ tests are sufficient. However, this is a worst-case

graph for existing work. It has a minimum cut of one edge, so [18] only allows $d = 1$. The mixing time of this graph is quite large: the probability of reaching the center edge is $O(1/n)$, so certainly $\tau = \Omega(n)$. The degree condition of [6] is not satisfied as $D \leq n/2$ which is not $O(1/n^2)$. However, even if we could ignore this condition, [6] would use at least $\tilde{O}(n^2 d^2 \log(m/d))$ tests (or, therefore $O(m)$ tests since this is the naive solution).

On the other hand, our results using (β, α)-expansion match the intuition that this example should be easy. Setting $\beta = \frac{1}{4}$ and $\alpha \geq \frac{n}{2} - \frac{n}{4} = \frac{n}{4}$, Theorem 8 gives an $O(d^2 \log(m/d))$ bound.

This is example is meant to highlight the difference between our work and existing work. While the barbell graph is unlikely to be used in practice, it illustrates the intuition that (β, α)-connectivity does better capture somewhat "clustery" graphs – that is, graphs with higher connectivity in some areas than in others – than either minimum cuts or mixing time. This notion may be useful for real-life networks; for example, networks may have higher connectivity within a rack than between racks.

A.5 Fat-Trees

The "Fat-Tree" graph [23] was originally designed for use in supercomputers, and is now widely used in datacenter networks [2]. As the name suggests, this is a "fattened" tree, where the fatness (number of links) near the top of the tree is greater than the fatness near the leaves.

We illustrate the fat-tree topology in Figure 3. For some parameter $D > 0$, each node in the fat-tree has degree D or $D/2$. The Fat-Tree consists of a "core" of $\left(\frac{D}{2}\right)^2$ nodes and a set of D pods. Each pod consists of a complete bipartite graph where each layer has $D/2$ nodes. Each node in the core has one edge to one node in the top layer of each pod.

The minimum cut of the fat tree is $D/2$ and it has $n = 5D^2/4$ nodes. If $D/2 \geq 5(d+1)\log(5D^2/4)$, or equivalently $d \leq \frac{D}{20\log(5D/4)} - 1 = O(D/\log D)$, Proposition 2 gives an upper bound of $O(d^2 \log(m/d))$ on the number of tests required.

On the other hand, the guarantee of [18] gives a graph-constrained group testing scheme for the fat-tree topology with $O(d^3 \log(m/d))$ tests. It is not clear what the mixing time of the fat-tree is, so we do not compare to the bound from [6] which is in terms of the mixing time. (However, as shown in Section 6, it seems that our approach requires fewer tests than that of [6] on this graph).

Near-Neighbor Preserving Dimension Reduction for Doubling Subsets of ℓ_1

Ioannis Z. Emiris
Department of Informatics & Telecommunications,
National & Kapodistrian University of Athens, Greece
ATHENA Research & Innovation Center, Greece
emiris@di.uoa.gr

Vasilis Margonis
Department of Informatics & Telecommunications,
National & Kapodistrian University of Athens, Greece
basilis.math@gmail.com

Ioannis Psarros[1]
Institute of Computer Science, University of Bonn, Germany
ipsarros@uni-bonn.de

Abstract

Randomized dimensionality reduction has been recognized as one of the fundamental techniques in handling high-dimensional data. Starting with the celebrated Johnson-Lindenstrauss Lemma, such reductions have been studied in depth for the Euclidean (ℓ_2) metric, but much less for the Manhattan (ℓ_1) metric. Our primary motivation is the approximate nearest neighbor problem in ℓ_1. We exploit its reduction to the decision-with-witness version, called approximate *near* neighbor, which incurs a roughly logarithmic overhead. In 2007, Indyk and Naor, in the context of approximate nearest neighbors, introduced the notion of nearest neighbor-preserving embeddings. These are randomized embeddings between two metric spaces with guaranteed bounded distortion only for the distances between a query point and a point set. Such embeddings are known to exist for both ℓ_2 and ℓ_1 metrics, as well as for doubling subsets of ℓ_2. The case that remained open were doubling subsets of ℓ_1. In this paper, we propose a dimension reduction by means of a *near* neighbor-preserving embedding for doubling subsets of ℓ_1. Our approach is to represent the pointset with a carefully chosen covering set, then randomly project the latter. We study two types of covering sets: c-approximate r-nets and randomly shifted grids, and we discuss the tradeoff between them in terms of preprocessing time and target dimension. We employ Cauchy variables: certain concentration bounds derived should be of independent interest.

2012 ACM Subject Classification Theory of computation → Nearest neighbor algorithms; Mathematics of computing → Dimensionality reduction

Keywords and phrases Approximate nearest neighbor, Manhattan metric, randomized embedding

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.47

Category RANDOM

Related Version A preliminary version is available at `https://arxiv.org/abs/1902.08815`.

Funding *Ioannis Z. Emiris*: Partially supported by the European Union's H2020 research and innovation programme under grant agreement No. 734242 (LAMBDA).
Ioannis Psarros: Generously supported by the Hausdorff Center for Mathematics.

Acknowledgements IZE is member of team AROMATH, joint between INRIA Sophia-Antipolis and NKUA. IP thanks Robert Krauthgamer for useful discussions on the topic.

[1] This work was done while the third author was a PhD candidate in the Department of Informatics & Telecommunications, National & Kapodistrian University of Athens, Greece.



© Ioannis Z. Emiris, Vasilis Margonis, and Ioannis Psarros;
licensed under Creative Commons License CC-BY



1 Introduction

Proximity search is a fundamental computational problem with several applications in Computer Science and beyond. Proximity problems in metric spaces of low dimension have been typically handled by methods which discretize the space and therefore are affected by the curse of dimensionality, making them unfit for high-dimensional spaces. In the past two decades, the increasing need for analyzing high-dimensional data led researchers to devise randomized and approximation algorithms with polynomial dependence on the dimension.

A fundamental proximity problem is Approximate Nearest Neighbor search. By known reductions [11], one can (up to polylogarithmic factors) focus on the decision version with witness, namely the (c, R)-Approximate Near Neighbor problem:

▶ **Definition 1** (Approximate Near Neighbor). *Let (X, d_X) be a metric space. Given $P \subseteq X$ and reals $R > 0$, $c \geq 1$, build a data structure $\mathcal{S}$ that, given a query point $q \in X$, performs as follows:*

- *If the nearest neighbor of q lies within distance at most R, then $\mathcal{S}$ is allowed to report any point $p^* \in P$ such that $d_X(q, p^*) \leq cR$.*
- *If all points lie at distance more than cR from q, then $\mathcal{S}$ should return $\perp$.*

In general, $\mathcal{S}$ returns either a point at distance $\leq cR$ or $\perp$, even when none of the above two cases occurs.

From now on, we assume $R = 1$ because we can re-scale the data set, and we refer to this problem as c-ANN, or simply ANN. We focus on subsets of ℓ_1^d: the input dataset consists of n vectors in $\mathbb{R}^d$ and the distance function is the standard ℓ_1 norm $\|\cdot\|_1$. Note that all logarithms are base 2.

Previous work. Some highlights in the study of data structures for high-dimensional normed spaces are the various variants, proofs, and applications of the Johnson Lindenstrauss Lemma (e.g. [1, 2, 3]), sketches based on p-stable distributions [14], and Locality Sensitive Hashing (e.g. [15, 4, 5]). In the core of most high-dimensional solutions lies the fact that for certain metric spaces e.g. $\ell_p, p \in [1, 2]$, the distance can be efficiently sketched. Spaces which are considered to be harder in this context, such as ℓ_∞, can also be treated [13], and are very interesting since they can be used as host spaces for various norms [6].

Significant amount of work has been undertaken for pointsets of low doubling dimension, since it is today one of the primary paradigms for capturing input structure (formal definitions in the next section). For any finite metric space X of doubling dimension $\dim(X)$, there exists a data structure [12, 9] with expected preprocessing time $O(2^{\dim(X)} n \log n)$, space usage $O(2^{\dim(X)} n)$ (or even $O(n)$) and query time $O(2^{\dim(X)} \log n + \varepsilon^{-O(\dim(X))})$.

In [16], they introduced the notion of nearest-neighbor preserving embeddings, and it was proven that in this context one can achieve dimension reduction for doubling subsets of ℓ_2, with the target dimension depending only on the dataset's doubling dimension. Even before, Indyk [14] had introduced a randomized embedding for dimension reduction in ℓ_1, which is suitable for proximity search purposes, and it achieves target dimension polylogarithmic in the size of the pointset. Naturally, such approaches can be easily combined with any known data structure to be used in the projection space. Randomized embeddings have been recently used in the ANN context [8], for doubling subsets of ℓ_p, $2 < p < \infty$.

It is known that dimension reduction in ℓ_1 cannot be achieved in the same generality as in ℓ_2, even assuming that the pointset is of low doubling dimension [18]: there are arbitrarily large n-point subsets $P \subseteq \ell_1$ which are doubling with constant 6, such that every embedding

with distortion D of P into ℓ_1^k requires dimension $n^{\Omega(1/D^2)}$. Aiming for more restrictive guarantees, e.g. preserving distances within some pre-defined range, is a relevant workaround. Then, dimension reduction techniques for doubling subsets of ℓ_p, $p \in [1,2]$, exist [7], but they rely on partition algorithms which require the whole pointset to be known in advance. Hence, applicability of such techniques is quite limited and, specifically, it is not clear whether they can be used in an online setting where query points are not known beforehand.

Contribution. In this paper, we establish two non-linear *near* neighbor-preserving embeddings for doubling subsets of ℓ_1^d. We use a definition which is essentially a modified version of the nearest neighbor preserving embedding of [16]: the guarantees which are required are weaker since we consider the decision version of the problem, therefore the embedding depends on some range parameter $R > 0$.

▶ **Definition 2** (Near-neighbor preserving embedding). *Let (Y, d_Y), (Z, d_Z) be metric spaces and $X \subseteq Y$. A distribution over mappings $f : Y \to Z$ is a* near-neighbor preserving embedding *with range $R > 0$, distortion $D \geq 1$ and probability of correctness $\mathcal{P} \in [0,1]$ if for every $\alpha \geq D$ and any $q \in Y$, if $x \in X$ is such that $d_Y(x,q) \leq R$, then with probability at least $\mathcal{P}$,*

- $d_Z(f(x), f(q)) \leq D \cdot R$,
- $\forall p \in X : \ d_Y(p,q) > D \cdot \alpha \cdot R \implies d_Z(f(p), f(q)) > \alpha \cdot R$.

Considering a pointset $P \subset \ell_1^d$ of cardinality n, our results concern ℓ_1^k as the target space, where k depends on the doubling dimension of P. We assume that $R = 1$, since we can rescale the dataset. More specifically:

1. In Theorem 10, we prove that for every $\varepsilon \in (0, 1/2)$ and $c \geq 1$, there is a randomized mapping $h : \ell_1^d \to \ell_1^k$ that can be computed in time $\tilde{O}(dn^{1+1/\Omega(c)})$ and is *near* neighbor-preserving for P with distortion $1+6\varepsilon$ and probability of correctness $\Omega(\varepsilon)$, where

$$k = (\log \lambda_P \cdot \log(c/\varepsilon))^{\Theta(1/\varepsilon)} / \zeta(\varepsilon),$$

for a function $\zeta(\varepsilon) > 0$ depending only on ε. Although the mapping h depends on the pointset, the parameter c is user-defined and therefore provides a trade-off between preprocessing time and target dimension.

2. In Theorem 13, we show that for every $\varepsilon \in (0, 1/2)$, there is a randomized mapping $h' : \ell_1^d \to \ell_1^k$ that can be computed in time $O(dkn)$ and is *near* neighbor-preserving for P with distortion $1+6\varepsilon$ and probability of correctness $\Omega(\varepsilon)$, where

$$k = (\log \lambda_P \cdot \log(d/\varepsilon))^{\Theta(1/\varepsilon)} / \zeta(\varepsilon),$$

for a function $\zeta(\varepsilon) > 0$ depending only on ε. In this case, the function h' is oblivious to P and well-defined over the whole space, but the target dimension depends on d.

On the low-preprocessing-time extreme, one can embed the dataset in near-linear time, but the target dimension is polynomial in $\log \log n$. This is to be juxtaposed to the analogous result by Indyk [14], which provides with target dimension polynomial in $\log n$, without any assumption on the doubling dimension of the dataset. On the other hand, one can obtain a preprocessing time of $dn^{1+\delta}$ for any constant $\delta > 0$, and target dimension which depends solely on the doubling dimension.

Techniques. Both embeddings consist of two basic components. First, we represent the pointset P with an ε-covering set, and then we apply a random linear projection à la Indyk [14] to that set, using Cauchy variables.

The role of the covering set is to exploit the doubling dimension of P. In the analogous result for ℓ_2 [16], no representative sets were used; the mapping was just a random linear projection of P. In the case of ℓ_1 however, a similar analysis of a linear projection with Cauchy variables without these representative sets seems to be impossible, since the Cauchy distribution is heavy tailed.

In Theorem 10, we consider c-approximate r-nets as a covering set. Inspired by the algorithm of [10] for ℓ_2, we design an algorithm that computes a c-approximate r-net in ℓ_1 in subquadratic –but superlinear– time. On the other hand, Theorem 13 relies on randomly shifted grids, which can be computed in linear time, but are inferior to nets in terms of capturing the doubling dimension of the pointset.

To bound the distortion incurred by the randomized projection, we exploit the 1-stability property of the Cauchy distribution. To this end, we prove a concentration bound for sums of independent Cauchy variables that should be of interest beyond the scope of this paper. To overcome the technical difficulties associated with the heavy tails of the Cauchy distribution, we study sums of *square roots* of Cauchy variables, where in [14], Indyk considers sums of *truncated* Cauchy variables instead. Although our concentration bound is rather weak, it is sufficient for our purposes and its analysis is much simpler compared to Indyk's.

Algorithmic implications. Our results show that efficient dimension reduction for doubling subsets of ℓ_1 is possible, in the context of ANN. In particular, these results imply efficient sketches, meaning that one can solve ANN with minimal storage per point. Dimension reduction also serves as a problem reduction from a high-dimensional hard instance to a low-dimensional easy instance. Since the algorithms presented in this paper are quite simple, they should also be of practical interest: they easily extend the scope of any implementation which has been optimized to solve the problem in low dimension, so that it may handle high-dimensional data.

Our embedding can be combined with the bucketing method of [11] for the $(1+\varepsilon)$-ANN problem in ℓ_1^d. For instance, setting $c = \log n$ in Theorem 10, yields preprocessing time $dn^{1+o(1)}$, space $n^{1+o(1)}$ and query time $O(d)\cdot(\log \lambda_P \cdot \log\log n)^{O(1/\varepsilon)}$ assuming that the doubling dimension is a fixed constant. This improves upon existing results: the query time of [17] depends on the aspect ratio of the dataset, while the data structures of [12, 9] support queries with time complexity which depends exponentially on the doubling dimension. However, it is worth noting that one could potentially improve the results of [17, 12, 9] in the special case of ℓ_1, by employing ANN data structures with fast query time, in order to accelerate the traversal of the net-tree. Hence, while our result gives a simple framework for exploiting the intrinsic dimension of doubling subsets of ℓ_1, it is unlikely that it shall improve upon simple variants of previous results in terms of complexity bounds.

Organization. The next section introduces basic concepts and some relevant existing results. Section 3 establishes a concentration bound on sums of independent Cauchy variables. Section 4, achieves dimensionality reduction by means of representing the pointset by a carefully chosen net, while Section 5 employs randomly shifted grids for the same task. We conclude with discussion of results and potential improvements.

2 Preliminaries

In this section, we define basic notions about doubling metrics and present useful previous results.

▶ **Definition 3.** *Consider any metric space (X, d_X) and let $B(p,r) = \{x \in X \mid \mathrm{d}_X(x,p) \leq r\}$. The* doubling constant *of X, denoted λ_X, is the smallest integer λ_X such that for any $p \in X$ and $r > 0$, the ball $B(p,r)$ can be covered by λ_X balls of radius $r/2$ centered at points in X.*

The *doubling dimension* of (X, d_X) is defined as $\log \lambda_X$. Nets play an important role in the study of embeddings, as well as in designing efficient data structures for doubling metrics.

▶ **Definition 4.** *For $c \geq 1$, $r > 0$ and metric space (V, d_V), a c-approximate r-net of V is a subset $\mathcal{N} \subseteq V$ such that no two points of $\mathcal{N}$ are within distance r of each other, and every point of V lies within distance at most $c \cdot r$ from some point of $\mathcal{N}$.*

▶ **Theorem 5.** *Let $P \subset \ell_1^d$ such that $|P| = n$. Then, for any $c > 0$, $r > 0$, one can compute a c-approximate r-net of P in time $\tilde{O}(dn^{1+1/c'})$, where $c' = \Omega(c)$. The result is correct with high probability. The algorithm also returns the assignment of each point of P to the point of the net which covers it.*

Proof. We employ some basic ideas from [11]. An analogous result for ℓ_2 is stated in [10]. First, we assume $r = 1$, since we are able to re-scale the point set. Now, we consider a randomly shifted grid with side-length 2. The probability that two points $p, q \in P$ fall into the same grid cell, is at least $1 - \|p-q\|_1/2$. For each non-empty grid cell we snap points to a grid: each coordinate is rounded to the nearest multiple of $\delta = 1/10dc$. Then, coordinates are multiplied by $1/\delta$ and each point $x = (x_1, \ldots, x_d) \in [2\delta]^d$ is mapped to $\{0,1\}^{2d/\delta}$ by a function G as follows: $G(x) = (g(x_1), \ldots, g(x_d))$, where $g(z)$ is a binary string of z ones followed by $2/\delta - z$ zeros. For any two points p, q in the same grid cell, let $f(p)$,$f(q)$ be the two binary strings obtained by the above mapping. Notice that,

$$\|f(p) - f(q)\|_1 \in (2/\delta) \cdot \|p-q\|_1 \pm 1.$$

Hence,

$$\|p-q\|_1 \leq 1 \implies \|f(p) - f(q)\|_1 \leq (2/\delta) + 1,$$

$$\|p-q\|_1 \geq c \implies \|f(p) - f(q)\|_1 \geq (2/\delta) \cdot c - 1.$$

Now, we employ the LSH family of [11], for the Hamming space. After standard concatenation, we can assume that the family is $(\rho, c'\rho, n^{-1/c'}, n^{-1})$-sensitive, where $\rho = (2/\delta) + 1$ and $c' = \Omega(c)$. Let $\alpha = n^{-1/c'}$ and $\beta = n^{-1}$.

Notice that for the above two-level hashing table we obtain the following guarantees. Any two points $p, q \in P$, such that $\|p-q\|_1 \leq 1$, fall into the same bucket with probability $\geq \alpha/2$. Any two points $p, q \in P$, such that $\|p-q\|_1 \geq c$, fall into the same bucket with probability $\leq \beta$.

Finally, we independently build $k = \Theta(n^{1/c'} \log n)$ hashtables as above, where the random hash function is defined as a concatenation of the function which maps points to their grid cell id and one LSH function. We pick an arbitrary ordering $p_1, \ldots, p_n \in P$. We follow a greedy strategy in order to compute the approximate net. We start with point p_1, and we add it to the net. We mark all (unmarked) points which fall at the same bucket with p_1, in one of the k hashtables, and are at distance $\leq cr$. Then, we proceed with point p_2. If p_2 is unmarked, then we repeat the above. Otherwise, we proceed with p_3. The above iteration stops when all points have been marked. Throughout the procedure, we are able to store one pointer for each point, indicating the center which covered it.

Correctness. The probability that a good pair p, q does not fall into the same bucket for any of the k hashtables is $\leq (1-\alpha/2)^k \leq n^{-10}$. Hence, with high probability, the packing property holds, and the covering property holds because the above algorithm stops when all points are marked.

Running time. The time to build the k hashtables is $k \cdot n = \tilde{O}(n^{1+1/c'})$. Then, at most n queries are performed: for each query, we investigate k buckets and the expected number of false positives is $\leq k \cdot n^2 \cdot \beta = \tilde{O}(n^{1+1/c'})$. Hence, if we stop after having seen a sufficient amount of false positives, we obtain time complexity $\tilde{O}(n^{1+1/c'})$ and the covering property holds with constant probability. We can repeat the above procedure $O(\log n)$ times to obtain high probability of success. ◀

The main result in the context of randomized embeddings for dimension reduction in ℓ_1^d is the following theorem, which exploits the 1-stability property of Cauchy random variables and provides with an asymmetric guarantee: The probability of non-contraction is high, but the probability of non-expansion is constant. Nevertheless, this asymmetric property is sufficient for proximity search.

▶ **Theorem 6** (Thm 5, [14]). *For any $\varepsilon \leq 1/2$, $\delta > 0$, $\varepsilon > \gamma > 0$ there is a probability space over linear mappings $f : \ell_1^d \to \ell_1^k$, where $k = (\ln(1/\delta))^{1/(\varepsilon-\gamma)}/\zeta(\gamma)$, for a function $\zeta(\gamma) > 0$ depending only on γ, such that for any pair of points $p, q \in \ell_1^d$:*

$$\mathbf{Pr}\left[\, \|f(p) - f(q)\|_1 \leq (1-\varepsilon)\, \|p-q\|_1 \,\right] \leq \delta,$$

$$\mathbf{Pr}\left[\, \|f(p) - f(q)\|_1 \geq (1+\varepsilon)\, \|p-q\|_1 \,\right] \leq \frac{1+\gamma}{1+\varepsilon}.$$

Note that the embedding is defined as $f(u) = Au/T$, where A is a $k \times d$ matrix with each element being an i.i.d. Cauchy random variable. In addition, T is a scaling factor defined as the expectation of a sum of truncated Cauchy variables, such that $T = \Theta(k \log(k/\varepsilon))$ (see Lemma 5 in [14]).

One key observation here is that given a pointset P in a space of bounded aspect ratio Φ, one can directly employ Theorem 6. The number of points can be upper bounded by a function of λ_P and Φ, and hence the new dimension, k, depends only on these parameters. This paper proves better bounds than the ones of Theorem 6 for doubling subsets of ℓ_1^d, without any assumption on the aspect ratio.

3 Concentration bounds for Cauchy variables

In this section, we prove some basic properties of the Cauchy distribution, which serves as our main embedding tool.

Let $C_{\mathcal{D}}$ denote the Cauchy distribution with density $c(x) = (1/\pi)/(1+x^2)$. One key property of the Cauchy distribution is the so-called 1-stability property: Let $v = (v_1, \ldots, v_k) \in \mathbb{R}^k$ and $X_1, \ldots, X_k$ be i.i.d. random variables following $C_{\mathcal{D}}$, then $\sum_{j=1}^k X_i v_i$ is distributed as $X \cdot \|v\|_1$, where $X \sim C_{\mathcal{D}}$.

The Cauchy distribution has undefined mean. However, for $0 < q < 1$, the mean of the q-th power of a Cauchy random variable can be defined. More specifically, for some $X \sim C_{\mathcal{D}}$ we have

$$\mathbb{E}\left[|X|^{1/2}\right] = \frac{2}{\pi}\int_0^\infty \frac{\sqrt{x}}{1+x^2}\,\mathrm{d}x = \frac{2}{\pi}\frac{\pi}{\sqrt{2}} = \sqrt{2}.$$

The following lemma provides a bound for the moment-generating function of $|X|^{1/2}$.

▶ **Lemma 7.** *Let $X \sim C_{\mathcal{D}}$. Then for any $\beta > 1$:*

$$\mathbb{E}\left[\exp\left(-\beta|X|^{1/2}\right)\right] \leq \frac{2}{\beta}.$$

Proof. For any constant β,

$$\int_0^1 \mathrm{e}^{-\beta x^{1/2}}\,\mathrm{d}x = \frac{2}{\beta^2}\left(1 - \frac{\beta+1}{\mathrm{e}^{\beta}}\right).$$

Then, for any $\beta > 1$,

$$\begin{aligned}
\mathbb{E}\left[\exp\left(-\beta|X|^{1/2}\right)\right] &= \int_{-\infty}^{\infty} \mathrm{e}^{-\beta|x|^{1/2}} \cdot c(x)\,\mathrm{d}x = \frac{2}{\pi}\int_0^{\infty} \mathrm{e}^{-\beta x^{1/2}} \cdot \frac{1}{1+x^2}\,\mathrm{d}x \\
&= \frac{2}{\pi}\int_0^{1} \mathrm{e}^{-\beta x^{1/2}} \cdot \frac{1}{1+x^2}\,\mathrm{d}x + \frac{2}{\pi}\int_1^{\infty} \mathrm{e}^{-\beta x^{1/2}} \cdot \frac{1}{1+x^2}\,\mathrm{d}x \\
&\leq \frac{2}{\pi}\int_0^{1} \mathrm{e}^{-\beta x^{1/2}}\,\mathrm{d}x + \frac{2}{\pi}\int_1^{\infty} \mathrm{e}^{-\beta} \cdot \frac{1}{1+x^2}\,\mathrm{d}x \\
&= \frac{2}{\pi}\cdot\frac{2}{\beta^2}\left(1 - \frac{\beta+1}{\mathrm{e}^{\beta}}\right) + \frac{1}{2\mathrm{e}^{\beta}} \\
&\leq \frac{4}{\pi\beta^2} + \frac{1}{2\mathrm{e}^{\beta}} \\
&\leq \frac{2}{\beta}.
\end{aligned}$$

◀

Let $S := \sum_{j=1}^{k} |X_j|$ where each X_j is an i.i.d. Cauchy variable. To prove concentration bounds for S, we study the sum $\tilde{S} := \sum_{j=1}^{k} |X_j|^{1/2}$. By Hölder's Inequality, for any $x \in \mathbb{R}^d$ and $p > q > 0$,

$$\|x\|_p \leq \|x\|_q \leq d^{1/q-1/p}\,\|x\|_p\,.$$

Consequently, for $x = (X_1, \ldots, X_k) \in \mathbb{R}^k$, $p = 1$ and $q = 1/2$ we have that $S \leq \tilde{S}^2 \leq k \cdot S$, hence for any $t > 0$,

$$\mathbf{Pr}[S \leq t] \leq \mathbf{Pr}[\tilde{S} \leq \sqrt{tk}]. \tag{1}$$

We use the bound on the moment-generating function, to prove a Chernoff-type concentration bound for $\tilde{S}$, which by Eq. (1) translates into a concentration bound for S.

▶ **Lemma 8.** *For every $D > 1$,*

$$\mathbf{Pr}\left[\tilde{S} \leq \frac{\mathbb{E}[\tilde{S}]}{D}\right] \leq \left(\frac{10}{D}\right)^k.$$

Proof. Since X_j's are independent, $\mathbb{E}[\tilde{S}] = \sqrt{2}k$. Then, by Lemma 7 and Markov's inequality, for any $\beta > 1$, it follows that

$$\begin{aligned}
\mathbf{Pr}\left[\tilde{S} \leq \frac{\mathbb{E}[\tilde{S}]}{D}\right] &= \mathbf{Pr}\left[\exp(-\beta\tilde{S}) \geq \exp\left(-\beta \cdot \frac{\mathbb{E}[\tilde{S}]}{D}\right)\right] \\
&\leq \frac{\mathbb{E}[\exp(-\beta\tilde{S})]}{\exp(-\beta\,\mathbb{E}[\tilde{S}]/D)} \\
&= \frac{\mathbb{E}[\exp(-\beta|X_j|^{1/2})]^k}{\exp(-\beta\sqrt{2}k/D)} \\
&\leq \left(\frac{2}{\beta}\right)^k \cdot \mathrm{e}^{\sqrt{2}\beta k/D}.
\end{aligned}$$

Setting $\beta = D$ completes the proof. ◀

4 Net-based dimension reduction

In this section we describe the dimension reduction mapping for ℓ_1 via r-nets. Let $P \subset \ell_1^d$ be a set of n points with doubling constant λ_P. For some point $x \in \mathbb{R}^d$ and $r > 0$, we denote by $B_1(x, r)$ the ℓ_1-ball of radius r around x. The embedding is non-linear and is carried out in two steps.

First, we compute a c-approximate (ε/c)-net $\mathcal{N}$ of P with the algorithm of Theorem 5. Moreover, the algorithm assigns each point of P to the point of $\mathcal{N}$ which covered it. Let $g : P \to \mathcal{N}$ be this assignment. In the second step, for every $s \in \mathcal{N}$ and any query point $q \in \ell_1^d$, we apply the linear map of Theorem 6. That is, $f(s) = As/T$, where A is a $k \times d$ matrix with each element being an i.i.d. Cauchy random variable. Recall that value $T = \Theta(k \log(k/\varepsilon))$. By the 1-stability property of the Cauchy distribution, $f(s)$ is distributed as $\|s\|_1 \cdot (Y_1, \ldots, Y_k)$, where each Y_j is i.i.d. and $Y_j \sim C_{\mathcal{D}}$. Hence, $\|f(s)\|_1 = \|s\|_1 \cdot S$ where $S := \sum_j |Y_j|$.

We define the embedding to be $h = f \circ g$. We apply h to every point in P, and f to any query point q. It is clear from the properties of the net that g incurs an additive error of $\pm\varepsilon$ on the distance between q and any point in P, so it is sufficient to consider the distortion of f.

Our analysis consists of studying separately the following disjoint subsets of $\mathcal{N}$: Points that lie at distance at most D_0 from the query and points that lie at distance at least D_0, for some $D_0 > 1$ chosen appropriately. For the former set, we directly apply Theorem 6, as it has bounded diameter.

The next lemma guarantees the low distortion for points of the latter set, namely those that are sufficiently far from the query. We consider the sum of the square roots of each $|Y_j|$, i.e., $\tilde{S} = \sum_j |Y_j|^{1/2}$, in order to employ the tools of Section 3.

▶ **Lemma 9.** *Fix a query point $q \in \ell_1^d$. For any $\varepsilon \le 1/2$, $c \ge 1$, $\delta \in (0,1)$, there exists $D_0 = O(\log(k/\varepsilon))$ such that for $k = \Theta\left(\log^2 \lambda_P \cdot \log(c/\varepsilon) + \log(1/\delta)\right)$, with probability at least $1 - \delta$,*

$$\forall s \in \mathcal{N} : \ \|s - q\|_1 \ge D_0 \implies \|f(s) - f(q))\|_1 \ge 4.$$

Proof. Assume wlog that the query point is the origin $(0, \ldots, 0)$. For some $D_0 > 1$, we define the following subsets of $\mathcal{N}$:

$$N_i := \{s \in \mathcal{N} \mid D_i \le \|s\|_1 < D_{i+1}\}, \ D_i = 2^{2i} D_0, \ i = 0, 1, 2, \ldots$$

By the definition of doubling constant and the fact that two points of $\mathcal{N}$ lie at distance at least ε,

$$|N_i| \le \lambda_P^{\lceil \log(4cD_{i+1}/\varepsilon) \rceil} \le \lambda_P^{4\log(cD_{i+1}/\varepsilon)}.$$

Therefore, by the union bound, and Eq. (1):

$$\begin{aligned}
\mathbf{Pr}\left[\exists i \exists s \in N_i : \|f(s)\|_1 \le \frac{4\|s\|_1}{D_i}\right] &= \mathbf{Pr}\left[\exists i \exists s \in N_i : S \le \frac{4T}{D_i}\right] \\
&\le \sum_{i=0}^{\infty} |N_i| \, \mathbf{Pr}\left[\tilde{S} \le \frac{\sqrt{4kT}}{\sqrt{D_i}}\right] \\
&= \sum_{i=0}^{\infty} |N_i| \, \mathbf{Pr}\left[\tilde{S} \le \mathbb{E}[\tilde{S}] \cdot \sqrt{\frac{2T}{k 2^{2i} D_0}}\right].
\end{aligned}$$

By Lemma 8, for $D_0{=}\lceil 800T/k\rceil{=}\Theta(\log(k/\varepsilon))$ and $k > 4{\cdot}\log\lambda_P{\cdot}\log(cD_0/\varepsilon) + 2\log(2\lambda_P/\delta)$:

$$\begin{aligned}
\sum_{i=0}^{\infty} |N_i|\,\mathbf{Pr}\left[\tilde{S} \le \frac{\mathbb{E}[\tilde{S}]}{10\cdot 2^{i+1}}\right] &\le \sum_{i=0}^{\infty} \lambda_P^{4\log(cD_{i+1}/\varepsilon)}\left(\frac{1}{2^{i+1}}\right)^k \\
&= \sum_{i=0}^{\infty} \frac{2^{\log(\lambda_P)(4\log(cD_0/\varepsilon)+2i+2)}}{2^{k(i+1)}} \\
&\le \sum_{i=0}^{\infty} \frac{2^{\log(\lambda_P)\cdot 4\log(cD_0/\varepsilon)}\cdot 2^{2\log(\lambda_P)(i+1)}}{2^{(4\cdot\log\lambda_P\cdot\log(cD_0/\varepsilon))(i+1)}\cdot 2^{2\log(2\lambda_P/\delta))(i+1)}} \\
&\le \sum_{i=0}^{\infty} 2^{-2\log(2/\delta))(i+1)} \\
&= \sum_{i=0}^{\infty}\left(\frac{\delta^2}{4}\right)^i - 1 \\
&= \frac{\delta^2}{4-\delta^2} \\
&\le \delta.
\end{aligned}$$

Finally, for some large enough constant C, we demand that

$$k > C\left(\log\lambda_P\cdot\log(c\log k/\varepsilon) + \log(1/\delta)\right) > 4\cdot\log\lambda_P\cdot\log(cD_0/\varepsilon) + 2\log(2\lambda_P/\delta)$$

which is satisfied for $k = \Theta\left(\log^2\lambda_P\cdot\log(c/\varepsilon) + \log(1/\delta)\right)$. ◀

▶ **Theorem 10.** *Let $P \subset \ell_1^d$ such that $|P| = n$. For any $\varepsilon \in (0,1/2)$ and $c \ge 1$, there is a non-linear randomized embedding $h = f\circ g : \ell_1^d \to \ell_1^k$, where $k = (\log\lambda_P\cdot\log(c/\varepsilon))^{\Theta(1/\varepsilon)}/\zeta(\varepsilon)$, for a function $\zeta(\varepsilon) > 0$ depending only on ε, such that, for any $q \in \ell_1^d$, if there exists $p^* \in P$ such that $\|p^* - q\|_1 \le 1$, then, with probability $\Omega(\varepsilon)$:*

$$\begin{aligned}
&\|h(p^*) - f(q)\|_1 \le 1 + 3\varepsilon, \\
&\forall p \in P:\ \|p-q\|_1 > 1+9\varepsilon \implies \|h(p) - f(q)\|_1 > 1 + 3\varepsilon.
\end{aligned}$$

Set P can be embedded in time $\tilde{O}(dn^{1+1/\Omega(c)})$, and any query $q \in \ell_1^d$ can be embedded in time $O(dk)$.

Proof. Let f, g be the mappings defined in the beginning of the section and $D_0 = \Theta(\log(k/\varepsilon))$. Assume wlog for simplicity that $q = 0^d$. Then, by Lemma 9 for $k = \Theta\left(\log^2\lambda_P\cdot\log(c/\varepsilon)\right)$, with probability at least $1-\varepsilon/5$, we have:

$$\forall p \in P:\ \|p-q\|_1 \ge D_0 + \varepsilon \implies \|h(p) - f(q)\|_1 \ge 4.$$

By Theorem 6, for $\gamma = \varepsilon/10$ and $\delta = \varepsilon/(5\lambda_P^{8\log(cD_0/\varepsilon)})$, with probability at least $1-\varepsilon/5$, we get:

$$\forall p \in P:\ \|p-q\|_1 \in (1+9\varepsilon, D_0+\varepsilon) \implies \|h(p) - f(q)\|_1 > (1+8\varepsilon)(1-\varepsilon) \ge 1 + 3\varepsilon.$$

Moreover,

$$\mathbf{Pr}\left[\,\|h(p^*) - f(q)\|_1 \le 1 + 3\varepsilon\right] \ge 1 - \frac{1+\varepsilon/10}{1+\varepsilon} \ge 1 - (1-\varepsilon/2).$$

Then, the target dimension needs to satisfy the following inequality:

$$k \geq \frac{\left(\ln\left(5\lambda_P^{8\log(cD_0/\varepsilon)}/\varepsilon\right)\right)^{2/\varepsilon}}{\zeta(\varepsilon)} = \frac{\left(\Theta(\log\log k \cdot \log\lambda_P + \log\lambda_P \cdot \ln(c/\varepsilon))\right)^{2/\varepsilon}}{\zeta(\varepsilon)}.$$

Hence, for $k = (\log\lambda_P \cdot \log(c/\varepsilon))^{\Theta(1/\varepsilon)}/\zeta(\varepsilon)$, we achieve a total probability of success in $\Omega(\varepsilon)$, which completes the proof. ◀

5 Dimension reduction based on randomly shifted grids

In this section, we explore some properties of randomly shifted grids, and we present a simplified embedding which consists of a first step of snapping points to a grid, and a second step of randomly projecting grid points.

Let $w > 0$ and t be chosen uniformly at random from the interval $[0, w]$. The function

$$h_{w,t}(x) = \left\lfloor \frac{x-t}{w} \right\rfloor$$

induces a random partition of the real line into segments of length w. Hence, the function

$$g_w(x) = (h_{w,t_1}(x_1), ..., h_{w,t_d}(x_d)),$$

for $t_1, \ldots, t_d$ independent uniform random variables in the interval $[0, w]$, induces a randomly shifted grid in $\mathbb{R}^d$. For a set $X \subseteq \mathbb{R}^d$, we denote by $g_w(X)$, the image of X on the randomly shifted grid points defined by g_w. For some $x \in \mathbb{R}^d$ and $r > 0$, the number of grid cells of $g_w(\ell_1^d)$ that $B_1(x, r)$ intersects per axis is independent, and in expectation is $1+2r/w$. Then, the expected total number of grid cells that $B_1(x, r)$ intersects is at most $(1+2r/w)^d$.

Now let $P \subset \ell_1^d$ be a set of n points with doubling constant λ_P and $q \in \ell_1^d$ a query point. For $w = \varepsilon/d$, the ℓ_1-diameter of each cell is ε and therefore $g_w(P)$ is an ε-covering set of P.

▶ **Lemma 11.** *Let $\mathcal{R} > 1$ and $P' := B_1(q, \mathcal{R}) \cap P$. Then, for $w = \varepsilon/d$*

$$\mathbb{E}\left[|g_w(P')|\right] \leq 8\lambda_P^{2\log(d\mathcal{R}/\varepsilon)}.$$

Proof. By the doubling constant definition, there exists a set of balls of radius ε/d^2 centered at points in P', of cardinality at most $\lambda_P^{2\log(d\mathcal{R}/\varepsilon)}$ which covers P'. For each ball of radius ε/d^2, the expected number of intersecting grid cells is $(1+2/d)^d \leq \mathrm{e}^2$. The lemma follows by linearity of expectation. ◀

The next lemma shows that, with constant probability, the growth on the number of representatives, as we move away from q, is bounded.

▶ **Lemma 12.** *Let $\{D_i\}_{i\in\mathbb{N}}$ be a sequence of radii such that, for any i, $D_{i+1} = 4D_i$. Let A_i be the points of $g_w(P)$ within distance $D_{i+1} = 2^{2(i+1)}D_0$ from q. Then, with probability at least $1/3$,*

$$\forall i \in \{-1, 0, \ldots\} : |A_i| \leq 4^{i+3}\lambda_P^{2\log(dD_{i+1}/\varepsilon)}.$$

Proof. By Lemma 11, $\mathbb{E}[|A_i|] \leq 8\lambda_P^{2\log(dD_{i+1}/\varepsilon)}$ for every $i \in \{-1, 0, \ldots\}$. Then, a union bound followed by Markov's inequality yields

$$\mathbf{Pr}\left[\exists i \in \{0, 1, \ldots\} : |A_i| \geq 4^{i+1}\,\mathbb{E}[|A_i|]\right] \leq 1/3.$$

In addition,

$$\mathbf{Pr}\left[|A_{-1}| \geq 4\,\mathbb{E}[|A_i|]\right] \leq 1/4.$$ ◀

▶ **Theorem 13.** *Let $P \subset \ell_1^d$ such that $|P| = n$. For any $\varepsilon \in (0, 1/2)$, there is a non-linear randomized embedding $h' : \ell_1^d \to \ell_1^k$, where $k = (\log \lambda_P \cdot \log(d/\varepsilon))^{\Theta(1/\varepsilon)} / \zeta(\varepsilon)$, for a function $\zeta(\varepsilon) > 0$ depending only on ε, such that for any $q \in \ell_1^d$, if there exists $p^* \in P$ such that $\|p^* - q\|_1 \leq 1$, then with probability $\Omega(\varepsilon)$,*

$$\|h'(p^*) - f(q)\|_1 \leq 1 + 3\varepsilon,$$
$$\forall p \in P: \ \|p - q\|_1 > 1 + 9\varepsilon \implies \|h'(p) - f(q)\|_1 > 1 + 3\varepsilon.$$

Any point can be embedded in time $O(dk)$.

Proof. We follow the same reasoning as in the proof of Theorem 10. The embedding is $h' = f \circ g_{\varepsilon/d}$, where f is the randomized linear map defined in Section 4. As before, we apply h' to every point in P, and only f to queries. The randomly shifted grid incurs an additive error of ε in the distances between q and P.

Assume wlog that $q = 0^d$ and let A_i be the points of $g_{\varepsilon/d}(P)$ within distance $D_{i+1} = 2^{2(i+1)} D_0$ from q. Hence, by Lemma 12,

$$\begin{aligned}\mathbf{Pr}\left[\exists i \exists s \in A_i : \|f(s)\|_1 \leq \frac{4\|s\|_1}{D_i}\right] &\leq \sum_{i=0}^{\infty} |A_i| \, \mathbf{Pr}\left[S \leq \frac{4T}{D_i}\right] \\ &\leq \sum_{i=0}^{\infty} 4^{i+3} \lambda_P^{2\log(dD_{i+1}/\varepsilon)} \, \mathbf{Pr}\left[\tilde{S} \leq \frac{\sqrt{4kT}}{\sqrt{D_i}}\right].\end{aligned}$$

As in Lemma 9, for $D_0 = \lceil 800T/k \rceil = \Theta(\log(k/\varepsilon))$, $k \geq 20 \log \lambda_P \cdot \log\left(\frac{dD_0}{\varepsilon\delta}\right)$ and $\delta = \varepsilon/5$,

$$\sum_{i=0}^{\infty} 4^{i+3} \lambda_P^{2\log(dD_{i+1}/\varepsilon)} \, \mathbf{Pr}\left[\tilde{S} \leq \frac{\sqrt{4kT}}{\sqrt{D_i}}\right] \leq \sum_{i=0}^{\infty} \frac{2^{2i+6+2\log\lambda_P[\log(dD_0/\varepsilon)+2(i+1)]}}{2^{k(i+1)}} \leq \varepsilon/5.$$

Hence, for $k = \Omega\left((\log^2 \lambda_P \cdot \log(d/\varepsilon)\right)$, with probability at least $1 - \varepsilon/5$, we have:

$$\forall p \in P: \ \|p - q\|_1 \geq D_0 + \varepsilon \implies \|h'(p) - f(q)\|_1 \geq 4.$$

Now, we are able to use Theorem 6 for points which are at distance at most $D_0 + \varepsilon$ from q, and the near neighbor. By Lemma 12, with constant probability, the number of grid points at distance $\leq D_0 + \varepsilon$, is at most $32 \cdot \lambda_P^{4\log(dD_0/\varepsilon)}$. Hence, by Theorem 6, for $\gamma = \varepsilon/10$ and $\delta = \varepsilon/(160\lambda_P^{4\log(dD_0/\varepsilon)})$, with probability at least $1 - \varepsilon/5$, it holds:

$$\forall p \in P: \ \|p - q\|_1 \in (1 + 9\varepsilon, D_0 + \varepsilon) \implies \|h'(p) - f(q)\|_1 > 1 + 3\varepsilon.$$

Moreover, with probability at least $\varepsilon/2$, we obtain:

$$\|h'(p^*) - f(q)\|_1 \leq 1 + 3\varepsilon.$$

As in Theorem 10, the target dimension needs to satisfy the following:

$$k \geq \frac{\left(\ln\left(160\lambda_P^{4\log(dD_0/\varepsilon)}/\varepsilon\right)\right)^{2/\varepsilon}}{\zeta(\varepsilon)}.$$

Hence, for $k = (\log \lambda_P \cdot \log(d/\varepsilon))^{\Theta(1/\varepsilon)} / \zeta(\varepsilon)$ we achieve total probability of success $\Omega(\varepsilon)$. ◀

6 Conclusion

We have filled in a gap in the spectrum of randomized embeddings with bounded distortion only for distances between the query and a pointset: such embeddings existed for ℓ_2 and ℓ_1 and for doubling subsets of ℓ_2. Here we settle the case of doubling subsets of ℓ_1 with a *near* neighbor-preserving embedding. In the meantime, we obtain concentration bounds on sums of independent Cauchy variables. Our algorithms are quite simple, therefore they should also be of practical interest.

We rely on approximate r-nets or randomly shifted grids. For the former, Theorem 10 provides with a trade-off between the preprocessing time required and the target dimension. On the other hand, Theorem 13 has the advantage of fast preprocessing: any point is embedded in $O(dk)$ time, and the embedding is oblivious to the pointset. In regards to the near-linear preprocessing time, the two results are comparable, since the dimension in Theorem 13 can be substituted by the target dimension of Theorem 6.

Notice that any potential improvements to Theorem 6 should lead to improvements to Theorems 10 and 13. The target dimension in these theorems follows from a direct application of Theorem 6 to the representative data points which lie inside a bounding ball centered at the query.

References

1 D. Achlioptas. Database-friendly Random Projections: Johnson-Lindenstrauss with Binary Coins. *J. Comput. Syst. Sci.*, 66(4):671–687, 2003.

2 N. Ailon and B. Chazelle. The Fast Johnson-Lindenstrauss Transform and Approximate Nearest Neighbors. *SIAM J. Comput.*, 39(1):302–322, May 2009.

3 E. Anagnostopoulos, I. Z. Emiris, and I. Psarros. Randomized Embeddings with Slack and High-Dimensional Approximate Nearest Neighbor. *ACM Trans. Algorithms*, 14(2):18:1–18:21, 2018. doi:10.1145/3178540.

4 A. Andoni and P. Indyk. Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. *Commun. ACM*, 51(1):117–122, 2008.

5 A. Andoni, T. Laarhoven, I. P. Razenshteyn, and E. Waingarten. Optimal Hashing-based Time-Space Trade-offs for Approximate Near Neighbors. In *Proc. ACM-SIAM Symposium on Discrete Algorithms, SODA, Barcelona, Spain*, pages 47–66, 2017.

6 A. Andoni, H. L. Nguyen, A. Nikolov, I. P. Razenshteyn, and E. Waingarten. Approximate near neighbors for general symmetric norms. In *Proc. ACM Symposium on Theory of Computing, STOC, Montreal, Canada*, pages 902–913, 2017.

7 Y. Bartal and L. A. Gottlieb. Dimension Reduction Techniques for $\ell_p, (1 < p < 2)$, with Applications. In *32nd International Symposium on Computational Geometry, SoCG 2016, June 14-18, 2016, Boston, MA, USA*, pages 16:1–16:15, 2016. doi:10.4230/LIPIcs.SoCG.2016.16.

8 Y. Bartal and L. A. Gottlieb. Approximate Nearest Neighbor Search for ℓ_p-Spaces $(2 < p < \infty)$ via Embeddings. In *Proc. LATIN: Theoretical Informatics - 13th Latin American Symp., Buenos Aires, Argentina*, pages 120–133, 2018. doi:10.1007/978-3-319-77404-6_10.

9 R. Cole and L. A. Gottlieb. Searching Dynamic Point Sets in Spaces with Bounded Doubling Dimension. In *Proc. ACM Symp. Theory of Computing*, pages 574–583, New York, USA, 2006. ACM.

10 D. Eppstein, S. Har-Peled, and A. Sidiropoulos. Approximate Greedy Clustering and Distance Selection for Graph Metrics. *CoRR*, abs/1507.01555, 2015. arXiv:1507.01555.

11 S. Har-Peled, P. Indyk, and R. Motwani. Approximate Nearest Neighbor: Towards Removing the Curse of Dimensionality. *Theory of Computing*, 8(1):321–350, 2012. doi:10.4086/toc.2012.v008a014.

12 S. Har-Peled and M. Mendel. Fast Construction of Nets in Low Dimensional Metrics, and Their Applications. In *Proc. Symp. Computational Geometry*, pages 150–158, 2005.

13 P. Indyk. On Approximate Nearest Neighbors under l_infinity Norm. *J. Comput. Syst. Sci.*, 63(4):627–638, 2001.

14 P. Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006. doi:10.1145/1147954.1147955.

15 P. Indyk and R. Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proc. ACM Symp. Theory of Computing*, pages 604–613, 1998.

16 P. Indyk and A. Naor. Nearest-neighbor-preserving Embeddings. *ACM Trans. Algorithms*, 3(3), 2007.

17 R. Krauthgamer and J. R. Lee. Navigating Nets: Simple Algorithms for Proximity Search. In *Proc. 15th Annual ACM-SIAM Symp. Discrete Algorithms*, SODA'04, pages 798–807, 2004.

18 J.R. Lee, M. Mendel, and A. Naor. Metric structures in L_1: dimension, snowflakes, and average distortion. *Eur. J. Comb.*, 26(8):1180–1190, 2005. doi:10.1016/j.ejc.2004.07.002.

Improved Strong Spatial Mixing for Colorings on Trees

Charilaos Efthymiou
Department of Computer Science, University of Warwick, UK
charilaos.efthymiou@warwick.ac.uk

Andreas Galanis
Department of Computer Science, University of Oxford, UK
andreas.galanis@cs.ox.ac.uk

Thomas P. Hayes
Department of Computer Science, University of New Mexico, Albuquerque, NM, USA
hayes@cs.unm.edu

Daniel Štefankovič
Department of Computer Science, University of Rochester, NY, USA
stefanko@cs.rochester.edu

Eric Vigoda
School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA
ericvigoda@gmail.com

Abstract

Strong spatial mixing (SSM) is a form of correlation decay that has played an essential role in the design of approximate counting algorithms for spin systems. A notable example is the algorithm of Weitz (2006) for the hard-core model on weighted independent sets. We study SSM for the q-colorings problem on the infinite $(d+1)$-regular tree. Weak spatial mixing (WSM) captures whether the influence of the leaves on the root vanishes as the height of the tree grows. Jonasson (2002) established WSM when $q > d + 1$. In contrast, in SSM, we first fix a coloring on a subset of internal vertices, and we again ask if the influence of the leaves on the root is vanishing. It was known that SSM holds on the $(d + 1)$-regular tree when $q > \alpha d$ where $\alpha \approx 1.763...$ is a constant that has arisen in a variety of results concerning random colorings. Here we improve on this bound by showing SSM for $q > 1.59d$. Our proof establishes an L^2 contraction for the BP operator. For the contraction we bound the norm of the BP Jacobian by exploiting combinatorial properties of the coloring of the tree.

2012 ACM Subject Classification Mathematics of computing → Discrete mathematics; Theory of computation → Random walks and Markov chains

Keywords and phrases colorings, regular tree, spatial mixing, phase transitions

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.48

Category RANDOM

Funding *Charilaos Efthymiou*: Supported by the Centre of Discrete Mathematics and its Applications (DIMAP), University of Warwick, EPSRC award EP/D063191/1.
Andreas Galanis: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828. The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.
Thomas P. Hayes: Partially supported by NSF CAREER award CCF-1150281.
Daniel Štefankovič: Research supported in part by NSF grant CCF-1563757.
Eric Vigoda: Research supported in part by NSF grants CCF-1617306 and CCF-1563838.



© Charilaos Efthymiou, Andreas Galanis, Thomas P. Hayes, Daniel Štefankovič, and Eric Vigoda; licensed under Creative Commons License CC-BY



1 Introduction

Consider random q-colorings of the complete tree T_h of height h with branching factor d. Does the influence of the leaves on the root decay to zero in the limit as the height grows? If so, this corresponds to weak spatial mixing, which we will define more precisely momentarily.

Now suppose we fix the coloring τ for a subset of internal vertices. Is it still the case that the influence of the leaves on the root decay to zero as the height grows? One might intuitively expect that these internal "agreements" defined by τ only help in the sense that the influence of the leaves decrease, however this problem is much more challenging; it corresponds to strong spatial mixing, which is the focus of this paper.

For statistical physics models, the key algorithmic problems are the counting problem of estimating the partition function and the problem of sampling from the Gibbs distribution, which corresponds to the equilibrium state of the system. Strong spatial mixing (SSM) is a key property of the system for the design of efficient counting/sampling algorithms.

SSM has a variety of algorithmic implications. A direct consequence of SSM on amenable graphs, such as the integer lattice $\mathbb{Z}^d$, is fast mixing of the Glauber dynamics, which is the simple Markov chain that updates the spin at a randomly chosen vertex in each step, see, e.g. [21, 22, 6, 9, 14, 4, 3]. SSM also plays a critical role in the efficiency of correlation-decay techniques of Weitz [26] which yields an FPTAS for the partition function of the hard-core model in the tree uniqueness region; this approach has been extended to 2-spin antiferromagnetic models [18] and other interesting examples, e.g., [19]; note, the approach of Barvinok [1] utilizing a zero-free region of the partition function in the complex plane has recently been extended to the same range of parameters for the hard-core model [23, 25].

The fundamental question in statistical physics is the uniqueness/non-uniqueness phase transition which corresponds to whether long-range correlations persist or die off, in the limit as the volume of the system tends to infinity. In the uniqueness region the correlations die off, which corresponds to *weak spatial mixing* (WSM). While WSM (or equivalently uniqueness) is a notoriously challenging problem on the 2-dimensional integer lattice $\mathbb{Z}^2$ (e.g., see the recent breakthrough work of Beffara and Duminil-Copin [2] for the ferromagnetic Potts model), the corresponding WSM problem on the infinite $(d+1)$-regular tree $\mathbb{T}_d$, known as the Bethe lattice, is typically simpler since it can be analyzed using recursions due to the absence of cycles (e.g., see Kelly [17] for the hard-core model). However, for the colorings problem, which is the focus of this paper, even WSM is far from trivial on the regular tree [16]. In fact, for the closely related antiferromagnetic Potts model the precise range of parameters for WSM is only known for fixed values of q, d [11].

The focus of this paper is on these correlation decay properties on the infinite $(d+1)$-regular tree $\mathbb{T}_d$ for the *colorings* problem. We give an informal definition of WSM and SSM, and refer the interested reader to Section 2 for formal definitions.

Let T_h denote the complete tree of height h where all internal vertices have degree $d+1$. For integer $q \geq 3$, let μ_h denote the uniform distribution over proper (vertex) q-colorings of T_h. Consider a pair of sequences of colorings (η_h) and (η'_h) for the leaves of T_h. Let p_h and p'_h denote the marginal probability that the root receives a specific color c under μ_h conditional on the leaves having the fixed coloring η_h and η'_h, respectively. Roughly, if $\lim_{h\to\infty} |p_h - p'_h| = 0$ for all sequences $(\eta_h), (\eta'_h)$ and colors c, then we say WSM holds (see also Section 2). Jonasson [16] proved that WSM holds when $q \geq d+2$. When $q \leq d+1$, the pair of boundary conditions can actually "freeze" the color at the root; moreover, Brightwell and Winkler [5] showed that there are multiple semi-translation invariant Gibbs measures on $\mathbb{T}_d$ when $q \leq d$.

Now consider an arbitrary coloring τ for a subset $S \subset \mathbb{T}_d$. Let r_h and r'_h denote the marginal probability that the root receives color c under μ_h conditional on $\eta_h \cup \tau$ and $\eta'_h \cup \tau$, respectively. If these limits are the same then we say SSM holds. The challenge of establishing SSM is illustrated by the fact that if WSM holds then we know that $\lim_{h\to\infty} p_h = 1/q$ but that is not necessarily the case in the SSM setting.

Ge and Štefankovič [13] proved that SSM holds on $\mathbb{T}_d$ when $q > \alpha d$ where $\alpha \approx 1.763...$ is the root of $\frac{1}{\alpha}\exp(1/\alpha) = 1$. Gamarnik, Katz, and Misra [12] extended this result to arbitrary triangle-free graphs of maximum degree d, under the same condition on q. Recent work of Liu, Sinclair, and Srivistava [20] builds upon [12] together with the approximate counting approach of [1, 23] to obtain an FPTAS for counting colorings of triangle-free graphs when $q > \alpha d$. Prior to these works, Goldberg, Martin, and Paterson [14] established the above form of SSM on triangle-free amenable[1] graphs, also when $q > \alpha d$. In addition to the above results, the threshold $\alpha \approx 1.76\ldots$ has arisen in numerous rapid mixing results, e.g., [7, 15, 8].

Our main result presents the first substantial improvement on the 1.76... threshold of [13], we establish SSM on the tree when $q > 1.59d$. We state a somewhat informal version of our main theorem here, the formal version will be given once we define more precisely SSM, cf. Theorem 3 below.

▶ **Theorem 1** (Informal version of Theorem 3). *There exists an absolute constant $\beta > 0$ such that, for all positive integers q, d satisfying $q \geq 1.59d + \beta$, the q-coloring model exhibits strong spatial mixing on the regular tree $\mathbb{T}_d$.*

We remark that the constant 1.59 in Theorem 1 can be replaced with any $\alpha' > 1$ satisfying

$$\frac{1}{\alpha'}\exp\Big(\frac{1}{\alpha'}\Big)\exp\Big(-\frac{1}{\alpha'-1+\exp\left(\frac{1}{\alpha'-1}\right)}\Big) < 1,$$

the smallest such value up to four decimal digits is 1.5897.

We give an overview of our proof approach in Section 3 after formally defining SSM in Section 2 and stating the formal version of Theorem 1. We then present detailed proofs of the three main lemmas in Section 3.

2 Definitions

Let $q \geq 3$ be an integer and $G = (V, E)$ be a graph. A proper q-coloring of G is an assignment $\sigma : V \to [q]$ such that for every $(u, v) \in E$ it holds that $\sigma(u) \neq \sigma(v)$. We use Ω_G to denote the set of all proper q-colorings of G and μ_G to denote the uniform probability distribution on Ω_G (provided that Ω_G is non-empty).

For $\sigma \in \Omega_G$ and a set $\Lambda \subset V$, we use σ_Λ to denote the restriction of σ to Λ. When Λ consists of a single vertex v, we will often use the shorthand σ_v to denote the color of v under σ. We say that an assignment $\eta : \Lambda \to [q]$ is *extendible* if there exists a coloring $\sigma \in \Omega_G$ such that $\sigma_\Lambda = \eta$.

We can now formally define SSM.

▶ **Definition 2.** *Let $\zeta : \mathbb{Z}_{\geq 0} \to [0, 1]$ be a real-valued function on the positive integers.*

The q-coloring model exhibits strong spatial mixing*, denoted* SSM*, on a finite graph $G = (V, E)$ with decay rate $\zeta(\cdot)$ iff for every $v \in V$, for every $\Lambda \subset V$, for any two extendible*

[1] Roughly, a graph is amenable if for every subset S of vertices, the neighborhood satisfies $|N(S)| \leq \text{poly}(|S|)$.

assignments $\eta, \eta' : \Lambda \to [q]$ *and any color* $c \in [q]$ *it holds that*

$$\left|\mu_G(\sigma_v = c \mid \sigma_\Lambda = \eta) - \mu_G(\sigma_v = c \mid \sigma_\Lambda = \eta')\right| \leq \zeta\big(\mathrm{dist}(v, \Delta)\big), \tag{1}$$

where $\Delta \subseteq \Lambda$ *denotes the set of vertices where* η *and* η' *disagree.*

In the case where G *is infinite, we say that the* q*-coloring model exhibits strong spatial mixing on* G *with decay rate* $\zeta(\cdot)$ *if it exhibits strong spatial mixing on every finite subgraph of* G *with decay rate* $\zeta(\cdot)$*.*

The definition of weak spatial mixing has one modification: in the RHS of (1) we replace $\mathrm{dist}(v, \Delta)$ by the weaker condition $\mathrm{dist}(v, \Lambda)$. WSM says that the influence of a pair of boundary conditions decays at rate $\zeta(\cdot)$ in the distance to the boundary Λ. In SSM the pair of boundaries η, η' might only differ on a subset $\Delta \subset \Lambda$; do these fixed "agreements" on $\Lambda \setminus \Delta$ influence the marginal at v? If SSM holds then the difference in the marginal at v decays at rate $\zeta(\cdot)$ in the distance to the "disagreements" in η, η'.

With these definitions in place, we are now ready to give the formal version of Theorem 1.

▶ **Theorem 3.** *There exists an absolute constant* $\beta > 0$ *such that, for all positive integers* q, d *satisfying* $q \geq 1.59d + \beta$*, the* q*-coloring model exhibits strong spatial mixing on the regular tree* $\mathbb{T}_d$ *with exponentially decaying rate.*

That is, there exist constants $\alpha, C > 0$ *and a function* ζ *satisfying* $\zeta(\ell) \leq C \exp(-\alpha\ell)$ *for all integers* $\ell \geq 0$ *such that for all finite subtrees* T *of* $\mathbb{T}_d$ *the* q*-coloring model exhibits strong spatial mixing on* T *with decay rate* ζ*.*

3 Proof Approach

For a set $\Lambda \subset V$ and an extendible assignment $\eta : \Lambda \to [q]$, we use $\boldsymbol{\pi}_{G,v,\eta}$ to denote the q-dimensional probability vector whose entries give the marginal distribution of colors at v under the boundary condition η, i.e., for a color $c \in [q]$, the c-th entry of $\boldsymbol{\pi}_{G,v,\eta}$ is given by $\mu_G(\sigma_v = c \mid \sigma_\Lambda = \eta)$.

The key ingredient to prove Theorem 3 is the following.

▶ **Theorem 4.** *There exist absolute constants* $\beta > 0$ *and* $U \in (0, 1)$ *such that the following holds for all positive integers* q, d *satisfying* $q \geq 1.59d + \beta$*.*

Let $T = \hat{\mathbb{T}}_{d,h,\rho}$ *be the* d*-ary tree with height* h *rooted at* ρ*,* Λ *be a subset of the vertices of* T*, and* $\eta, \eta' : \Lambda \to [q]$ *be two extendible assignments of* T *with* $\mathrm{dist}(\rho, \Delta) \geq 3$ *where* $\Delta \subseteq \Lambda$ *is the set of vertices where* η *and* η' *disagree. Let* $v_1, \ldots, v_d$ *be the children of* ρ *and for* $i \in [d]$ *let* $T_i = (V_i, E_i)$ *be the subtree of* T *rooted at* v_i *which consists of all descendants of* v_i *in* T*. Then*

$$\left\|\boldsymbol{\pi} - \boldsymbol{\pi}'\right\|_2^2 \leq U \max_{i \in [d]} \left\|\boldsymbol{\pi}_i - \boldsymbol{\pi}_{i'}\right\|_2^2,$$

where $\boldsymbol{\pi} = \boldsymbol{\pi}_{T,\rho,\eta}$*,* $\boldsymbol{\pi}' = \boldsymbol{\pi}_{T,\rho,\eta'}$ *and for* $i \in [d]$ *we denote* $\boldsymbol{\pi}_i = \boldsymbol{\pi}_{T_i,v_i,\eta(\Lambda \cap V_i)}$*,* $\boldsymbol{\pi}'_i = \boldsymbol{\pi}_{T_i,v_i,\eta'(\Lambda \cap V_i)}$*.*

Intuitively, Theorem 4 says that disagreements between η and η' have smaller impact on the marginals as we move upwards on the tree. More precisely, the marginals of the root under η and under η' are closer in L^2 distance than the distance between the marginals of any child (under the induced distributions on the subtrees hanging from them).

Using Theorem 4, the proof of Theorem 3 of strong spatial mixing follows from rather standard considerations, the proof can be found in Section 7. In the following section, we focus on the more interesting proof of Theorem 4 and explain the new aspects of our analysis.

3.1 The three main lemmas

In this section, we lay down the main technical steps in proving Theorem 4. In particular, we will assume throughout that, for appropriate integers q, d, h, $T = \hat{\mathbb{T}}_{d,h,\rho}$ is the d-ary tree with height h rooted at ρ, Λ is a subset of the vertices of T, and $\eta, \eta' : \Lambda \to [q]$ are two extendible assignments of T with $\text{dist}(\rho, \Delta) \geq 3$ where $\Delta \subseteq \Lambda$ is the set of vertices where η and η' disagree. We als let $v_1, \ldots, v_d$ be the children of ρ and for $i \in [d]$ let $T_i = (V_i, E_i)$ be the subtree of T rooted at v_i which consists of all descendants of v_i in T.

To prove Theorem 4, we will use tree recursions to express the marginal at the root in terms of the marginals at the children (as in previous works on WSM/SSM, see, e.g., [5, 13, 11]). This recursion is the well-known *Belief Propagation* (BP) equation [24]; our proof of Theorem 4 will be based on bounding appropriately the gradient of the BP equations. The new ingredient in our analysis is that we incorporate the combinatorial structure of agreements close to the root into a refined L^2 analysis of the gradient.

Prior to delving into the analysis, we first describe the BP equation for the colorings model. Following the notation of Theorem 4, let $\boldsymbol{\pi} = \boldsymbol{\pi}_{T,\rho,\eta}$, $\boldsymbol{\pi}' = \boldsymbol{\pi}_{T,\rho,\eta'}$ be the marginal distributions at the root of the tree T under the boundary conditions η and η', respectively. Similarly, for $i \in [d]$, let $\boldsymbol{\pi}_i$, $\boldsymbol{\pi}'_i$ be the marginals at the root v_i of the subtree T_i under $\eta(\Lambda \cap V_i)$ and $\eta'(\Lambda \cap V_i)$, respectively.

We can now relate the distribution $\boldsymbol{\pi}$ with the distributions $\{\boldsymbol{\pi}_i\}_{i\in[d]}$ (and similarly, $\boldsymbol{\pi}'$ with the distributions $\{\boldsymbol{\pi}'_i\}_{i\in[d]}$) as follows. For q-dimensional probability vectors $\mathbf{x}_1, \ldots, \mathbf{x}_d$ and a color $c \in [q]$, let f_c be the function

$$f_c(\mathbf{x}_1, \ldots, \mathbf{x}_d) = \frac{\prod_{i\in[d]} (1 - x_{i,c})}{\sum_{j\in[q]} \prod_{i\in[d]} (1 - x_{i,j})}, \tag{2}$$

where, for $i \in [d]$ and $j \in [q]$, $x_{i,j}$ denotes the j-th entry of the vector $\mathbf{x}_i$. Then, with π_c and π'_c denoting the c-th entries of $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$, we have that

$$\begin{aligned} \pi_c &= \mu_T(\sigma_\rho = c \mid \sigma_\Lambda = \eta) = f_c(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_d), \\ \pi'_c &= \mu_T(\sigma_\rho = c \mid \sigma_\Lambda = \eta') = f_c(\boldsymbol{\pi}'_1, \ldots, \boldsymbol{\pi}'_d). \end{aligned} \tag{3}$$

The functions $\{f_c\}_{c\in[q]}$ correspond to the BP equations for the coloring model.

We are now ready to describe in more detail our SSM analysis. Specifically, to get a bound on the norm $\|\boldsymbol{\pi} - \boldsymbol{\pi}'\|_2$, we will study the gradient of f_c as we change the arguments $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_d)$ to $(\boldsymbol{\pi}'_1, \ldots, \boldsymbol{\pi}'_d)$ along the line connecting them. Our gradient analysis will take account of the following combinatorial notions.

▶ **Definition 5.** *A vertex v of T is called* frozen *under η if $v \in \Lambda$ and* non-frozen *otherwise. For a non-frozen vertex v of T, a color k is* blocked *for v (under η) if there is a neighbor $u \in \Lambda$ of v such that $\eta(u) = k$; the color is called* available *for v otherwise.*

▶ **Observation 6.** *In the setting of Theorem 4, we have that the disagreements between η and η' occur at distance at least 3 from the root. It follows that the set of the root's children that are frozen as well as the set of blocked colors for each of the non-frozen children are identical under both η and η'.*

We will utilize that the gradient components that correspond to either frozen children or blocked colors can be disregarded since, by Observation 6, the corresponding arguments in (3) are fixed to the same value. Namely, we will track, for each color c, the fraction of non-frozen children which have color c available. This will allow us in the upcoming Lemma 10 to aggregate accurately the gradient components corresponding to color c. The following definitions setup some relevant notation.

▶ **Definition 7.** *Let D be the indices of the children of the root which are non-frozen under η and η'. For a color $c \in [q]$, let $\gamma_c \in [0,1]$ be the* fraction *of indices $i \in D$ such that color c is available for v_i under η and η' (cf. Observation 6). Let $\boldsymbol{\gamma}$ and $\sqrt{\boldsymbol{\gamma}}$ be the q-dimensional vector with entries $\{\gamma_c\}_{c\in[q]}$ and $\{\sqrt{\gamma_c}\}_{c\in[q]}$, respectively.*

Intuitively, if γ_c is close to 0, color c is blocked at a lot of the children and hence the distance $\|\boldsymbol{\pi} - \boldsymbol{\pi}'\|_2$ at the root should not depend a lot on the color c (since most components of the gradient corresponding to color c are zero).

The following couple of definitions will be relevant for capturing more precisely the gradient of the functions $\{f_c\}_{c\in[q]}$. To begin with, the gradient will actually turn out to be related to the value of f_c as we move along the line $(\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_d)$ to $(\boldsymbol{\pi}'_1, \dots, \boldsymbol{\pi}'_d)$. More precisely, we have the following definition.

▶ **Definition 8.** *For $t \in [0,1]$, let $\hat{\boldsymbol{\pi}}(t) = \{\hat{\pi}_c(t)\}_{c\in[q]}$ be the q-dimensional probability vector whose c-th entry is given by $f_c\big(t\boldsymbol{\pi}_1 + (1-t)\boldsymbol{\pi}'_1, \dots, t\boldsymbol{\pi}_d + (1-t)\boldsymbol{\pi}'_d\big)$.*

Note that $\hat{\boldsymbol{\pi}}(1) = \boldsymbol{\pi}$ and $\hat{\boldsymbol{\pi}}(0) = \boldsymbol{\pi}'$; in this sense, we can think of the vector $\hat{\boldsymbol{\pi}}(t)$ as having the marginals at the root as we interpolate between $(\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_d)$ to $(\boldsymbol{\pi}'_1, \dots, \boldsymbol{\pi}'_d)$.

The next definition will be relevant for bounding the L^2 norm of the gradient along the line connecting to $(\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_d)$ to $(\boldsymbol{\pi}'_1, \dots, \boldsymbol{\pi}'_d)$. The bound will be in terms of the "marginals" at the root, as captured by the vector $\hat{\boldsymbol{\pi}}(t)$ (cf. Definition 8), and the availability of the q colors at the children, as captured by the vector $\boldsymbol{\gamma}$ (cf. Definition 7). In particular, we will be interested in the L^2 norm of the following matrix, which is an idealized version to the Jacobian of the BP equation (see (13) for the precise formula).[2]

▶ **Definition 9.** *Let $\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\gamma}}$ be q-dimensional vectors with non-negative entries. The matrix $\mathbf{M}_{\hat{\boldsymbol{\pi}},\hat{\boldsymbol{\gamma}}}$ corresponding to the vectors $\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\gamma}}$ is given by $\big(\mathrm{diag}(\hat{\boldsymbol{\pi}}) - \hat{\boldsymbol{\pi}}\hat{\boldsymbol{\pi}}^\intercal\big)\mathrm{diag}(\hat{\boldsymbol{\gamma}})$.*[3]

Our first main lemma shows how to bound the distance between the marginals at the root under η and η', i.e., $\big\|\boldsymbol{\pi} - \boldsymbol{\pi}'\big\|_2^2$, in terms of the aggregate distance at the children. The new ingredient in our bound is to account more carefully for the availability of the colors at the children (i.e., the vector $\boldsymbol{\gamma}$).

▶ **Lemma 10.** *Let q, d be positive integers so that $q \geq d+2$. Then*

$$\big\|\boldsymbol{\pi} - \boldsymbol{\pi}'\big\|_2^2 \leq |D|K^2 \sum_{i\in[d]} \big\|\boldsymbol{\pi}_i - \boldsymbol{\pi}_{i'}\big\|_2^2 \qquad \text{where } K := \tfrac{1}{1-\frac{1}{q-d}} \max_{t\in(0,1)} \big\|\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}\big\|_2,$$

where $D, \boldsymbol{\gamma}, \sqrt{\boldsymbol{\gamma}}$ are as in Definition 7, $\hat{\boldsymbol{\pi}}(t)$ is as in Definition 8, and $\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}$ is as in Definition 9.

Given Lemma 10, we are left with obtaining a good upper bound on the norm $\big\|\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}\big\|_2$ that takes advantage of the presence of the vector $\boldsymbol{\gamma}$. It is not hard to see that the L^2 norm of the matrix $\big(\mathrm{diag}(\hat{\boldsymbol{\pi}}) - \hat{\boldsymbol{\pi}}\hat{\boldsymbol{\pi}}^\intercal\big)$ is bounded by $\max_{j\in[q]} \hat{\pi}_j$. The following result can be seen as a generalisation of this fact, which is however significantly more involved to prove. The proof is given in Section 4.

---

[2] For a square matrix $\mathbf{M}$, we use $\|\mathbf{M}\|_2$ to denote its L^2 norm, i.e., $\|\mathbf{M}\|_2 = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{M}\mathbf{x}\|_2$. A fact that will be useful later is that $\|\mathbf{M}\|_2 = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{x}^\intercal\mathbf{M}\|_2$, even for non-symmetric matrices $\mathbf{M}$.

[3] For a vector $\mathbf{x}$, $\mathrm{diag}(\mathbf{x})$ denotes the diagonal matrix with the entries of $\mathbf{x}$ on the diagonal.

▶ **Lemma 11.** *Let q be a positive integer, $\hat{\boldsymbol{\pi}}$ be a q-dimensional probability vector and $\hat{\boldsymbol{\gamma}}$ be a q-dimensional vector with non-negative entries which are all bounded by 1. Then, the L^2 norm of the matrix $\mathbf{M}_{\hat{\boldsymbol{\pi}},\hat{\boldsymbol{\gamma}}} = \big(\mathrm{diag}(\hat{\boldsymbol{\pi}}) - \hat{\boldsymbol{\pi}}\hat{\boldsymbol{\pi}}^{\intercal}\big)\mathrm{diag}(\hat{\boldsymbol{\gamma}})$ satisfies*

$$\|\mathbf{M}_{\hat{\boldsymbol{\pi}},\hat{\boldsymbol{\gamma}}}\|_2 \leq \frac{1}{2}\max_{j\in[q]} \hat{\pi}_j\big(1+(\hat{\gamma}_j)^2\big),$$

where $\{\hat{\pi}_j\}_{j\in[q]}, \{\hat{\gamma}_j\}_{j\in[q]}$ are the entries of $\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\gamma}}$, respectively.

The final component of our proof is to utilize the bound in Lemma 11 to derive an upper bound on the norm of the matrix $\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}$ appearing in Lemma 10. To prove Theorem 4, we roughly need to show that the norm is bounded by $1/|D|$. We show that this is indeed the case in Section 6.

▶ **Lemma 12.** *There exist absolute constants $\beta > 0$ and $K' \in (0,1)$ such that the following holds for all positive integers q, d satisfying $q \geq 1.59d + \beta$.*

Let $\boldsymbol{\gamma}, \hat{\boldsymbol{\pi}}(t)$ be the q-dimensional vectors of Definitions 7 and 8, respectively. Then, for all $t \in [0,1]$ and all colors $k \in [q]$, it holds that

$$\frac{1}{2}\hat{\pi}_k(t)(1+\gamma_k) < K'/|D|,$$

where D is the set of non-frozen children of ρ under η and η'.

Assuming Lemmas 10, 11 and 12 for now, we next conclude the proof of Theorem 4.

Proof of Theorem 4. Let $U' := (1+K')/2$ where $K' \in (0,1)$ is the constant in Lemma 12. Let $\beta > 0$ be a sufficiently large constant so that, for all $q \geq 1.59d + \beta$, the conclusion of Lemma 12 applies and $\frac{1}{1-\frac{1}{q-d)}}K' < U'$. We will show that

$$\big\|\boldsymbol{\pi} - \boldsymbol{\pi}'\big\|_2^2 \leq U \max_{i\in[d]} \big\|\boldsymbol{\pi}_i - \boldsymbol{\pi}_{i'}\big\|_2^2, \text{ with } U := (U')^2. \tag{4}$$

Indeed, by Lemmas 10, 11 and 12, we have that

$$\big\|\boldsymbol{\pi} - \boldsymbol{\pi}'\big\|_2^2 \leq \frac{U}{|D|}\sum_{i\in[d]} \big\|\boldsymbol{\pi}_i - \boldsymbol{\pi}'_i\big\|_2^2.$$

Note that an index $i \notin D$ corresponds to a frozen child v_i and therefore $\boldsymbol{\pi}_i = \boldsymbol{\pi}'_i$ for all $i \notin D$ and hence

$$\frac{1}{|D|}\sum_{i\in[d]} \big\|\boldsymbol{\pi}_i - \boldsymbol{\pi}'_i\big\|_2^2 \leq \max_{i\in[d]} \big\|\boldsymbol{\pi}_i - \boldsymbol{\pi}_{i'}\big\|_2^2,$$

proving (4). This completes the proof of Theorem 4. ◀

4 Bound on the matrix norm: proof of Lemma 11

In this section, we prove Lemma 11.

Proof of Lemma 11. For this proof, it will be convenient to simplify notation and use $\boldsymbol{\pi}$ instead of $\hat{\boldsymbol{\pi}}$ and $\boldsymbol{\gamma}$ instead of $\hat{\boldsymbol{\gamma}}$, so that $\mathbf{M}_{\boldsymbol{\pi},\boldsymbol{\gamma}}$ becomes $\big(\mathrm{diag}(\boldsymbol{\pi}) - \boldsymbol{\pi}\boldsymbol{\pi}^{\intercal}\big)\mathrm{diag}(\boldsymbol{\gamma})$. Let $C := \frac{1}{2}\max_{j\in[q]} \pi_j(1+\gamma_j^2)$. We will establish that $\|\mathbf{M}_{\boldsymbol{\pi},\boldsymbol{\gamma}}\|_2 \leq C$ by showing that for an arbitrary q-dimensional vector $\mathbf{x}$ it holds that

$$\|\mathbf{x}^{\intercal}\mathbf{M}_{\boldsymbol{\pi},\boldsymbol{\gamma}}\|_2^2 \leq C^2\,\|\mathbf{x}\|_2^2\,. \tag{5}$$

We will focus on proving (5) in the case where the entries of the vector $\boldsymbol{\gamma}$ are all nonnegative and strictly less than one; the case where some of the entries of $\boldsymbol{\gamma}$ are equal to 1 follows from the continuity of (5) with respect to $\boldsymbol{\gamma}$.

So, assume that $\gamma_j \in [0,1)$ for all $j \in [q]$. Observe that

$$\|\mathbf{x}^\intercal \mathbf{M}_{\boldsymbol{\pi},\boldsymbol{\gamma}}\|_2^2 = \sum_{j\in[q]} \pi_j^2\gamma_j^2(x_j - w)^2 \text{ where } w := \sum_{j\in[q]} \pi_j x_j.$$

Let $y_j = x_j - w$ for $j \in [q]$. Since $\boldsymbol{\pi}$ is a probability vector, we have

$$\sum_{j\in[q]} \pi_j y_j = 0.$$

Moreover, we can rewrite (5) as

$$\sum_{j\in[q]} \frac{\pi_j^2\gamma_j^2}{C^2} y_j^2 \le \sum_{j\in[q]} (y_j + w)^2. \tag{6}$$

Note that the function $f(z) = \sum_{j\in[q]}(y_j + z)^2$ achieves its minimum for $z^* = -\frac{1}{q}\sum_{j\in[q]} y_j$ and $f(z^*) = \sum_{j\in[q]} y_j^2 - \frac{1}{q}\big(\sum_{j\in[q]} y_j\big)^2$. Hence, to prove (6) (and therefore (5)), it suffices to show that

$$\Big(\sum_{j\in[q]} y_j\Big)^2 \le q \sum_{j\in[q]} \frac{y_j^2}{A_j}, \text{ where } A_j := \frac{C^2}{C^2 - \pi_j^2\gamma_j^2} \tag{7}$$

Note that the A_j's are well-defined and greater than 1 for all $j \in [q]$ by our assumption that $\gamma_j \in [0,1)$, cf. the argument below (5). Using that $\sum_{j\in[q]} \pi_j y_j = 0$, we therefore obtain that (7) is equivalent to

$$\Big(\sum_{j\in[q]} y_j(1 + t\pi_j)\Big)^2 \le q \sum_{j\in[q]} \frac{y_j^2}{A_j}, \text{ where } A_j := \frac{C^2}{C^2 - \pi_j^2\gamma_j^2}, \tag{8}$$

for any real number t – we will specify t soon (cf. the upcoming (10)). In particular, by the Cauchy-Schwarz inequality, we have

$$\Big(\sum_{j\in[q]} y_j(1 + t\pi_j)\Big)^2 \le \sum_{j\in[q]} \frac{y_j^2}{A_j} \sum_{j\in[q]} A_j(1 + t\pi_j)^2,$$

so (8) and hence (7) will follow if we find t such that

$$\sum_{j\in[q]} A_j(1 + t\pi_j)^2 \le q. \tag{9}$$

We will choose t to minimise the l.h.s. in (9), i.e., set

$$t := -\frac{\sum_{j\in[q]} A_j\pi_j}{\sum_{j\in[q]} A_j\pi_j^2}, \quad \text{so that} \quad \sum_{j\in[q]} A_j(1 + t\pi_j)^2 = \sum_{j\in[q]} A_j - \frac{\big(\sum_{j\in[q]} A_j\pi_j\big)^2}{\sum_{j\in[q]} A_j\pi_j^2}. \tag{10}$$

Therefore, for this choice of t, (9) becomes

$$\sum_{j\in[q]} (A_j - 1) \sum_{j\in[q]} A_j\pi_j^2 \le \Big(\sum_{j\in[q]} A_j\pi_j\Big)^2. \tag{11}$$

Using that $A_j = \frac{C^2}{C^2-\pi_j^2\gamma_j^2}$, (11) is equivalent to (note the division by C^2 of both sides)

$$\sum_{j\in[q]} \frac{\pi_j^2\gamma_j^2}{C^2-\pi_j^2\gamma_j^2} \sum_{j\in[q]} \frac{\pi_j^2}{C^2-\pi_j^2\gamma_j^2} \leq \Big(\sum_{j\in[q]} \frac{C\pi_j}{C^2-\pi_j^2\gamma_j^2}\Big)^2. \tag{12}$$

We next establish (12). We can upper bound the l.h.s. of (12) using the inequality $ab \leq \left(\frac{a+b}{2}\right)^2$, which gives that

$$\sum_{j\in[q]} \frac{\pi_j^2\gamma_j^2}{C^2-\pi_j^2\gamma_j^2} \sum_{j\in[q]} \frac{\pi_j^2}{C^2-\pi_j^2\gamma_j^2} \leq \Big(\sum_{j\in[q]} \frac{\pi_j^2(1+\gamma_j^2)}{2(C^2-\pi_j^2\gamma_j^2)}\Big)^2.$$

So, to prove (12), it suffices to show that for each $i \in [q]$, it holds that

$$\frac{\pi_j^2(1+\gamma_j^2)}{2(C^2-\pi_j^2\gamma_j^2)} \leq \frac{C\pi_j}{C^2-\pi_j^2\gamma_j^2}$$

which is indeed true, since $C \geq \frac{1}{2}\pi_j(1+\gamma_j^2)$ for all $i \in [q]$ by the definition of C.

This proves (12), which in turn establishes (8) for the choice of t in (10). This yields (7) and hence (5) as well, finishing the proof of Lemma 11. ◀

5 Gradient analysis with blocked colors: proof of Lemma 10

In this section, we prove Lemma 10.

Proof of Lemma 10. For $i \in [d]$ and $j \in [q]$, let $F_{c,j}^{(i)}(\mathbf{x})$ be the partial derivative $\frac{\partial f_c}{\partial x_{i,j}}$ viewed as a function of the "concatenated" vector $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_d)$. Note that, whenever $x_{i,j} \neq 1$, we have that

$$\begin{aligned} F_{c,j}^{(i)}(\mathbf{x}) &= -\frac{f_c(\mathbf{x}_1,\ldots,\mathbf{x}_d) - \big(f_c(\mathbf{x}_1,\ldots,\mathbf{x}_d)\big)^2}{1-x_{i,j}} \text{ if } j = c, \\ F_{c,j}^{(i)}(\mathbf{x}) &= \frac{f_c(\mathbf{x}_1,\ldots,\mathbf{x}_d) f_j(\mathbf{x}_1,\ldots,\mathbf{x}_d)}{1-x_{i,j}} \text{ if } j \neq c. \end{aligned} \tag{13}$$

As mentioned earlier, we will interpolate between $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ by interpolating along the straight-line segment connecting $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_d)$ and $(\boldsymbol{\pi}'_1, \ldots, \boldsymbol{\pi}'_d)$. In particular, for $t \in [0,1]$, let $\hat{\pi}_c(t)$ denote the c-th entry of the vector $\hat{\boldsymbol{\pi}}(t)$ defined in the statement of the lemma. Then, we have that

$$\hat{\pi}_c(t) = f_c(\mathbf{z}(t)), \text{ where } \mathbf{z}(t) \text{ is the vector } \big(t\boldsymbol{\pi}_1 + (1-t)\boldsymbol{\pi}'_1, \ldots, t\boldsymbol{\pi}_d + (1-t)\boldsymbol{\pi}'_d\big). \tag{14}$$

We will use $z_{i,j}(t)$ to denote the j-th entry of the i-th vector in $\mathbf{z}(t)$, i.e., $z_{i,j}(t) = t\pi_{i,j} + (1-t)\pi'_{i,j}$.

Let D be the set of indices i such that v_i is not frozen under η and η' (cf. Observation 6). Observe that, for all $i \notin D$ and $c, j \in [q]$, we have that $z_{i,j}(t) = \pi_{i,j} = \pi'_{i,j}$ for $t \in [0,1]$. Moreover, for $i \in D$ and $j \in [q]$ we have that $\pi_{i,j}, \pi'_{i,j} \leq 1/(q-d)$ (since the child v_i has at least $q-d$ available colors in the subtree T_i) and hence

$$0 \leq z_{i,j}(t) \leq 1/(q-d). \tag{15}$$

Since $z_{i,j}(t) \neq 1$ for $i \in D$ and $j \in [q]$, it follows that

$$\frac{d\hat{\pi}_c}{dt} = \sum_{i\in D} \sum_{j=1}^{q} F_{c,j}^{(i)}(\mathbf{z}(t))(\pi_{i,j} - \pi'_{i,j}).$$

Using (3), we therefore have that

$$(\pi_c - \pi'_c)^2 = \big(\hat{\pi}_c(1) - \hat{\pi}_c(0)\big)^2 = \Big(\int_0^1 \frac{d\hat{\pi}_c}{dt} dt\Big)^2 = \Big(\int_0^1 \sum_{i\in D}\sum_{j=1}^{q} F^{(i)}_{c,j}(\mathbf{z}(t))(\pi_{i,j} - \pi'_{i,j})dt\Big)^2$$
$$\leq \int_0^1 \Big(\sum_{i\in D}\sum_{j=1}^{q} F^{(i)}_{c,j}(\mathbf{z}(t))(\pi_{i,j} - \pi'_{i,j})\Big)^2 dt,$$

where the last inequality follows by applying the Cauchy-Schwarz inequality for integrals. By summing over all colors $c \in [q]$, we obtain

$$\big\|\boldsymbol{\pi} - \boldsymbol{\pi}'\big\|_2^2 \leq \int_0^1 \sum_{c=1}^{q} \Big(\sum_{i\in D}\sum_{j=1}^{q} F^{(i)}_{c,j}(\mathbf{z}(t))(\pi_{i,j} - \pi'_{i,j})\Big)^2 dt. \tag{16}$$

To simplify the r.h.s. of (16), we first note that, by (13) and (14), we have

$$F^{(i)}_{c,j}(\mathbf{z}(t)) = \frac{A_{c,j}(t)}{1 - z_{i,j}(t)} \text{ where } A_{c,j} := \begin{cases} (\hat{\pi}_c(t))^2 - \hat{\pi}_c(t), & \text{if } j = c, \\ \hat{\pi}_c(t)\hat{\pi}_j(t), & \text{if } j \neq c \end{cases} \tag{17}$$

Moreover, for $j \in [q]$, set

$$u_j(t) = \frac{1}{|D|\gamma_j} \sum_{i\in D} \frac{\pi_{i,j} - \pi'_{i,j}}{1 - z_{i,j}(t)} \text{ if } \gamma_j > 0, \text{ else set } u_j(t) = 0. \tag{18}$$

Note that if color j is blocked for the child v_i we have that $\pi_{i,j} - \pi'_{i,j} = 0$, so using the power mean inequality we have that

$$\gamma_j (u_j(t))^2 \leq \frac{1}{|D|} \sum_{i\in D} \Big(\frac{\pi_{i,j} - \pi'_{i,j}}{1 - z_{i,j}}\Big)^2. \tag{19}$$

Then, for $c \in [q]$, we have that

$$\sum_{i\in D}\sum_{j=1}^{q} F^{(i)}_{c,j}(\mathbf{z}(t))(\pi_{i,j} - \pi'_{i,j}) = \sum_{j=1}^{q} A_{c,j}(t) \sum_{i\in D} \frac{\pi_{i,j} - \pi'_{i,j}}{1 - z_{i,j}(t)} = |D| \sum_{j=1}^{q} A_{c,j}(t)\gamma_j u_j(t), \tag{20}$$

where the last equality follows from (18) and observing that if $\gamma_j = 0$ then $\pi_{i,j} - \pi'_{i,j} = 0$ for all $i \in D$. Note that the (c,q)-entry of $\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}$ is exactly $-A_{c,j}(t)\sqrt{\gamma_j}$ (cf. (17) and Definition 9) and hence, using (20), we can write the integrand in the r.h.s. of (16) as

$$\sum_{c=1}^{q} \Big(\sum_{i\in D}\sum_{j=1}^{q} F^{(i)}_{c,j}(\mathbf{z}(t))(\pi_{i,j} - \pi'_{i,j})\Big)^2 = |D|^2 \, \big\|\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}\mathbf{u}(t)\big\|_2^2, \tag{21}$$

where, for $t \in [0,1]$, $\mathbf{u}(t)$ is the q-dimensional vector with entries $\{\sqrt{\gamma_j}\, u_j(t)\}_{j\in[q]}$. Let

$$W := \max_{t\in[0,1]} \big\|\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}\big\|_2, \text{ so that } K = \frac{W}{1 - \frac{1}{q-d}}.$$

Then, for $t \in [0,1]$, we have that

$$\begin{aligned} \big\|\mathbf{M}_{\hat{\boldsymbol{\pi}}(t),\sqrt{\boldsymbol{\gamma}}}\mathbf{u}(t)\big\|_2^2 &\leq W^2 \, \|\mathbf{u}(t)\|_2^2 = W^2 \sum_{j\in[q]} \gamma_j (u_j(t))^2 \leq \frac{W^2}{|D|} \sum_{j\in[q]} \sum_{i\in D} \Big\|\frac{\pi_{i,j} - \pi'_{i,j}}{1 - z_{i,j}(t)}\Big\|_2^2 \\ &\leq \frac{K^2}{|D|} \sum_{j\in[q]} \sum_{i\in D} \big\|\pi_{i,j} - \pi'_{i,j}\big\|_2^2 = \frac{K^2}{|D|} \sum_{i\in[d]} \|\boldsymbol{\pi}_i - \boldsymbol{\pi}'_i\|_2^2, \end{aligned} \tag{22}$$

where the first inequality is by definition of the norm, the second inequality follows from (19), the third inequality follows from $0 \leq z_{i,j}(t) \leq 1/(q-d)$, and the last equality follows from the fact that for $i \notin D$ we have that $\boldsymbol{\pi}_i = \boldsymbol{\pi}'_i$. Combining (16), (21) and (22), we obtain that

$$\left\|\boldsymbol{\pi} - \boldsymbol{\pi}'\right\|_2^2 \leq |D|K^2 \sum_{i\in[d]} \|\boldsymbol{\pi}_i - \boldsymbol{\pi}'_i\|_2^2 .$$

This finishes the proof of Lemma 10. ◀

6 Bounds on the marginals: proof of Lemma 12

In this section, we prove Lemma 12. We begin with the following lemma.

▶ **Lemma 13.** *Let q, d, h be positive integers so that $q \geq d+1$ and $h \geq 1$. Let $T = \mathbb{T}_{d,h,\rho}$ be the d-ary tree with height h rooted at ρ, Λ be a subset of the vertices of T such that $\rho \notin \Lambda$, and $\eta : \Lambda \to [q]$ be an extendible assignment of T. Then, for all colors $k \in [q]$ that are available for ρ under η, it holds that*

$$\mu_T(\sigma_\rho = k \mid \sigma_\Lambda = \eta) \geq \frac{\left(1 - \frac{1}{q-d}\right)^d}{d + (q-d)\left(1 - \frac{1}{q-d}\right)^d}.$$

Proof. Let $Q \subseteq [q]$ be the set of all colors that are available for ρ under η and let $k \in Q$. Let $v_1, \ldots, v_d$ be the children of ρ in T and let $D = \{i \in [d] \mid v_i \notin \Lambda\}$ be the indices of the children of ρ that do not belong to Λ.

For $i \in [d]$, let $T_i = (V_i, E_i)$ be the subtree of T rooted at v_i which consists of all descendants of v_i in T (together with v_i itself). Further, for a color $j \in [q]$, let

$$x_{i,j} = \mu_{T_i}\left(\sigma_{v_i} = j \mid \sigma_{\Lambda\cap V_i} = \eta_{\Lambda\cap V_i}\right),$$

i.e., $x_{i,j}$ is the marginal probability that v_i takes the color j at v_i in μ_{T_i} with boundary condition $\eta_{\Lambda\cap V_i}$. Note that

$$0 \leq x_{i,j} \leq \frac{1}{q-d} \text{ for all } i \in D \text{ and } j \in [q], \qquad \sum_{j\in[q]} x_{i,j} = 1 \text{ for all } j \in [q]. \tag{23}$$

Using the tree recursion (2) and ignoring summands that are 0 or factors that are equal to 1, the marginal $\mu_T(\sigma_\rho = k \mid \sigma_\Lambda = \eta)$ is expressed in terms of $x_{i,j}$ as follows:

$$\mu_T(\sigma_\rho = k \mid \sigma_\Lambda = \eta) = \frac{\prod_{i\in D}(1 - x_{i,k})}{\sum_{j\in Q}\prod_{i\in D}(1 - x_{i,j})}. \tag{24}$$

We prove the lemma by deriving an appropriate lower bound on the quantity at the r.h.s. of (24) subject to the constraint in (23). For the numerator in (24), we have that

$$\prod_{i\in D}(1 - x_{i,k}) \geq \left(1 - \frac{1}{q-d}\right)^{|D|}. \tag{25}$$

For the denominator we are going to show the following:

$$\sum_{j\in Q}\prod_{i\in D}(1 - x_{i,j}) \leq d + (q-d)\left(1 - \frac{1}{q-d}\right)^{|D|}. \tag{26}$$

Before showing that (26) is indeed true, note that the lemma follows by plugging (25), (26) into (24), yielding

$$\mu_T(\sigma_\rho = k \mid \sigma_\Lambda = \eta) \geq \frac{\left(1 - \frac{1}{q-d}\right)^{|D|}}{d + (q-d)\left(1 - \frac{1}{q-d}\right)^{|D|}} \geq \frac{\left(1 - \frac{1}{q-d}\right)^{d}}{d + (q-d)\left(1 - \frac{1}{q-d}\right)^{d}},$$

where the last inequality follows by noting that the ratio in the middle is decreasing in $|D|$ and $|D| \leq d$.

We now proceed with the proof of (26). First, we have the simple bound

$$\sum_{j \in Q} \prod_{i \in D} (1 - x_{i,j}) \leq \sum_{j \in [q]} \prod_{i \in D} (1 - x_{i,j}). \tag{27}$$

For $j \in [q]$, let $x_j = \frac{1}{|D|} \sum_{i \in D} x_{i,j}$ and note that $(x_1, \ldots, x_q)$ is a probability vector whose entries are in $[0, 1/(q-d)]$. By the AM-GM inequality, we can bound the r.h.s. of (27) by

$$\sum_{j \in [q]} \prod_{i \in D} (1 - x_{i,j}) \leq \sum_{j \in [q]} (1 - x_j)^{|D|}. \tag{28}$$

It remains to observe that the function $f(\mathbf{z}) = \sum_{j \in [q]} (1 - z_j)^{|D|}$ is convex over the space of probability vectors $\mathbf{z} = (z_1, \ldots, z_q)$ whose entries are in $[0, 1/(q-d)]$, and hence f attains its maximum at the extreme points of the space, which are given by (the permutations of) the probability vector whose first d entries are equal to zero and the rest are equal to $1/(q-d)$. It follows that

$$\sum_{j \in [q]} (1 - x_j)^{|D|} \leq d + (q-d)\Big(1 - \frac{1}{q-d}\Big)^{|D|}. \tag{29}$$

Combining (27), (28) and (29) yields (26), thus concluding the proof of Lemma 13. ◀

We are now ready to prove Lemma 12.

Proof of Lemma 12. For convenience, let $r = 1.59$, so that $q/d \geq r$. We will use that r satisfies

$$C := \frac{1}{r} \exp\left(\frac{1}{r}\right) \exp\Big(-\frac{1}{r - 1 + \exp\left(\frac{1}{r-1}\right)}\Big) < 1. \tag{30}$$

We will show the result with the constant $K' = (1 + C)/2$. For the rest of this proof, we will focus on the case $q \in [1.59d + \beta, 2.01d]$, for some large constant $\beta > 0$ (when $q > 2.01d$ the desired bound follows rather crudely, see Footnote 4 below for details).

Recall that $v_1, \ldots, v_d$ are the children of ρ in T and D is the set of (indices of the) non-frozen children of the root ρ. Let $Q \subseteq [q]$ be the set of all colors that are available for ρ under η; since at most $d - |D|$ colors can be blocked for ρ, we have that

$$|Q| \geq q - (d - |D|). \tag{31}$$

For $i \in [d]$, let $T_i = (V_i, E_i)$ be the subtree of T rooted at v_i which consists of all descendants of v_i in T (together with v_i itself). Further, for a color $j \in [q]$, recall that

$$\begin{aligned} \pi_{i,j} &= \mu_{T_i}\big(\sigma_{v_i} = j \mid \sigma_{\Lambda \cap V_i} = \eta_{\Lambda \cap V_i}\big), \\ \pi'_{i,j} &= \mu_{T_i}\big(\sigma_{v_i} = j \mid \sigma_{\Lambda \cap V_i} = \eta'_{\Lambda \cap V_i}\big), \end{aligned} \tag{32}$$

i.e., $\pi_{i,j}$ is the marginal probability that v_i takes the color j at v_i in μ_{T_i} with boundary condition $\eta_{\Lambda\cap V_i}$. For a non-frozen child v_i (i.e., $i \in D$), note that, if color j is available for v_i (in T_i), then we have from Lemma 13 the bounds

$$L \leq \pi_{i,j}, \pi_{i,j}, \text{ where } L = \frac{\left(1-\frac{1}{q-d}\right)^d}{d+(q-d)\left(1-\frac{1}{q-d}\right)^d}. \tag{33}$$

Another useful bound to observe for later is that

$$dL < 1/3 \text{ for all } d \geq 2.$$

Consider arbitrary $k \in Q$. For $t \in [0,1]$, let $\mathbf{z}(t)$ be the vector $\left(t\boldsymbol{\pi}_1 + (1-t)\boldsymbol{\pi}'_1, \ldots, t\boldsymbol{\pi}_d + (1-t)\boldsymbol{\pi}'_d\right)$. Using the tree recursion (2) and ignoring summands that are 0 or factors that are equal to 1, we obtain

$$\hat{\pi}_k(t) = \frac{\prod_{i\in D}(1-z_{i,k}(t))}{\sum_{j\in Q}\prod_{i\in D}(1-z_{i,j}(t))}. \tag{34}$$

Recall, our goal is to show that $\frac{1}{2}\hat{\pi}_k(t)(1+\gamma_k) < K'/|D|$ for all $t \in [0,1]$, where $\gamma_k \in [0,1]$ is the fraction of non-frozen children that have color k available. [4]Note that, if color j is available for the child v_i, (33) gives that

$$L \leq z_{i,j}(t) \text{ for } t \in [0,1],$$

so, using the fact that the color k is available for $|D|\gamma_k$ non-frozen children, we obtain that the numerator of (34) is bounded by

$$\prod_{i\in D}\left(1-z_{i,k}(t)\right) \leq (1-L)^{|D|\gamma_k} \leq \exp(-L|D|\gamma_k), \tag{35}$$

whereas the denominator, using the AM-GM inequality analogously to [7, Lemma 2.1], by

$$\sum_{j\in Q}\prod_{i\in D}\left(1-z_{i,j}(t)\right) \geq q\exp(-|D|/q) - (d-|D|). \tag{36}$$

From (34), (35), and (36), it follows that $\hat{\pi}_k(t) \leq \frac{\exp(-L|D|\gamma_k)}{q\exp(-|D|/q)-(d-|D|)}$. Therefore, the lemma will follow by showing that

$$\frac{|D|\exp(-|D|L\gamma_k)}{q\exp(-|D|/q)-(d-|D|)}(1+\gamma_k) < 2K'. \tag{37}$$

Note that the function $h(x) = (1+x)\exp(-dLx)$ is increasing when $x \in [0,1]$, since

$$h'(x) = \exp(-dLx)\left(1-dL(1+x)\right) \geq \exp(-dLx)(1-2dL) > 0.$$

Therefore, to prove (37), it suffices to show that

$$\frac{|D|\exp(-|D|L)}{q\exp(-|D|/q)-(d-|D|)} < K', \text{ or equivalently that } f(|D|) > 0 \tag{38}$$

[4] For $q > 2.01d$, we have from (34) and (31) that $\hat{\pi}_k(t) \leq \frac{1}{|Q|-|D|} \leq \frac{1}{q-d} < \frac{1}{1.01d} \leq K'/|D|$, yielding the desired inequality.

where $f(x) := K'\big(q\exp(-x/q) - d + x\big) - x\exp(-Lx)$ for $x \in [0, d]$. We claim that $f(x)$ is decreasing in x. We have

$$f'(x) = K' - K'\exp(-x/q) - \exp(-Lx)(1 - Lx)$$

which is maximised for $x = d$. In particular,

$$\begin{aligned} f'(x) &\le f'(d) = K' - K'\exp(-d/q) - \exp(-dL)(1 - dL) \\ &\le K' - K'\exp(-1/r) - \exp(-1/3)(1 - 1/3) \le 0, \end{aligned}$$

where the second to last inequality follows from the fact that $dL < 1/3$ and the last inequality using that $K' < 1$. For $|D| = d$, (38) becomes

$$\frac{d\exp(-dL)}{q\exp(-d/q)} < K'. \tag{39}$$

Now, we have that

$$dL \ge \frac{1}{r - 1 + \exp\left(\frac{d}{(r-1)d-1}\right)}.$$

Therefore, by choosing β large enough, we can ensure that

$$\frac{d\exp(-dL)}{q\exp(-d/q)} < \frac{1 + C}{2} = K',$$

where C is the constant in (30). This proves (39) and therefore concludes the proof of Lemma 12. ◀

7 Proof of Theorem 3

Finally, utilizing Theorem 4, we give the proof of Theorem 3.

Proof of Theorem 3. From Theorem 4, we know that there exist constants $\beta > 0$ and $U \in (0, 1)$ such that for all $q \ge 1.59d + \beta$ the conclusion of Theorem 4 applies. Note that Theorem 4 applies to the d-ary tree rather than the $(d+1)$-regular tree but these trees differ only at the degree of the root. To account for it, we will assume that $q \ge 1.59(d+1) + \beta$, i.e., prove Theorem 3 with constant $\beta' = \beta + 1.59$. Consider the function ζ given by $\zeta(\ell) = 2U^{\ell-2}$ for $\ell \ge 0$ and note that ζ is exponentially decaying. We will show that the q-coloring model has strong spatial mixing on the $(d+1)$-regular tree with decay rate ζ.

We first show by induction on h that, for the tree $T = \hat{\mathbb{T}}_{d+1,h,\rho}$ (that is, the $(d+1)$-ary tree with height h rooted at ρ), for any subset Λ of vertices of T and arbitrary extendible assignments $\eta, \eta' : \Lambda \to [q]$ of T, it holds that

$$\left\|\boldsymbol{\pi}_{T,\rho,\eta} - \boldsymbol{\pi}_{T,\rho,\eta'}\right\|_2^2 \le \zeta(\mathrm{dist}(\rho, \Delta)), \tag{40}$$

where $\Delta \subseteq \Lambda$ is the set of vertices where η and η' disagree. The base cases $h = 0, 1, 2$ are trivial so assume $h \ge 3$ in what follows. Let $\ell = \mathrm{dist}(\rho, \Delta)$. Once again, (40) is trivial when $\ell \le 2$, so assume $\ell \ge 3$ in what follows. Let $v_1, \ldots, v_{d+1}$ be the children of ρ and, for $i \in [d+1]$, let $T_i = (V_i, E_i)$ be the subtree of T rooted at v_i which consists of all descendants of v_i in T. Further, let $\boldsymbol{\pi}_i = \boldsymbol{\pi}_{T_i,v_i,\eta(\Lambda\cap V_i)}$, $\boldsymbol{\pi}'_i = \boldsymbol{\pi}_{T_i,v_i,\eta'(\Lambda\cap V_i)}$. Then, by Theorem 4 and since $q \ge 1.59(d+1) + \beta$, we have that

$$\left\|\boldsymbol{\pi}_{T,\rho,\eta} - \boldsymbol{\pi}_{T,\rho,\eta'}\right\|_2^2 \le U \max_{i\in[d+1]} \left\|\boldsymbol{\pi}_i - \boldsymbol{\pi}_{i'}\right\|_2^2. \tag{41}$$

For $i \in [d+1]$, since T_i is isomorphic to $\hat{\mathbb{T}}_{d+1,h-1,\rho}$ we have by the induction hypothesis that

$$\left\|\boldsymbol{\pi}_i - \boldsymbol{\pi}_{i'}\right\|_2^2 \leq \zeta(\ell - 1).$$

Combining this with (41) and the fact that $\zeta(\ell) = U\zeta(\ell - 1)$ yields (40), completing the induction and therefore that strong spatial mixing holds on T with decay rate ζ.

Now, let $T = (V, E)$ be a finite subtree of the $(d+1)$-regular tree, v be an arbitrary vertex of T, Λ be a subset of vertices of T and $\eta, \eta' : \Lambda \to [q]$ be arbitrary extendible assignments of T. Then, we can view T as a subgraph of $T_v = \hat{\mathbb{T}}_{d+1,h,v}$ for some appropriate height h. It also holds that (see, for example, [10, Lemma 25])

$$\left\|\boldsymbol{\pi}_{T,v,\eta} - \boldsymbol{\pi}_{T,v,\eta'}\right\|_2 = \left\|\boldsymbol{\pi}_{T_v,v,\eta} - \boldsymbol{\pi}_{T_v,v,\eta'}\right\|_2.$$

Therefore, from (40) (applied to the tree T_v) we obtain that

$$\left\|\boldsymbol{\pi}_{T,v,\eta} - \boldsymbol{\pi}_{T,v,\eta'}\right\|_2^2 \leq \zeta(\mathrm{dist}(v, \Delta)),$$

where $\Delta \subseteq \Lambda$ is the set of vertices where η and η' disagree.

This completes the proof of Theorem 3. ◀

References

1. A. Barvinok. *Combinatorics and Complexity of Partition Functions*. Algorithms and Combinatorics. Springer International Publishing, 2017.
2. V. Beffara and H. Duminil-Copin. The self-dual point of the two-dimensional random-cluster model is critical for $q \geq 1$. *Probability Theory and Related Fields*, 153(3):511–542, 2012.
3. A. Blanca, P. Caputo, A. Sinclair, and E. Vigoda. Spatial Mixing and Non-local Markov Chains. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1965–1980, 2018.
4. A. Blanca and A. Sinclair. Random-cluster dynamics in $\mathbb{Z}^2$. *Probability Theory and Related Fields*, 168(3-4):821–847, 2017.
5. G. R. Brightwell and P. Winkler. Random Colorings of a Cayley Tree. In *Contemporary Combinatorics*, pages 247–276, 2002.
6. F. Cesi. Quasi-factorization of the entropy and logarithmic Sobolev inequalities for Gibbs random fields. *Probability Theory and Related Fields*, 120(4):569–584, 2001.
7. M. Dyer and A. Frieze. Randomly coloring graphs with lower bounds on girth and maximum degree. *Random Structures & Algorithms*, 23(2):167–179, 2003.
8. M. Dyer, A. Frieze, T. P. Hayes, and E. Vigoda. Randomly coloring constant degree graphs. *Random Structures & Algorithms*, 43(2):181–200, 2013.
9. M. Dyer, A. Sinclair, E. Vigoda, and D. Weitz. Mixing in time and space for lattice spin systems: A combinatorial view. *Random Structures & Algorithms*, 24(4):461–479, 2004.
10. C. Efthymiou. A Simple Algorithm for Sampling Colorings of $G(n, d/n)$ Up to The Gibbs Uniqueness Threshold. *SIAM Journal on Computing*, 45(6):2087–2116, 2016.
11. A. Galanis, L. A. Goldberg, and K. Yang. Uniqueness for the 3-state antiferromagnetic Potts model on the tree. *Electron. J. Probab.*, 23, 2018.
12. D. Gamarnik, D. Katz, and S. Misra. Strong spatial mixing of list coloring of graphs. *Random Structures & Algorithms*, 46(4):599–613, 2015.
13. Q. Ge and D. Štefankovič. Strong spatial mixing of q-colorings on Bethe lattices. *CoRR*, abs/1102.2886, 2011. `arXiv:1102.2886`.
14. L. A. Goldberg, R. Martin, and M. Paterson. Strong Spatial Mixing with Fewer Colors for Lattice Graphs. *SIAM Journal on Computing*, 35(2):486–517, 2005.
15. T. P. Hayes. Local uniformity properties for Glauber dynamics on graph colorings. *Random Structures & Algorithms*, 43(2):139–180, 2013.

16 J. Jonasson. Uniqueness of uniform random colorings of regular trees. *Statistics & Probability Letters*, 57(3):243–248, 2002.

17 F. P. Kelly. Stochastic Models of Computer Communication Systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 47(3):379–395, 1985.

18 L. Li, P. Lu, and Y. Yin. Correlation Decay Up to Uniqueness in Spin Systems. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 67–84, 2013.

19 J. Liu and P. Lu. FPTAS for #BIS with degree bounds on one side. In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 549–556, 2015.

20 J. Liu, A. Sinclair, and P. Srivastava. A deterministic algorithm for counting colorings with 2Δ colors. *CoRR*, abs/1906.01228, 2019. `arXiv:1906.01228`.

21 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. I. The attractive case. *Communications in Mathematical Physics*, 161(3):447–486, 1994.

22 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. II. The General Case. *Communications in Mathematical Physics*, 161(3):487–514, 1994.

23 V. Patel and G. Regts. Deterministic Polynomial-Time Approximation Algorithms for Partition Functions and Graph Polynomials. *SIAM Journal on Computing*, 46(6):1893–1919, 2017.

24 J. Pearl. Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach. In *Proceedings of the Second AAAI Conference on Artificial Intelligence*, AAAI'82, pages 133–136, 1982.

25 H. Peters and G. Regts. On a Conjecture of Sokal Concerning Roots of the Independence Polynomial. *The Michigan Mathematical Journal*, pages 33–55, 2019.

26 D. Weitz. Counting Independent Sets Up to the Tree Threshold. In *Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing*, STOC '06, pages 140–149, 2006.

(Near) Optimal Adaptivity Gaps for Stochastic Multi-Value Probing



Domagoj Bradac
Department of Mathematics, Faculty of Science, University of Zagreb, Croatia
domagoj.bradac@gmail.com

Sahil Singla
Department of Computer Science, Princeton University[1], NJ, USA
singla@cs.princeton.edu

Goran Zuzic
Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
gzuzic@cs.cmu.edu

Abstract

Consider a kidney-exchange application where we want to find a max-matching in a random graph. To find whether an edge e exists, we need to perform an expensive test, in which case the edge e appears independently with a *known* probability p_e. Given a budget on the total cost of the tests, our goal is to find a testing strategy that maximizes the expected maximum matching size.

The above application is an example of the stochastic probing problem. In general the optimal stochastic probing strategy is difficult to find because it is *adaptive* – decides on the next edge to probe based on the outcomes of the probed edges. An alternate approach is to show the *adaptivity gap* is small, i.e., the best *non-adaptive* strategy always has a value close to the best adaptive strategy. This allows us to focus on designing non-adaptive strategies that are much simpler. Previous works, however, have focused on Bernoulli random variables that can only capture whether an edge appears or not. In this work we introduce a multi-value stochastic probing problem, which can also model situations where the weight of an edge has a probability distribution over multiple values.

Our main technical contribution is to obtain (near) optimal bounds for the (worst-case) adaptivity gaps for multi-value stochastic probing over prefix-closed constraints. For a monotone submodular function, we show the adaptivity gap is at most 2 and provide a matching lower bound. For a weighted rank function of a k-extendible system (a generalization of intersection of k matroids), we show the adaptivity gap is between $O(k \log k)$ and k. None of these results were known even in the Bernoulli case where both our upper and lower bounds also apply, thereby resolving an open question of Gupta et al. [23].

2012 ACM Subject Classification Theory of computation → Stochastic control and optimization; Theory of computation → Submodular optimization and polymatroids; Theory of computation → Approximation algorithms analysis; Theory of computation → Design and analysis of algorithms

Keywords and phrases stochastic programming, adaptivity gaps, stochastic multi-value probing, submodular functions, k-extendible systems, adaptive strategy, matroid intersection

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.49

Category RANDOM

Related Version `https://arxiv.org/abs/1902.01461`

Funding *Domagoj Bradac*: The Author thanks the Computer Science Department at Carnegie Mellon University for their support; part of his work was done with visiting in Summer 2018.
Sahil Singla: Supported in part by NSF awards CCF-1319811, CCF-1536002, and CCF-1617790 and in part by the Schmidt foundation.
Goran Zuzic: Supported in part by NSF grants CCF-1527110, CCF-1618280, CCF-1814603, CCF-1910588, NSF CAREER award CCF-1750808 and a Sloan Research Fellowship.

Acknowledgements We would like to thank Anupam Gupta for helpful discussions.

[1] Most of this work was done when the author was a graduate student at Carnegie Mellon University.


© Domagoj Bradac, Sahil Singla, and Goran Zuzic;
licensed under Creative Commons License CC-BY

1 Introduction

Consider a kidney-exchange application where we want to find a maximum matching in a random graph. To find whether an edge e exists, we need to perform an expensive test, in which case the edge e appears independently with a known probability p_e. Given a budget on the total cost of the tests, our goal is to design a testing strategy that maximizes the expected size of the found matching.

The above application can be modeled as a constrained *stochastic probing* problem [5, 21, 3, 22, 23]. In this setting, we are given a universe V of *elements* (e.g., the set of all possible edges), each with an *activation* probability p_v for $v \in V$ (e.g., the probability an edge exists). We define a random set $A \subseteq V$ of *active* elements that contains every v independently with probability p_v. A *probe* at v reveals whether $v \in A$ or $v \notin A$, and we are only allowed to probe certain *feasible subsets* $S \in \mathcal{F} \subseteq 2^V$ (e.g., subsets of edges whose tests fit in our budget). Our goal is to design a *probing strategy* to find a feasible set $S \in \mathcal{F}$ of elements to maximize $\mathbb{E}_A[f(A \cap S)]$, where f is some combinatorial function $f : 2^V \to \mathbb{R}_{\geq 0}$ (e.g., the cardinality of the maximum matching). Notice our probing strategy could be *adaptive*, i.e., we could decide which element to probe next based on the outcomes of already probed elements.

Besides matching [13, 7], stochastic probing has applications for stochastic variants of several other combinatorial problems. E.g., it can be used for Bayesian mechanism design problems [21], robot path-planning problems [22, 23], and stochastic set cover problems that arise in database applications [27, 16]. As observed in these prior works, the optimal strategy for stochastic probing can be represented as a binary *decision tree* where each node represents an element of V: You first probe the root node element, and then depending on whether it is active or inactive, you either move to the right or the left subtree. In general, such an optimal decision tree can be exponentially sized and is hard to describe. We do not even understand how to capture it for very simple functions and constraints (e.g., the max function with cardinality constraints [24]).

An alternate approach is to focus on *non-adaptive* strategies. Such a strategy commits to probing a feasible set $S \in \mathcal{F}$ in the beginning, irrespective of which of these elements turn out active. A non-adaptive strategy has several benefits: (a) it is easy to represent since we can just store the set S, (b) it is easy to find for many classes of functions and constraints (e.g., submodular functions over intersection of matroids [12]), and (c) it is parallelizable because we do not need feedback. The concern is that the expected value of the optimal non-adaptive strategy might be much smaller than that of the optimal adaptive strategy. This raises the (worst-case instance) *adaptivity gap* question: What is the maximum ratio between the expected values of the optimal adaptive and the optimal non-adaptive strategies for stochastic probing? If this ratio is small then we can focus on non-adaptive strategies and reap its benefits with only a small loss in value (see Figure 1).

Since for general combinatorial functions or constraints the adaptivity gaps can be made arbitrarily large, we need to consider special classes of functions and constraints. In a surprising result, Gupta et al. prove that for any *monotone submodular* function and any *prefix-closed constraints*[2], the adaptivity gap is at most 3 [23]. The best known lower bound in this setting, however, is only $\frac{e}{e-1} \approx 1.58$ due to Asadpour et al. [5]. This leaves open the following question:

> *For stochastic probing, what is the (worst-case) adaptivity gap for monotone submodular functions over prefix-closed constraints?*

[2] Prefix-closed constraints stipulate that any prefix of a feasible probing sequence is also feasible. This class contains any downward-closed/packing constraint.

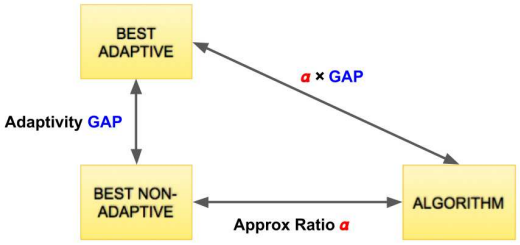


Figure 1 An α-approximation to the best non-adaptive solution implies an $(\alpha \cdot \text{GAP})$-approximation to the best adaptive algorithm, where GAP is the adaptivity gap.

We show that both the previously known upper bound of 3 and the lower bound of $\frac{e}{e-1}$ are not tight. Instead, the adaptivity gap is exactly 2.

One might notice that submodular functions do not capture the max-matching function used to model kidney-exchanges. This motivates us to consider more general combinatorial functions; in particular, we study the weighted rank function of a k-extendible system (defined in §2). This class generalizes intersection of k-matroids [29], e.g., a 2-extendible system captures matching in general graphs (unlike intersections of two matroids). Our goal is to bound the adaptivity gap for such functions over arbitrary prefix-closed constraints.

A major drawback of the stochastic probing model is that it only considers Bernoulli random variables. One would ideally allow for more modeling power by permitting the outcome of a probe to be a non-binary value. For example, in the kidney-exchange application, one might desire to summarize an edge probe by the risk involved in performing the match: a value of 0 describes an impossible match, a value of 1 indicates a safe match, and the possibilities in between are represented by intermediate values. Notice that the optimal adaptive strategy is still a decision tree; however, it may no longer be binary.

The main contributions of this paper are (1) a model that extends the binary stochastic probing to the multi-value setting, (2) the exact calculation of the adaptivity gap for stochastic probing of monotone submodular functions (in both the binary and multi-value setting), and (3) a nearly-tight adaptivity gap for stochastic probing of weighted rank functions over k-extendible systems.

1.1 Overview of Results

Our conceptual contribution is to present a generalization of the stochastic probing model to *stochastic multi-value probing* (SMP) described in §2. Roughly, the idea is that each element has t potential types, and a probe reveals which one of its types it takes. This trivially captures stochastic probing for $t = 2$, where the two types are active and inactive. In general these different types can be used to model different weights of an element, or to even encode different kinds of complementary relationships in the element values.

Although the SMP model is more general than the stochastic probing model, our main technical result in §3 is that for monotone submodular functions the adaptivity gap is bounded by 2. We also give a matching lower bound which proves this cannot be further reduced. This is despite the fact that the optimal decision tree for SMP may no longer be binary.

▶ **Theorem 1.** *The adaptivity gap for SMP where the constraints are prefix-closed and the function is monotone non-negative submodular is exactly* 2.

Since SMP is strictly more general than stochastic probing, Theorem 1 also improves the previously known upper bound of 3 for monotone submodular stochastic probing. In fact, our lower bound SMP instance in Theorem 1 is Bernoulli. Thus it resolves an open question of [23] of finding the optimal adaptivity gaps for submodular stochastic probing.

Our main technical result in §4 is that the adaptivity gap for weighted rank function of a k-extendible system is $\tilde{\Theta}(k)$.

▶ **Theorem 2.** *The adaptivity gap for SMP where the constraints are prefix-closed and the function is a weighted rank function of a k-extendible system is between k and $O(k \log k)$. Moreover, for unweighted rank functions, the adaptivity gap is between k and $2k$.*

Since the weighted rank of function of intersection of k-matroids is a k-extendible system, Theorem 2 implies as a corollary that the adaptivity gaps for this class is at most $\tilde{\Theta}(k)$. This improves the previously best known upper bound for intersection of k matroids of $O(k^4 \cdot \log n)$ due to Gupta et al. [22][3]. We also give an $\Omega(\sqrt{k})$-lower bound in this setting.

1.2 Techniques and Challenges

In this section we outline our main techniques and challenges for SMP adaptivity gaps.

Submodular Functions. To prove a small adaptivity gap, we need to show existence of a "good" non-adaptive solution. A priori it is not clear how to construct such a solution, e.g., LP based approaches do not extend beyond matroid constraints because of large integrality gaps. Since we only need to show *existence*, we can assume the optimal (exponential sized) decision tree is known. A crucial idea of [22] is to perform a random walk on this optimal decision tree (with probabilities given by the tree) and probing elements on the sampled root-leaf path. In other words, consider a non-adaptive strategy that randomly chooses a root-leaf path in the decision tree with the same probability as the optimal adaptive strategy. While this idea is natural in hindsight, its analysis for the non-adaptive strategy has been challenging.

In [22], the authors use Freedman's inequality – linear functions are "well-concentrated" for a martingale – to argue that *simple* submodular functions are well-concentrated. This step requires massive union bounds over a polynomial number of linear functions, which loses logarithmic factors. To overcome this super-constant loss, in [23] the authors use an inductive approach and induct over subtrees where in each step a *stem* – the all-no path – is observed. A "stem lemma" allows them to argue that for every stem the expected value of the non-adaptive algorithm is within a factor 2 to the expected adaptive strategy. Finally, they "stitch" back the stem for induction by using submodularity, overall losing a factor of 3.

In this work, to prove the improved adaptivity gap of 2 in Theorem 1, our insight is to modify the above induction to observe a *single node* at each step (instead of a *stem* as in [23]). While we still induct over subtrees, this allows us to avoid any additional loss due to the stitching step. This induction turns out to be nontrivial because the adaptive and non-adaptive strategies can observe different types of the root element. In other words, although the non-adaptive random walk strategy follows the distribution of root-leaf paths of the adaptive strategy, it has to independently re-sample (re-probe) all the nodes on the chosen path. This hinders a direct application of induction as the marginal values in the subtrees change between the two strategies. We remedy this issue using two main ideas. First,

[3] We remark that although not explicitly stated in Gupta et al. [23], their techniques can be used to remove the dependency on n, but it still only gives $\Omega(k^2)$ adaptivity gaps.

we compare the non-adaptive strategy to a "super-strategy" that can choose from both the elements chosen by the adaptive and the non-adaptive strategies. (This is also the intuition for the gap of 2 since the "super-strategy" has two chances to sample an element.) Second, the non-adaptive strategy forfeits any potential future value that the adaptive strategy gained at the root but the non-adaptive missed due to re-sampling. (This can be done by contracting the element sampled by the adaptive strategy without receiving its value.) Notice that both these steps are pessimistic and hence give a valid upper bound on the adaptivity gap. Together these ideas suffice to match the marginal values in the subtrees and apply induction without the stiching step, yielding an adaptivity gap of 2. Our lower bounds in §3.2 show examples where the super-strategy does not have any advantage over the adaptive strategy. Thus the adaptivity gap of 2 is optimal.

Rank Functions. A technical challenge in extending the above *inductive* approach to k-extendible system rank functions is that their marginal values do not belong to the same class. Namely, after contracting an element, the marginal value of a submodular function is submodular but the marginal value of a k-extendible system rank function may not even be subadditive. To overcome this, we first focus on *unweighted* rank functions. Instead of directly comparing the non-adaptive strategy to the adaptive strategy, our insight is to compare it to a *greedy procedure*. We show that this greedy procedure is a k-approximation to the adaptive strategy. Moreover, we show it has a notion of a marginal value. This allows us to compare the non-adaptive strategy to the greedy procedure in a similar way as for submodular functions, by losing another factor of 2. Our lower bound in §4.3 shows that the factor k loss in comparing to a greedy procedure is unavoidable, thereby making our analysis tight up to constants.

Finally, the challenge in proving Theorem 2 for *weighted* k-extendible system rank functions is that the greedy procedure only guarantees a k-approximation if we go in the order of decreasing weights. Instead, our adaptivity gap proofs only work when we are greedy in the root-to-leaf path order. One way around this is to partition the elements into $O(\log n)$ exponentially weighted *classes* (e.g., $1, 2, 2^2, \ldots$) and apply the unweighted argument to the most valuable class. Unfortunately, this loses an $\Omega(\log n)$ factor. To obtain bounds independent of the universe size n, our insight is that picking an element in a class "removes" at most k elements from a lower weight class. We can therefore improve the $\log n$ factor loss to a $\log k$ by increasing the gap between successive classes to $\Omega(k)$. To achieve this we further combine $O(\log k)$ consecutive classes into a "super-class" (bucket). It is an interesting open question to find if this $\log k$ loss is essential in going from unweighted to weighted k-extendible system rank functions.

1.3 Further Related Work

The adaptivity gap of stochastic packing problems has seen much interest; see, e.g., for knapsack [14, 10, 28], packing integer programs [15, 13, 7], budgeted multi-armed bandits [17, 19, 26, 28], and orienteering [18, 20, 8]. All except the orienteering results rely on having relaxations that capture the constraints of the problem via linear constraints. For stochastic monotone submodular functions where the probing constraints are given by matroids, Asadpour et al. [4] bounded the adaptivity gap by $\frac{e}{e-1}$; Hellerstein et al. [25] bound it by $\frac{1}{\tau}$, where τ is the smallest probability of some set being materialized. Other relevant papers are [27, 16].

The work of Chen et al. [13] (see also [1, 7, 9, 2]) sought to maximize the size of a matching subject to b-matching constraints; this was motivated by applications to online dating and kidney exchange. See also [30, 6] for pointers to other work on kidney exchange

problems. The work of [21] abstracted out the general problem of maximizing a function (in their case, the rank function of the intersection of matroids or knapsacks) subject to probing constraints (again, intersection of matroids and knapsacks). This was improved and generalized by Adamczyk et al. [3] to submodular objectives. All these results use LPs or geometric relaxations, and do not extend to arbitrary packing constraints due to large integrality gaps of the relaxations.

2 Stochastic Multi-Value Probing Model

In this section we formally define our *stochastic multi-value probing* (SMP) model using the idea of combinatorial valuation over independent elements. We also discuss some preliminaries.

2.1 Combinatorial Valuation over Independent Elements

The multi-value paradigm is based on the notion of *type*, which represents different "values" an element can take. This leads to combinatorial valuations over independent elements where each element *independently* takes its type. Similar notions have been defined before; e.g., see [31] and references therein.

▶ **Definition 3** (Combinatorial valuation $\mathrm{val}_{\mathbf{X}}$ over independent elements)**.** *Consider a finite universe V of elements and size $n = |V|$. Each element $e \in V$ obtains exactly one* type *from a finite set T_e according to a given probability distribution $\mathcal{D}_e$ over T_e. These types are assigned independently across different elements, i.e., the random vector of types $\mathbf{X} \in (T_e)_{e \in V}$ is drawn from the product distribution $\prod_{e \in V} \mathcal{D}_e$. Given a combinatorial function $f : 2^T \to \mathbb{R}_{\geq 0}$ for $T \stackrel{\text{def}}{=} \bigcup_{e \in V} T_e$, the valuation of a set $S \subseteq V$ is*

$$\mathrm{val}_{\mathbf{X}}(S) \stackrel{\text{def}}{=} f\big(\{\mathbf{X}_e \mid e \in S\}\big) = f(\mathbf{X}_S),$$

where we define $\mathbf{X}_S \stackrel{\text{def}}{=} \big\{\mathbf{X}_e \mid e \in S\big\}$ to simplify notation.

For example, in the Bernoulli case studied in the stochastic probing literature, each element has two types: active and inactive, the distributions $\mathcal{D}_e$ are Bernoulli, and the valuation function $\mathrm{val}_{\mathbf{X}}(S) = f(\{e \in S \mid e \text{ is active}\})$. Another example is the multi-value max-weight matching problem described in the introduction. Here different types of an element (edge) correspond to its different weights and $\mathrm{val}_{\mathbf{X}}(S)$ is the max-weight matching in the induced subgraph on S.

In this work we always assume the combinatorial function $f : 2^T \to \mathbb{R}_{\geq 0}$ satisfies $f(\emptyset) = 0$ and is *monotone*, i.e., $f(A) \leq f(B)$ for all $A \subseteq B$. We also assume it belongs to one of the following classes.

- *subadditive* if $f(A \cup B) \leq f(A) + f(B)$ for all $A, B \subseteq T$.
- *submodular* if $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$ for all $A, B \subseteq T$. For $S \subseteq T$, the contraction

$$f_S(A) \stackrel{\text{def}}{=} f(S \cup A) - f(S) \tag{1}$$

  of a monotone submodular function is also monotone submodular.
- *weighted rank function* of a family $\mathcal{F} \subseteq 2^T$ if $f(A) = \max_{B \in \mathcal{F}} w(A \cap B)$ where $w : 2^T \to \mathbb{R}_{\geq 0}$ is a linear function with non-negative weights. When w is the all ones vector (i.e., $w(A) = |A|$), we call it the *unweighted rank function* of $\mathcal{F}$.

In particular, we work with rank functions of two special families $\mathcal{F} \in 2^V$. Subsets in the family are called *independent* subsets. A family $\mathcal{F} \ni \emptyset$ forms a

- *matroid* if for every $A, B \in \mathcal{F}$ with $|A| > |B|$ there exists $x \in A \setminus B$ such that $B \cup \{e\} \in \mathcal{F}$.
- *k-extendible system* if for every $A \subseteq B \in \mathcal{F}$ and $e \in T$ where $A \cup \{e\} \in \mathcal{F}$, we have that there is a set $Z \subseteq B \setminus A$ such that $|Z| \leq k$ and $B \setminus Z \cup \{e\} \in \mathcal{F}$.

This latter family is important because it generalizes the family of intersection of k matroids, e.g., a 2-extendible systems captures general graph matchings (see [11] for further discussion).

2.2 Adaptive Strategies and SMP

Roughly, the goal of an SMP problem is to maximize a combinatorial function over independent elements under some "feasibility constraints". We define a *probe* of an element $e \in V$ to be an operation that reveals its random type $X_e \in T_e$. A *probing sequence* is an ordered sequence of probes on some elements.

The SMP problem only allows a family of probing sequences $\mathcal{C}$, which are called *feasible*. We assume minimal properties from this family. Specifically, it is *prefix-closed*, i.e., for every sequence in $\mathcal{C}$, each of its prefix is also in $\mathcal{C}$. This prefix-closed family is powerful because it generalizes any *downward-closed* family $\mathcal{F}$ (i.e., for all $A \in \mathcal{F}$ and $B \subseteq A$ we have $B \in \mathcal{F}$) and can also capture precedence constraints.

We now define an *adaptive strategy* which constitutes a feasible solution for SMP. The nodes in this tree correspond to probes of elements

▶ **Definition 4** (Adaptive strategy $\mathcal{T}$). *It is a rooted decision tree where each non-leaf node is labeled with an element $e \in V$ and has $|T_e|$ arcs to child nodes. Each arc is uniquely labeled with a type $t \in T_e$. Whenever we encounter a node labeled e, the adaptive strategy probes e and proceeds to the subtree corresponding to the arc labeled $X_e \sim \mathcal{D}_e$. The strategy terminates on reaching a leaf and receives a value of $\mathrm{val}_{\mathbf{X}}(S(\mathbf{X}))$, where $S(\mathbf{X}) \subseteq V$ is the set of probed elements by strategy $\mathcal{T}$ for type vector $\mathbf{X}$. The objective is the expected valuation, which we denote by*

$$\mathit{adap}(\mathcal{T}, f) \stackrel{\text{def}}{=} \mathbb{E}_{\mathbf{X}}[\mathrm{val}_{\mathbf{X}}(S(\mathbf{X}))]. \tag{2}$$

Notice, since f is monotone, a strategy never gains value by removing a probed element. We say a strategy $\mathcal{T}$ is *feasible* for $\mathcal{C}$ if every root-leaf path belongs to $\mathcal{C}$. We now formally define an SMP problem.

▶ **Definition 5** (SMP problem $(\mathcal{C}, \mathrm{val}_{\mathbf{X}})$). *Given a prefix-closed family of probing constraints $\mathcal{C}$ and a combinatorial valuation $\mathrm{val}_{\mathbf{X}}$ over independent elements, an SMP problem is to find a feasible adaptive strategy $\mathcal{T}$ to maximize the expected valuation $\mathit{adap}(\mathcal{T}, f)$.*

2.3 Non-Adaptive Strategies and Adaptivity Gaps

A strategy to solve an SMP problem can benefit from adjusting its probing sequence based on the outcomes of the already probed elements. For instance, in the kidney-exchange example if one finds an edge incident to a vertex u, one may choose not to probe any other edges incident to u. On the other hand, a strategy that always decides the next probe independent of the outcomes of the probed elements is called *non-adaptive*. Our goal is to study the largest ratio between adaptive and non-adaptive strategies.

▶ **Definition 6** (Adaptivity gap for $\mathcal{P}$). *Let $\mathcal{P}$ be a class of SMP problems (e.g., monotone submodular functions over prefix-closed constraints). Define the* adaptivity gap *as the largest (worst-case instance) ratio of the optimal adaptive and optimal non-adaptive strategies for a problem $(\mathcal{C}, \mathrm{val}_{\mathbf{X}}) \in \mathcal{P}$, i.e.,*

$$\sup_{(\mathcal{C},\mathrm{val}_{\mathbf{X}})\in\mathcal{P}} \frac{\sup_{\mathcal{T} \text{ is feasible in } P} \mathit{adap}(\mathcal{T}, f)}{\sup_{S\in\mathcal{C}} \mathbb{E}_{\mathbf{X}}[\mathrm{val}_{\mathbf{X}}(S)]}.$$

Notice that in the denominator S does not depend on $\mathbf{X}$.

The adaptivity gap for a general combinatorial function f is unbounded [22]. In this work we focus on monotone submodular functions and (weighted) rank functions of a k-extendible system. We bound adaptivity gaps by analyzing the following natural random walk non-adaptive strategy.

▶ **Definition 7** (Random walk non-adaptive strategy). *For any given adaptive strategy $\mathcal{T}$, there is a corresponding non-adaptive strategy that (virtually) draws a sample $\mathbf{X} \sim \prod_{e\in V} \mathcal{D}_e$ from the product distribution and traverses $\mathcal{T}$ along the root-leaf path for $\mathbf{X}$ (i.e., when at a node labeled e, traverse the unique arc labeled X_e). Let $S(\mathbf{X})$ be the random set of elements probed by such a root-leaf path. The true (non-virtual) types of elements correspond to the vector of outcomes $\mathbf{X}' \sim \prod_{e\in V} \mathcal{D}_e$. Here $\mathbf{X}$ and $\mathbf{X}'$ are i.i.d. r.v.s. The random walk non-adaptive strategy probes S according to the above distribution and receives the valuation*

$$\mathit{alg}(\mathcal{T}, f) \stackrel{\text{def}}{=} \mathbb{E}_{\mathbf{X},\mathbf{X}'}[\mathrm{val}_{\mathbf{X}'}(S(\mathbf{X}))]. \tag{3}$$

3 Adaptivity Gaps for a Monotone Submodular Function

In this section we prove our first main result, the optimal adaptivity gap for submodular functions. In §3.1 we prove the upper bound and in §3.2 we prove the lower bound of Theorem 1.

▶ **Theorem 1.** *The adaptivity gap for SMP where the constraints are prefix-closed and the function is monotone non-negative submodular is exactly 2.*

3.1 Upper Bound of 2

Our non-adaptive strategy samples a random root-leaf path using the optimal adaptive strategy tree $\mathcal{T}$ (Definition 7). In other words, it performs a "dry-run" of a random walk along the tree without probing anything. In the end it queries all the elements on this random root-leaf path. We argue that its expected value is at least half of the adaptive strategy. We encourage the reader to follow the proof idea outlined in §1.2 since algebra can conceal the main ideas.

Proof of the upper bound in Theorem 1. We induct over the depth of the tree $\mathcal{T}$, i.e., for any monotone submodular function f and tree $\mathcal{T}$ of depth at most d, we have

$$\mathsf{alg}(\mathcal{T}, f) \geq \frac{1}{2}\mathsf{adap}(\mathcal{T}, f).$$

The base case for $d = 1$ is trivially true because the tree is a single node. For induction, let e be the root node of the optimal decision tree $\mathcal{T}$. Denote by $I \stackrel{\text{def}}{=} X_e$ the (random) type of element e when probed by the adaptive strategy (and also the virtual type of the

non-adaptive strategy), while $R \stackrel{\text{def}}{=} X'_e$ be the (random) true type when probed by the non-adaptive strategy. Also, let $\mathcal{T}_I$ denote the subtree the adaptive strategy goes to when the root element is in type I and let f_I be the contraction from Eq. (1). This implies

$$\mathsf{adap}(\mathcal{T}, f) = \mathbb{E}_I[f(I) + \mathsf{adap}(\mathcal{T}_I, f_I)] \qquad \text{and} \qquad \mathsf{alg}(\mathcal{T}, f) = \mathbb{E}_{I,R}[f(R) + \mathsf{alg}(\mathcal{T}_I, f_R)]. \tag{4}$$

Now using submodularity and monotonicity of f on every root-leaf path of the adaptive strategy,

$$\begin{aligned}\mathsf{adap}(\mathcal{T}, f) &\le \mathbb{E}_{I,R}[f(I \cup R) + \mathsf{adap}(\mathcal{T}_I, f_{I\cup R})] \\ &\le \mathbb{E}_{I,R}[f(I) + f(R) + \mathsf{adap}(\mathcal{T}_I, f_{I\cup R})],\end{aligned}$$

where the last inequality uses that every monotone submodular function is subadditive. Notice that I and R are i.i.d. variables. This along with linearity of expectation implies

$$\mathsf{adap}(\mathcal{T}, f) \le \mathbb{E}_{I,R}[2 \cdot f(R) + \mathsf{adap}(\mathcal{T}_I, f_{I\cup R})]. \tag{5}$$

Next, we lower bound the expected value of the non-adaptive strategy from Eq. (4). We use monotonicity of f to get

$$\mathsf{alg}(\mathcal{T}, f) \quad = \quad \mathbb{E}_{I,R}[f(R) + \mathsf{alg}(\mathcal{T}_I, f_R)] \quad \ge \quad \mathbb{E}_{I,R}[f(R) + \mathsf{alg}(\mathcal{T}_I, f_{I\cup R})]. \tag{6}$$

Since $f_{I\cup R}$ is also a monotone submodular function over independent elements and $\mathcal{T}_I$ is an adaptive strategy tree of depth at most $d-1$, by induction hypothesis

$$\mathsf{alg}(\mathcal{T}_I, f_{I\cup R}) \ge \frac{1}{2}\mathsf{adap}(\mathcal{T}_I, f_{I\cup R}).$$

Combining this with Eq. (5) and Eq. (6), we get

$$\mathsf{alg}(\mathcal{T}, f) \ge \frac{1}{2}\mathsf{adap}(\mathcal{T}, f),$$

which finishes the proof of the upper bound by induction. ◀

3.2 Lower Bound of 2

In this section we show a monotone non-negative submodular function and a prefix-closed set of constraints where the adaptivity gap for stochastic probing is arbitrarily close to 2. Combined with §3.1, this proves Theorem 1 that the optimal adaptivity gap is exactly 2.

The proof below uses a stochastic probing instance on an infinite universe. Since submodular functions are defined only on finite sets, the proof below is informal. We do this to explain our main ideas and defer the formal proof to Appendix A.

Informal proof of the lower bound in Theorem 1. Our example is on a universe $V := \{e_{(k,l)} \mid k, l \in \mathbb{Z}_{\ge 0}\}$ where every element is independently active with probability ϵ for some $0 < \epsilon < 1$.

Example. We define our submodular objective f to be the weighted rank function of a partition matroid that selects at most one element from each part. The elements are partitioned according to their first label – for every $k \in \mathbb{Z}_{\ge 0}$ the set $\{e_{(k,l)} \mid l \in \mathbb{Z}_{\ge 0}\}$ is a

part of the partition matroid with weight $(1-\epsilon)^k$. In other words, for any set $S \subseteq V$ let $K(S) := \{k \mid e_{(k,l)} \in S\}$ be the (unique) set of first labels, then

$$f(S) \stackrel{\text{def}}{=} \sum_{k \in K(S)} (1-\epsilon)^k.$$

Note that this series always converges so f is well defined.

To define our prefix-closed constraints, we consider an infinite directed acyclic graph where every element is identified with a single node in the graph. Every node/element $e_{(k,l)}$ has exactly two outgoing edges: towards $e_{(k,l+1)}$ and towards $e_{(k+l+1,0)}$. We denote $\{e_{(k,0)}, e_{(k,1)}, \ldots\}$ as the elements on *column* k. The probing constraint is that a sequence of elements can be probed if and only if it corresponds to a directed path starting at $e_{(0,0)}$. See Figure 2 for an illustration.

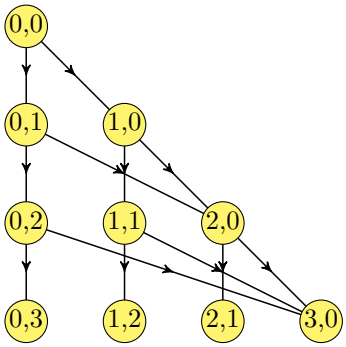


Figure 2 Adaptivity gap lower bound example for monotone submodular functions.

Analysis. We first give an adaptive strategy with value $2-\epsilon$ (in Eq. (7)) and later argue that every non-adaptive strategy has value at most 1 (in Eq. (8)); thereby, proving this theorem. Although, the probing constraint allows for infinite strategies, and in a different setting it would not be clear how to define their expected values, since f is monotone we include every active element in the solution. So the expected value of an infinite strategy can be defined as the limit of strategies that only probe a finite number of elements. The finite lower bound example in Appendix A is constructed by reducing V so that the resulting strategies are close to this limit.

Our adaptive strategy adap starts with probing element $e_{(0,0)}$. It is defined recursively: after probing $e_{(k,l)}$, the next element to probe is either $e_{(k+l+1,0)}$ if $e_{(k,l)}$ is found active, or $e_{(k,l+1)}$ otherwise. In other words, it probes elements on a column until it finds one active, and then probes another column.

Let $\mathsf{adap}(k)$ denote the expected additional value our above adaptive strategy if the next probed element is $e_{(k,0)}$ and let $\mathsf{adap} \stackrel{\text{def}}{=} \mathsf{adap}(0)$ denote the expected value of the entire strategy. Note that $\mathsf{adap}(k)$ does not depend on the set of elements found active before probing $e_{(k,0)}$ (i.e., the elements $e_{(k',l')}$ where $k' < k$). Furthermore, the subgraph reachable from $e_{(k,0)}$ is similar to the entire graph on V in the sense that one can relabel the elements in the subgraph to match the entire graph exactly, the only difference being that the value of any subset is multiplied by a factor of $(1-\epsilon)^k$. Therefore, we have

$$\mathsf{adap}(k) = (1-\epsilon)^k \cdot \mathsf{adap}(0).$$

Now, summing over the number of inactive elements on column 0, we get

$$\begin{aligned}\mathsf{adap}(0) &= \sum_{k=0}^{\infty}(1-\epsilon)^k \cdot \epsilon \cdot \Big(1+\mathsf{adap}(k+1)\Big) \\ &= \sum_{k=0}^{\infty}(1-\epsilon)^k \cdot \epsilon\Big(1+(1-\epsilon)^{k+1}\cdot \mathsf{adap}(0)\Big),\end{aligned}$$

which uses $\mathsf{adap}(k) = (1-\epsilon)^k \cdot \mathsf{adap}(0)$. Solving this equation yields the result:

$$\mathsf{adap} = \mathsf{adap}(0) = 2-\epsilon. \tag{7}$$

Similarly, let $\mathsf{alg}(k)$ denote the expected additional value of the optimal non-adaptive strategy if the next probed element is $e_{(k,0)}$, and let $\mathsf{alg} = \mathsf{alg}(0)$ denote the expected value of the optimal non-adaptive strategy. By the same argument as $\mathsf{adap}(k)$, we have

$$\mathsf{alg}(k) = (1-\epsilon)^k \cdot \mathsf{alg}(0).$$

Let k denote the number of elements the optimal non-adaptive strategy probes on column 0. We get

$$\mathsf{alg}(0) = \sup_{k\geq 1}\Big\{1-(1-\epsilon)^k + \mathsf{alg}(k)\Big\} = \sup_{k\geq 1}\Big\{1-(1-\epsilon)^k + (1-\epsilon)^k\cdot\mathsf{alg}(0)\Big\},$$

which uses $\mathsf{alg}(k) = (1-\epsilon)^k \cdot \mathsf{alg}(0)$. This implies

$$\mathsf{alg} = \mathsf{alg}(0) = 1. \tag{8}$$

Combining Eq. (7) and Eq. (8), we get an adaptivity gap arbitrarily close to 2 for $\epsilon \to 0$. ◀

4 Adaptivity Gaps for a Weighted Rank Function of a k-Extendible System

For a downward-closed family $\mathcal{F}$, recollect that we define its rank function $f_{\mathcal{F}} : 2^V \to \mathbb{R}_{\geq 0}$ to be the largest cardinality subset in $\mathcal{F}$, i.e., $f_{\mathcal{F}}(S) \stackrel{\text{def}}{=} \max_{T\subseteq S \ \&\ T\in\mathcal{F}} |T| = \max_{T\in\mathcal{F}} |S\cap T|$. In this section we prove our results on the adaptivity gaps of a weighted rank function of a k-extendible system.

▶ **Theorem 2.** *The adaptivity gap for SMP where the constraints are prefix-closed and the function is a weighted rank function of a k-extendible system is between k and $O(k\log k)$. Moreover, for unweighted rank functions, the adaptivity gap is between k and $2k$.*

In §4.1 we prove the upper bound for unweighted k-extendible systems, and in §4.2 we give a reduction from weighted to unweighted k-extendible systems that loses a factor $O(\log k)$ in the adaptivity gap. Our lower bound is presented in §4.3.

To simplify our proofs, we define an element $e \in T$ as a *loop* in $\mathcal{F} \subseteq 2^T$ if $\{e\} \notin \mathcal{F}$. Furthermore, given a non-loop element $e \in T$, we define the *contraction* $\mathcal{F}/e$ as $\{F \setminus \{e\} \mid F \in \mathcal{F}, e \in F\}$, i.e., the family of subsets that contain e but with e removed. We also need the following property of k-extendible systems, which intuitively means a set $E \in \mathcal{F}$ hurts at most $k\cdot|E|$ from another set $B \in \mathcal{F}$. We include the proof for completeness in Appendix B.

Let $\mathcal{F} \subseteq 2^T$ be a k-extendible system. For every $A \subseteq B \in \mathcal{F}$ and $E \subseteq T$ where $A\cup E \in \mathcal{F}$, there exists a set $Z \subseteq B \setminus A$ such that $|Z| \leq k\cdot|E|$ and $B\setminus Z \cup E \in \mathcal{F}$.

4.1 Upper Bound of $2k$ for an Unweighted k-Extendible System

Let $\mathcal{T}$ denote the optimal adaptive strategy for maximizing the rank function f of a given k-extendible system $\mathcal{F}$. We prove the following unweighted upper bound of Theorem 2.

▶ **Theorem 8.** *The adaptivity gap for SMP where the constraints are prefix-closed and the function is an unweighted rank function of a k-extendible system is at most $2k$.*

We use the random walk strategy to convert the adaptive strategy $\mathcal{T}$ into a non-adaptive strategy. To analyze our algorithm, we define a natural *greedy procedure* to select a subset of $A \subseteq T$ that is also in $\mathcal{F} \subseteq 2^T$. First, consider elements of A in an arbitrary order (which can even be determined on the fly). If the currently considered element is a non-loop, it gets contracted in $\mathcal{F}$; otherwise it gets ignored. Any such computed set is in $\mathcal{F}$ and the final output, the number of contracted elements, is denoted by $\text{greedy}(A)$. We first show that for k-extendible systems such a greedy procedure produces a k-approximation to the largest subset in $\mathcal{F}$. A similar statement has been proven by Mestre [29].

▶ **Lemma 9.** *Let f be a rank function of a k-extendible system $\mathcal{F} \subseteq 2^T$. Fix any subset $A \subseteq T$ and consider the output of the greedy procedure* $\text{greedy}(A)$ *with an arbitrary ordering of A. We have that $f(A) \leq k \cdot \text{greedy}(A)$. Even more, for any $A \subseteq B \subseteq T$ we have that $f(A) \leq k \cdot \text{greedy}(B)$.*

Proof. Let $G \subseteq B$ be the set picked by $\text{greedy}(B)$. Notice that G is a maximal set in $\mathcal{F}$ (need not be maximum). On the other hand, let $\text{OPT} \subseteq A$ be the set picked by $f(A)$, i.e., the maximum set in $\mathcal{F}$ on A. Our goal is to prove $|\text{OPT}| \leq k \cdot |G|$.

Let $C \stackrel{\text{def}}{=} \text{OPT} \cap G$, note that $G = C \cup (G \setminus C) \in \mathcal{F}$ and $C \subseteq \text{OPT}$, hence by Section 4 there is a $Z \subseteq \text{OPT} \setminus C$ with $|Z| \leq k \cdot |G \setminus C| = k \cdot |G| - k \cdot |C|$ such that $\text{OPT} \setminus Z \cup (G \setminus C) = (\text{OPT} \setminus C) \setminus Z \cup G \in \mathcal{F}$. However, since G is a maximal set and $(\text{OPT} \setminus C) \cap G = \emptyset$ we know that $\text{OPT} \setminus C \setminus Z = \emptyset$ and hence $|\text{OPT}| \leq |Z| + |C| \leq k \cdot |G| - k \cdot |C| + |C| = k \cdot |G| - (k-1)|C| \leq k \cdot |G|$. ◀

Given the above properties of a k-extendible system, we can now prove Theorem 8.

Proof of Theorem 8. Let $\mathbf{X}$ and $\mathbf{X}'$ denote the element types for the adaptive and the non-adaptive algorithms, respectively. The adaptive strategy on the optimal decision tree $\mathcal{T}$ gets value $f(\mathbf{X}_S)$, where $S \subseteq V$ is the set of probed elements by strategy $\mathcal{T}$ for type vector $\mathbf{X}$. We compare this value to a greedy strategy $\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)$ in which

1. we consider the elements of S in root-to-leaf order in which they appear on the tree and
2. for any $e \in S$ we first consider $\mathbf{X}'_e$ (the true type) before $\mathbf{X}_e$ (the virtual type) in the greedy order.

Note by Lemma 9 we have

$$\mathsf{adap}(\mathcal{T}, f) = \mathbb{E}_{\mathbf{X}}[f(\mathbf{X}_S)] \leq k \cdot \mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)].$$

By induction on the subtrees, below we prove

$$\mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)] \leq 2 \cdot \mathsf{alg}(\mathcal{T}, f). \tag{9}$$

This finishes the proof of Theorem 8 because the optimal non-adaptive algorithm has value at least

$$\mathsf{alg}(\mathcal{T}, f) \geq \frac{1}{2} \cdot \mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)] \geq \frac{1}{2k} \cdot \mathsf{adap}(\mathcal{T}, f).$$

To prove the missing Eq. (9), we induct on the height of the tree and $\mathcal{F}$ being any downward-closed family. For consistency, we define the notation of $\text{greedy}(\mathcal{T}, f)$ to denote the value of the above greedy strategy when run on $\mathcal{T}$ with a rank function f. Thus, $\text{greedy}(\mathcal{T}, f) = \mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)]$. Suppose $e \in V$ is the label of the root of $\mathcal{T}$. Denote by $I \stackrel{\text{def}}{=} X_e$ the (random) type of element e when probed by the adaptive strategy (which is also the virtual type of the non-adaptive strategy), and denote $R \stackrel{\text{def}}{=} X'_e$ the (random) true type when probed by the non-adaptive strategy. Also, let $\mathcal{T}_I$ denote the subtree the adaptive strategy goes to when the root e is in state I. We have

$$\text{greedy}(\mathcal{T}, f) \leq \mathbb{E}_{I,R}[f(I \cup R) + \text{greedy}(\mathcal{T}_I, (f/R)/I)],$$

where by $(f/R)/I$ we mean the rank function of $\mathcal{F}$ after we first contract R if it a non-loop, and then contract I if it is still a non-loop. Now subadditivity of f gives

$$\begin{aligned}\text{greedy}(\mathcal{T}, f) &\leq \mathbb{E}_{I,R}[f(I) + f(R) + \text{greedy}(\mathcal{T}_I, (f/R)/I)] \\ &= \mathbb{E}_{I,R}[2 \cdot f(R) + \text{greedy}(\mathcal{T}_I, (f/R)/I)], \end{aligned} \tag{10}$$

where the last equality uses linearity of expectation as I and R are identically distributed.

Next, we lower bound the value of our non-adaptive algorithm. Although it takes a random root-leaf path and decides the set of elements to retain in the end, we lower bound its value by an online algorithm that greedily selects R (unless it is a loop), however, always also contracts I if it is a non-loop. This gives,

$$\mathsf{alg}(\mathcal{T}, f) \geq \mathbb{E}_{I,R}[f(R) + \mathsf{alg}(\mathcal{T}_I, (f/R)/I)]. \tag{11}$$

Since $(f/R)/I$ is also a rank function of a downward-closed system and $\mathcal{T}_I$ is an adaptive strategy, by induction hypothesis we have

$$\mathsf{alg}(\mathcal{T}_I, (f/R)/I) \geq \frac{1}{2}\,\text{greedy}(\mathcal{T}_I, (f/R)/I).$$

Combining this with Eq. (10) and Eq. (11), we get

$$\text{greedy}(\mathcal{T}, f) \leq 2 \cdot \mathsf{alg}(\mathcal{T}, f),$$

which proves Eq. (9) by induction. ◀

4.2 Reducing Weighted to Unweighted k-Extendible System by Losing $O(\log k)$

We show how to extend the adaptivity gap result for an unweighted k-extendible system to a weighted k-extendible system by losing an $O(\log k)$ factor.

▶ **Theorem 10.** *For SMP over prefix-closed constraints, the adaptivity gap for a weighted rank function of a k-extendible system is at most $32k \log_2 k$.*

Proof. Given a weighted rank function f of a k-extendible system $\mathcal{F} \subseteq 2^T$ over a set of types T, we define f_j for $j \in \mathbb{Z}$ to be an unweighted rank function of the k-extendible system $\mathcal{F}$; however, the new weights are changed such that only the types with original weights in

$(2^{j-1}, 2^j]$ participate with new weight of 1, while the other elements have a new weight of 0. Note that this partitions the set of types T into pairwise disjoint *classes*. Notice, we have

$$\mathsf{adap}(\mathcal{T}, f) \leq \sum_j 2^j \cdot \mathsf{adap}(\mathcal{T}, f_j), \tag{12}$$

where $adap(\mathcal{T}, f_j)$ denotes the expected value of an adaptive strategy given by the common decision tree $\mathcal{T}$ with respect to the rank function f_j.

Now, since $\mathsf{adap}(\mathcal{T}, f_j)$ is an unweighted k-extendible system problem, we know that a random root-leaf path returns a solution with expected value

$$\mathsf{alg}(\mathcal{T}, f_j) \geq \frac{1}{2k} \cdot \mathsf{adap}(\mathcal{T}, f_j). \tag{13}$$

In the following lemma, we show that these non-adaptive solutions for f_j can be combined to obtain a feasible and "high-value" non-adaptive solution for f.

▶ **Lemma 11.** *The random-walk non-adaptive algorithm* alg *has expected value*

$$\mathit{alg}(\mathcal{T}, f) \geq \frac{1}{16 \cdot \log k} \sum_j 2^j \cdot \mathit{alg}(\mathcal{T}, f_j).$$

Before proving Lemma 11, we finish the proof of Theorem 10 by combining it with Eq. (13) and Eq. (12):

$$\begin{aligned}\mathsf{alg}(\mathcal{T}, f) \geq \frac{1}{16 \cdot \log k} \sum_j 2^j \cdot \mathsf{alg}(\mathcal{T}, f_j) &\geq \frac{1}{32k \log k} \sum_j 2^j \cdot \mathsf{adap}(\mathcal{T}, f_j) \\ &\geq \frac{1}{32k \log k} \cdot \mathsf{adap}(\mathcal{T}, f).\end{aligned}$$ ◀

Informally, in the proof of Lemma 11 we combine the unweighted solutions of $\mathsf{alg}(\mathcal{T}, f_i)$ by running a "greedy-optimal" algorithm from the higher weight to the smaller weight classes and fixing the types chosen in earlier classes. Unfortunately, in general such an approach loses an extra factor k in the approximation. To fix this, our second idea is to increase the weight gap between successive classes. We achieve this by combining $O(\log k)$ consecutive classes into a *bucket*, where in each bucket we focus on the class with the largest non-adaptive value. Because of boundary issues, we only take either odd or even buckets.

Proof of Lemma 11. Let $a \leq b \in \mathbb{Z}$ denote the indices of the smallest and the highest weight classes. We define buckets consisting of $2 \log k$ consecutive classes, where bucket B_i consists of classes $\{b - 2i \log k, b - 2i \log k - 1, \ldots, b - 2(i-1) \log k\}$. For each B_i, let

$$j(i) \stackrel{\text{def}}{=} \operatorname{argmax}_{j \in B_i} \{2^j \cdot \mathsf{alg}(\mathcal{T}, f_j)\}.$$

Since each bucket has size $2 \log k$, this implies

$$\sum_i 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)}) \geq \frac{1}{2 \cdot \log k} \sum_j 2^j \cdot \mathsf{alg}(\mathcal{T}, f_j).$$

Without loss of generality we can assume the odd indices satisfy

$$\sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)}) \geq \frac{1}{2} \sum_i 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)}).$$

Otherwise, use the same argument for even indices. Combining the last two equations, we get

$$\sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)}) \geq \frac{1}{4 \cdot \log k} \sum_{j} 2^{j} \cdot \mathsf{alg}(\mathcal{T}, f_j). \tag{14}$$

We now claim that a *greedy-optimal* algorithm has a large value: It goes over classes $j(i)$ in decreasing order of (odd) buckets, but it always selects the maximum independent set (instead of selecting a maximal greedy set) in the current class $j(i)$ given its choices in the previous. This algorithm is, therefore, a combination of greedy and optimal algorithms. The proof of the following is deferred to Appendix C.

▷ Claim 12. Consider an algorithm that goes over the odd numbered buckets in decreasing order of weights and selects the *maximum* set from class $j(i)$ in bucket i such that the resulting set is still feasible in $\mathcal{F}$. (After a set in a class is selected, it gets fixed for all smaller choices.) The finally chosen set has value at least $\frac{1}{4} \sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)})$.

Using Claim 12, we have

$$\mathsf{alg}(\mathcal{T}, f) \geq \frac{1}{4} \sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)}),$$

which combined when with Eq. (14) proves Lemma 11. ◀

4.3 Lower Bounds

We present two very similar lower bound examples: one where the adaptivity gap is $k - o(1)$ for a rank function of an unweighted k-extendible system and another where the adaptivity gap is $\Omega(\sqrt{k})$ for a rank function of an intersection of k matroids. A related example was also shown in [23].

Example. For generality we work in the Bernoulli setting where each element in V is either active or inactive. Consider a perfect w-ary tree of depth k whose edges correspond to the ground set V. Each edge is active with probability $p > 0$. For any leaf ℓ, let P_ℓ denote the unique path from the root to ℓ. The objective value on any set is the maximum number of edges in the set on the *same* root-leaf path, i.e., for any $S \subseteq V$,

$$f(S) \stackrel{\text{def}}{=} \max_{\text{leaf } \ell} |P_\ell \cap S|.$$

The feasibility constraints are such that a set of edges can be probed if and only if there exists some root-leaf path P_ℓ such that every probed edge has at least one endpoint on P_ℓ. Note that this implies that a maximum of $w \cdot k$ edges can be probed.

Analysis. Let the adaptive strategy be the following: probe all w edges incident to the root. If any of them is active, start probing the edges directly below the active edge, otherwise below the first edge. Continue recursively until a leaf is reached. On every level, the adaptive strategy has $1 - (1-p)^w$ probability of finding an active edge. Therefore, the expected value of the adaptive strategy is $k \cdot (1 - (1-p)^w)$.

For any non-adaptive strategy, the feasibility constraints imply there exists a root-leaf path P_ℓ such that all probed edges have an endpoint on it. Suppose all $w \cdot k$ edges incident to P_ℓ are probed. The non-adaptive strategy can get value at most 1 from the edges not on P_ℓ and in expectation at most $k \cdot p$ from the edges on P_ℓ. So, the non-adaptive strategy has an expected value of at most $1 + k \cdot p$.

Lower Bound of k for an unweighted k-extendible system

Consider the example described above and set $w \stackrel{\text{def}}{=} k^4$ and $p \stackrel{\text{def}}{=} \frac{1}{k^3}$. The function f is trivially a rank function of a k-extendible system because the rank of the system is k, i.e., $f(V) = k$. The adaptive strategy has an expected value

$$k \cdot \left(1 - \left(1 - \frac{1}{k^3}\right)^{k^4}\right) \geq k \cdot \left(1 - \frac{1}{e^k}\right) = k - o(1),$$

whereas any non-adaptive strategy has an expected value at most $1 + \frac{1}{k^2}$. This gives an adaptivity gap of $k - o(1)$.

Lower Bound of $\Omega(\sqrt{k})$ for an unweighted intersection of k matroids

In this section we show how to model the above example as an intersection of $t = k^2$ matroids, yielding an adaptivity gap of $\Omega(\sqrt{t})$ for an intersection of t matroids. Consider the example described above and set $w \stackrel{\text{def}}{=} k$ and $p \stackrel{\text{def}}{=} \frac{1}{k}$. The adaptive strategy has an expected value of

$$k \cdot \left(1 - \left(1 - \frac{1}{k}\right)^{k}\right) \geq k \cdot \left(1 - \frac{1}{e}\right) = \Omega(k)$$

and the non-adaptive strategy gets at most 2 in expectation; so the adaptivity gap is $\Omega(k)$.

All that remains to show is that f can be represented as an intersection of k^2 simple partition matroids. We use the term simple partition matroid for a matroid that partitions the V into multiple parts and a set is independent if it contains at most one element in every part.

Suppose that k is prime and label each node v with a list L_v as follows: the root's label is an empty list $()$. Let $L(i)$ denote the i^{th} element of the list L and $L + x$ a list equal to L with x appended to it. All the other nodes are labeled recursively: let v be a node with children $\{v_0, v_1, ...v_{k-1}\}$. Define $L_{v_i} \stackrel{\text{def}}{=} L_v + i$. Hence, u is an ancestor of v if and only if L_u is a prefix of L_v, and otherwise $L_u(i) \neq L_v(i)$ for some i.

Let e_v denote the edge/element between v and its parent. We define k^2 partition matroids $M_{i,j}$ for $i \in \{1, 2, ..., k\}$ and $j \in \{0, 1, ..., k-1\}$. Each $M_{i,j}$ consists of k *big* partitions indexed from 0 to $k - 1$, and all other partitions contain only a single element. Let

$$I_v(i, j) \stackrel{\text{def}}{=} L_v(i)j + d_v (\text{mod } k).$$

For a node v on depth $d_v \geq i$, element e_v is in the $I_v(i, j)^{th}$ *big* partition of $M_{i,j}$. For a node v on depth $d_v < i$, e_v is the only element in its partition in $M_{i,j}$.

We claim that f is the rank function of $\mathcal{F} \stackrel{\text{def}}{=} \bigcap_{i=1}^{k} \bigcap_{j=0}^{k-1} M_{i,j}$, which is an intersection of k^2 matroids. Since $\mathcal{F}$ is an intersection of simple partition matroids, $S \in \mathcal{F}$ if and only if $\{a, b\} \in \mathcal{F}$ for every $a, b \in S$. Now consider two nodes u, v such that $\{e_u, e_v\} \notin \mathcal{F}$. This means $I_u(i, j) = I_v(i, j)$ for some $i \leq d_u, d_v$ and $j \in \{0, 1, ..., k - 1\}$, which is equivalent to

$$L_u(i) \cdot j + d_u \equiv L_v(i) \cdot j + d_v (\text{mod } k).$$

Since k is prime, this holds for some i, j if and only if $d_u = d_v$ (for $j = 0, i = 1$) or $L_u(i) \neq L_v(i)$ for any i. That is, $\{e_u, e_v\} \notin \mathcal{F}$ if and only if u and v are not ancestors of one another, which completes the proof.

References

1 Marek Adamczyk. Improved analysis of the greedy algorithm for stochastic matching. *Inf. Process. Lett.*, 111(15):731–737, 2011.

2 Marek Adamczyk, Fabrizio Grandoni, and Joydeep Mukherjee. Improved approximation algorithms for stochastic matching. In *Algorithms-ESA 2015*, pages 1–12. Springer, 2015.

3 Marek Adamczyk, Maxim Sviridenko, and Justin Ward. Submodular Stochastic Probing on Matroids. In *STACS*, pages 29–40, 2014.

4 Arash Asadpour and Hamid Nazerzadeh. Maximizing stochastic monotone submodular functions. *Management Science*, 62(8):2374–2391, 2016.

5 Arash Asadpour, Hamid Nazerzadeh, and Amin Saberi. Stochastic submodular maximization. In *International Workshop on Internet and Network Economics*, pages 477–489. Springer, 2008. Full version appears as [4].

6 Itai Ashlagi and Alvin E. Roth. New Challenges in Multihospital Kidney Exchange. *American Economic Review*, 102(3):354–59, 2012.

7 Nikhil Bansal, Anupam Gupta, Jian Li, Julián Mestre, Viswanath Nagarajan, and Atri Rudra. When LP Is the Cure for Your Matching Woes: Improved Bounds for Stochastic Matchings. *Algorithmica*, 63(4):733–762, 2012.

8 Nikhil Bansal and Viswanath Nagarajan. On the Adaptivity Gap of Stochastic Orienteering. In *IPCO*, pages 114–125, 2014.

9 Alok Baveja, Amit Chavan, Andrei Nikiforov, Aravind Srinivasan, and Pan Xu. Improved Bounds in Stochastic Matching and Optimization. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, pages 124–134, 2015.

10 Anand Bhalgat, Ashish Goel, and Sanjeev Khanna. Improved Approximation Results for Stochastic Knapsack Problems. In *SODA*, pages 1647–1665, 2011.

11 Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a Monotone Submodular Function Subject to a Matroid Constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.

12 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular Function Maximization via the Multilinear Relaxation and Contention Resolution Schemes. *SIAM J. Comput.*, 43(6):1831–1879, 2014. doi:10.1137/110839655.

13 Ning Chen, Nicole Immorlica, Anna R. Karlin, Mohammad Mahdian, and Atri Rudra. Approximating Matches Made in Heaven. In *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, pages 266–278, 2009.

14 Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 208–217. IEEE, 2004.

15 Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Adaptivity and approximation for stochastic packing problems. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, pages 395–404, 2005.

16 Amol Deshpande, Lisa Hellerstein, and Devorah Kletenik. Approximation Algorithms for Stochastic Boolean Function Evaluation and Stochastic Submodular Set Cover. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1453–1466, 2014. doi:10.1137/1.9781611973402.107.

17 Sudipto Guha and Kamesh Munagala. Approximation algorithms for budgeted learning problems. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 104–113. ACM, 2007.

18 Sudipto Guha and Kamesh Munagala. Multi-armed Bandits with Metric Switching Costs. In *Automata, Languages and Programming, 36th Internatilonal Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part II*, pages 496–507, 2009.

19 Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R. Ravi. Approximation Algorithms for Correlated Knapsacks and Non-martingale Bandits. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 827–836, 2011.

20 Anupam Gupta, Ravishankar Krishnaswamy, Viswanath Nagarajan, and R. Ravi. Approximation algorithms for stochastic orienteering. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1522–1538, 2012.

21 Anupam Gupta and Viswanath Nagarajan. A Stochastic Probing Problem with Applications. In *Integer Programming and Combinatorial Optimization - 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings*, pages 205–216, 2013.

22 Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Algorithms and adaptivity gaps for stochastic probing. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1731–1747. SIAM, 2016.

23 Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Adaptivity Gaps for Stochastic Probing: Submodular and XOS Functions. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1688–1702. SIAM, 2017.

24 Jian Li Hao Fu and Pan Xu. A PTAS for a Class of Stochastic Dynamic Programs. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2018*, 2018.

25 Lisa Hellerstein, Devorah Kletenik, and Patrick Lin. Discrete Stochastic Submodular Maximization: Adaptive vs. Non-adaptive vs. Offline. In *Algorithms and Complexity - 9th International Conference, CIAC 2015, Paris, France, May 20-22, 2015. Proceedings*, pages 235–248, 2015. doi:10.1007/978-3-319-18173-8_17.

26 Jian Li and Wen Yuan. Stochastic combinatorial optimization via poisson approximation. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 971–980, 2013. doi:10.1145/2488608.2488731.

27 Zhen Liu, Srinivasan Parthasarathy, Anand Ranganathan, and Hao Yang. Near-optimal algorithms for shared filter evaluation in data stream systems. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 133–146, 2008. doi:10.1145/1376616.1376633.

28 Will Ma. Improvements and Generalizations of Stochastic Knapsack and Multi-Armed Bandit Approximation Algorithms: Extended Abstract. In *SODA*, pages 1154–1163, 2014.

29 Julián Mestre. Greedy in approximation algorithms. In *European Symposium on Algorithms*, pages 528–539. Springer, 2006.

30 Alvin E. Roth, Tayfun Sönmez, and M.Ũtku Ünver. Pairwise kidney exchange. *J. Econom. Theory*, 125(2):151–188, 2005. doi:10.1016/j.jet.2005.04.004.

31 Aviad Rubinstein and Sahil Singla. Combinatorial prophet inequalities. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1671–1687. SIAM, 2017.

A Adaptivity Gap Lower Bound of 2 for Submodular Functions

Proof. As mentioned, the finite lower bound example is constructed by reducing the infinite example given in Section 3.2. However, this reduction loses the nice similarity properties of the graph so much more calculation is required in order to bound the strategies.

Let $0 < \epsilon < 1/2$ and D be the smallest integer such that $(1-\epsilon)^D < \epsilon^2$. The ground set is the result of removing elements $e_{(k,l)}$ where $k + l > D$, that is $V \stackrel{\text{def}}{=} \{e_{(k,l)} : k, l \in \mathbb{Z}_{\geq 0}, k + l \leq D\}$ where each node is active with probability ϵ. The probing constraint

and the objective function f are naturally reduced to this set: a sequence of elements can be probed if they correspond to a (finite) path starting at $e_{(0,0)}$ in the given graph, and $f(S) \stackrel{\text{def}}{=} \sum_{k \in K(S)} (1-\epsilon)^k$ where $K(S)$ is the set of (unique) first labels which now finite. Similarly as before, we will denote $\{e_{(k,0)}, e_{(k,1)}, \ldots, e_{(k,D-k)}\}$ as the vertices on the *column* k.

We first show that any non-adaptive strategy has expectation at most 1. Let $\mathsf{alg}(k)$ denote the additional expected value of the optimal non-adaptive strategy if the next probed element is $e_{(k,0)}$. We will inductively prove $\mathsf{alg}(k) < (1-\epsilon)^k$, which is sufficient for our claim. For the base case $k = D$, the inequality clearly holds since $\mathsf{alg}(D) = \epsilon(1-\epsilon)^D < (1-\epsilon)^D$. For $0 \leq k < D$ let i be the second label of the last vertex probed on the column k.

$$\begin{aligned}
\mathsf{alg}(k) &= \max_{i=0}^{D-k} \Big[(1-\epsilon)^k \Pr[k \in K(\text{active})] + \mathsf{alg}(k+i+1) \Big] \\
&= \max_{i=0}^{D-k} \Big[(1-\epsilon)^k (1-(1-\epsilon)^{i+1}) + \mathsf{alg}(k+i+1) \Big] \\
&< \max_{i=0}^{D-k} \Big[(1-\epsilon)^k (1-(1-\epsilon)^{i+1}) + (1-\epsilon)^{k+i+1} \Big] \quad = \quad (1-\epsilon)^k.
\end{aligned}$$

This completes the induction and proves that non-adaptive strategies get at most 1.

Finally, we show that there exists an adaptive strategy with expected value at least $2 - O(\epsilon)$ for sufficiently small $\epsilon > 0$. This finalizes the proof since it implies a gap of 2 by taking $\epsilon \to 0$. The strategy is naturally reduced: first probe $e(0,0)$ and after probing some $e_{(k,l)}$ terminate if $k + l = D$, otherwise probe $e_{(k+l+1,0)}$ if $e_{(k,l)}$ is active and $e_{(k,l+1)}$ if not. Let $\mathsf{adap}(k)$ denote the expected value this strategy gets when the next probed element is $e_{(k,0)}$, for $0 \leq k \leq D$. For convenience, define $\mathsf{adap}(D+i) \stackrel{\text{def}}{=} 0$ for all $i \geq 1$.

We prove by induction that $\mathsf{adap}(k) > \frac{4-6\epsilon}{2-\epsilon}(1-\epsilon)^k - 8\epsilon$, which is sufficient to finalize the proof since then $\mathsf{adap}(0) > 2 - O(\epsilon)$. For k large enough that $\frac{4-6\epsilon}{2-\epsilon}(1-\epsilon)^k < 8\epsilon$, the inequality clearly holds and presents our base case. Otherwise, $(1-\epsilon)^k \geq 8\frac{2-\epsilon}{4-6\epsilon}\epsilon > 4\epsilon$. Let i be the second label of the last vertex probed on the column k and let A denote the set of active elements.

$$\begin{aligned}
\mathsf{adap}(k) &= \sum_{i=0}^{D-k} \Pr\Big[v_{(k,i)} \in A, v_{(k,0)} \notin A, \ldots, v_{(k,i-1)} \notin A \Big] \Big[(1-\epsilon)^k + \mathsf{adap}(k+i+1) \Big] \\
&= \sum_{i=0}^{D-k} (1-\epsilon)^i \epsilon \Big[(1-\epsilon)^k + \mathsf{adap}(k+i+1) \Big] \\
&= \epsilon \cdot \sum_{i=0}^{D-k} (1-\epsilon)^{k+i} + \epsilon \cdot \sum_{i=0}^{D-k} (1-\epsilon)^i \mathsf{adap}(k+i+1) \\
&= \epsilon \cdot \frac{1}{\epsilon}(1-\epsilon)^k \Big(1 - (1-\epsilon)^{D-k+1} \Big) + \epsilon \cdot \sum_{i=0}^{D-k} (1-\epsilon)^i \cdot \mathsf{adap}(k+i+1).
\end{aligned}$$

Using the induction hypothesis, we get

$$\begin{aligned}
\mathsf{adap}(k) &> (1-\epsilon)^k - (1-\epsilon)^{D+1} + \epsilon \sum_{i=0}^{D-k} (1-\epsilon)^i \Big(\frac{4-6\epsilon}{2-\epsilon}(1-\epsilon)^{k+i+1} - 8\epsilon \Big) \\
&= (1-\epsilon)^k - (1-\epsilon)^{D+1} + \epsilon \sum_{i=0}^{D-k} \frac{4-6\epsilon}{2-\epsilon}(1-\epsilon)^{k+2i+1} - 8\epsilon^2 \sum_{i=0}^{D-k} (1-\epsilon)^i \\
&= (1-\epsilon)^k - (1-\epsilon)^{D+1} + (1-\epsilon)^{k+1} \frac{4-6\epsilon}{(2-\epsilon)^2} \Big(1 - (1-\epsilon)^{2(D-k+1)} \Big) \\
&\qquad - 8\epsilon \Big(1 - (1-\epsilon)^{D-k+1} \Big).
\end{aligned}$$

After dropping some positive summands and using $(1-\epsilon)^D < \epsilon$ and $(1-\epsilon)^k > \epsilon$, we get

$$\mathsf{adap}(k) > (1-\epsilon)^k - \epsilon^2 + (1-\epsilon)^{k+1}\frac{4-6\epsilon}{(2-\epsilon)^2}(1-\epsilon^2) - 8\epsilon.$$

It is sufficient to prove

$$(1-\epsilon)^k - \epsilon^2 - 8\epsilon + (1-\epsilon)^{k+1}\frac{4-6\epsilon}{(2-\epsilon)^2}(1-\epsilon^2) > \frac{4-6\epsilon}{2-\epsilon}(1-\epsilon)^k - 8\epsilon.$$

Multiplying by $\frac{(2-\epsilon)^2}{(1-\epsilon)^k} > 0$, we get an equivalent statement to prove:

$$(2-\epsilon)^2 - \epsilon^2 \cdot \frac{(2-\epsilon)^2}{(1-\epsilon)^k} + (1-\epsilon)(4-6\epsilon)(1-\epsilon^2) > (4-6\epsilon)(2-\epsilon).$$

Finally, using $\epsilon^2\frac{(2-\epsilon)^2}{(1-\epsilon)^k} < \epsilon^2(2-\epsilon)^2\frac{1}{4\epsilon} = \epsilon + O(\epsilon^2)$ and expanding out, we note that the left-hand side is $8 - 15\epsilon + O(\epsilon^2)$, while the right-hand side is $8 - 16\epsilon + O(\epsilon^2)$. Therefore, the inequality holds for sufficiently small $\epsilon > 0$. This concludes the proof. ◀

B Proof of the k-Extendible Property for Set Extension

Let $\mathcal{F} \subseteq 2^T$ be a k-extendible system. For every $A \subseteq B \in \mathcal{F}$ and $E \subseteq T$ where $A \cup E \in \mathcal{F}$, there exists a set $Z \subseteq B \setminus A$ such that $|Z| \leq k \cdot |E|$ and $B \setminus Z \cup E \in \mathcal{F}$.

Proof. Enumerate the elements $E = \{e_1, \ldots, e_r\}$ where $r \stackrel{\text{def}}{=} |E|$ and denote by $E_i \stackrel{\text{def}}{=} \{e_1, \ldots, e_i\}$ for $0 \leq i \leq r$. Initialize $Z_0 \stackrel{\text{def}}{=} \emptyset$ and consider the following procedure to construct $Z_1, Z_2, \ldots, Z_r$ that satisfies the invariants $A \subseteq B \setminus Z_i$, $B \setminus Z_i \cup E_i \in \mathcal{F}$ and $|Z_i| \leq k \cdot i$.

In the i^{th} step we have that $A \cup E_{i-1} \cup \{e_i\} \in \mathcal{F}$ by downward-closeness and $A \cup E_{i-1} \subseteq B \setminus Z_{i-1} \cup E_{i-1}$ by the induction hypothesis. Hence by k-extendibility we can find $Z' \subseteq B \setminus (Z_{i-1} \cup A \cup E_{i-1})$ with $|Z'| \leq k$ and where $(B \setminus Z_{i-1} \cup E_{i-1}) \setminus Z' \cup \{e_i\} = B \setminus (Z_{i-1} \cup Z') \cup E_i \in \mathcal{F}$. Set $Z_i \stackrel{\text{def}}{=} Z_{i-1} \cup Z'$ and note that $|Z_i| \leq |Z_{i-1}| + |Z'| \leq (i-1) \cdot k + k = i \cdot k$. Furthermore, already deduced that $B \setminus Z_i \cup E_i \in \mathcal{F}$ and finally $A \subseteq B \setminus Z_i = B \setminus Z_{i-1} \setminus Z'$ since $Z' \cap A = \emptyset$. We satisfied all stipulations of the induction, hence we report Z_r as the solution. ◀

C Proof of Claim 12

▷ Claim 12. Consider an algorithm that goes over the odd numbered buckets in decreasing order of weights and selects the *maximum* set from class $j(i)$ in bucket i such that the resulting set is still feasible in $\mathcal{F}$. (After a set in a class is selected, it gets fixed for all smaller choices.) The finally chosen set has value at least $\frac{1}{4}\sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)})$.

Proof. The intuition is that for a k-extendible system by Section 4 any selected member can "hurt" at most k members from lower buckets. Since we only consider odd numbered buckets, two types in different buckets differ in their weights by at least a factor of $2^{2\log k} = k^2$. Thus, losing k types of lower weight should not significantly impact the value.

Let ℓ be the random variable denoting the leaf reached by the random walk on the decision tree $\mathcal{T}$, and let R be the random set of elements seen by the random-walk non-adaptive strategy on this path. Furthermore, let A_i denote the set of elements picked by the non-adaptive strategy with respect to $f_{j(i)}$, let $A'_i \subseteq A_i$ be the set of elements picked by our greedy-optimal non-adaptive strategy from bucket i, and let $A'_{<i}$ denote $\bigcup_{i' < i \,:\, i' \text{ is odd}} A_{i'}$.

In other words, $A'_{<i}$ is the greedy-optimal solution up to bucket number i and A'_i is the maximum subset of A_i such that $A'_i \cup A'_{<i} \in \mathcal{F}$. Note that A_i, A'_i and $A'_{<i}$ are random variables depending on ℓ and R.

Using Section 4 on the k-extendible system $\mathcal{F}$ with the preconditions $\emptyset \cup A'_{<i} \in \mathcal{F}$ and $\emptyset \subseteq A_i$, there exists a set Z with $|Z| \leq k \cdot |A'_{<i}|$ such that $A_i \setminus Z \in \mathcal{F}$. Hence, we have

$$|A'_i| \geq |A_i \setminus Z| \geq |A_i| - k \cdot |A'_{<i}|.$$

Multiplying by $2^{j(i)}$ and summing over all odd i gives

$$\begin{aligned}\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_i| &\geq \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - k \cdot \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_{<i}| \\ &= \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - k \cdot \sum_{i \text{ is odd}} |A'_i| \sum_{i' > i \,:\, i' \text{ is odd}} 2^{j(i')}. \end{aligned} \tag{15}$$

Now, since every bucket i contains $2 \log k$ classes, where two successive class weights differ by a factor of 2, we know

$$2^{j(i+2)} \leq \frac{2^{j(i)}}{k^2}.$$

Combining this with Eq. (15) gives

$$\begin{aligned}\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_i| &\geq \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - k \cdot \sum_{i \text{ is odd}} |A'_i| \sum_{i' > i \,:\, i' \text{ is odd}} \frac{2^{j(i'+2)}}{k^2} \\ &\geq \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - \sum_{i \text{ is odd}} |A'_i| \cdot 2^{j(i)},\end{aligned}$$

where the last inequality uses

$$\sum_{i' > i \,:\, i' \text{ is odd}} 2^{j(i'+2)} = \sum_{i' \geq i \,:\, i' \text{ is odd}} 2^{j(i')} \leq 2 \cdot 2^{j(i)} \leq k \cdot 2^{j(i)}.$$

After rearranging,

$$\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_i| \geq \frac{1}{2} \cdot \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i|.$$

Notice that by definition of a class, each type in class $j(i)$ has weight at least $2^{j(i)-1}$. Using this fact and taking expectation over ℓ and R, we get

$$\begin{aligned}\mathsf{alg}(\mathcal{T}, f) &\geq \mathbb{E}_{\ell,R}\Big[\sum_{i \text{ is odd}} 2^{j(i)-1} \cdot |A'_i|\Big] \\ &\geq \frac{1}{4}\mathbb{E}_{\ell,R}\Big[\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i|\Big] = \frac{1}{4} \sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathsf{alg}(\mathcal{T}, f_{j(i)}),\end{aligned}$$

which finishes the proof of Claim 12. ◁

Testing Odd Direct Sums Using High Dimensional Expanders

Roy Gotlib
Bar-Ilan University, Ramat Gan, Israel
roy.gotlib@gmail.com

Tali Kaufman
Bar-Ilan University, Ramat Gan, Israel
kaufmant@mit.edu

Abstract

In this work, using methods from high dimensional expansion, we show that the property of *k-direct-sum* is testable for *odd* values of k . Previous work of [9] could inherently deal only with the case that k is even, using a reduction to linearity testing. Interestingly, our work is the first to combine the topological notion of high dimensional expansion (called co-systolic expansion) with the combinatorial/spectral notion of high dimensional expansion (called colorful expansion) to obtain the result.

The classical k-direct-sum problem applies to the complete complex; Namely it considers a function defined over *all* k-subsets of some n sized universe. Our result here applies to any collection of k-subsets of an n-universe, assuming this collection of subsets forms a high dimensional expander.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases High Dimensional Expanders, Property Testing, Direct Sum

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.50

Category RANDOM

Funding *Roy Gotlib*: Research supported by ERC.
Tali Kaufman: Research supported by ERC and BSF.

1 Introduction

Given a collection X of k-subsets of $[n]$, a function $F : X \to \{0,1\}$ is a k-direct-sum if there exists a function $f : [n] \to \{0,1\}$ such that for every A in X: $F(A) = \sum_{a \in A} f(a)$ (where the sum is performed modulo 2). A (Q, E)-tester for k-direct-sums is an algorithm that queries F on Q inputs from X, accepts k-direct-sums and rejects with probability of at least ξ, every function whose distance from the k-direct-sums is at least $E\xi$ (see Definition 16 for distance and [8] for a survey on property testing). In this work we present a new novel method for testing k-direct-sums using high dimensional expanders. Our method is the first to deal with k-direct-sums for odd constant values of k.

The question of testing whether a function is a k-direct-sum, as well as the entire area of testability, has strong relations to PCP constructions. For example, one can consider the gap amplification proof of the PCP theorem [5]. This proof uses two steps: First powering the graph which results in every node having an "opinion" about its neighbors' color (which increases the alphabet size) and then reducing the alphabet. A better understanding of the direct sum problem could potentially help in replacing the direct product done in the graph powering phase, and might even allow omitting the alphabet reduction stage which would yield a simpler proof to the PCP theorem and, possibly, better parameters.


© Roy Gotlib and Tali Kaufman;
licensed under Creative Commons License CC-BY



Previous Work

There were several works on k-direct-sums, but none of them could deal with the odd constant case due to inherent limitations of their methods: The first work to link direct-sums and high dimensional expanders was done by Kaufman and Lubotzky [9], who showed a test for the 2-direct-sum problem on any simplicial complex that is a high dimensional expander. Their proof is tailored to the case where $k = 2$. Following the work of Kaufman and Lubotzky was a work by David, Dinur, Goldenberg, Kindler, and Shinkar [4] that proposed a tester for k-direct-sums on the for the case where the input set is $\binom{[n]}{k}$. Their tester is based on linearity testing: It picks $x, y \in X$ such that $x\Delta y \in X$, and tests whether $f(x) + f(y) = f(x\Delta y)$ (for more papers on linearity testing see [1, 2, 3]). But in order to get $x, y, x\Delta y \in X$, $\boldsymbol{k}$ **must be even**. In a recent work by Dinur and Kaufman [6], it is shown that the result of David et al. [4] can be applied to testing functions whose inputs are taken from a subset of $\binom{[n]}{k}$ that forms a high dimensional expander. However the limitation above still stands.

In this paper we introduce a new method for testing k-direct-sums that can tackle the odd case for the first time. Specifically we show:

▶ **Theorem 1** (Main Theorem Informal, for formal see Theorem 34)**.** *If X is a collection of subsets that forms a high dimensional expander then there is an $\left(O\left(k^2\right), O\left(k^2\right)\right)$-tester for the k-direct-sums where k is an odd constant.*

Interestingly we combine two notions of high dimensional expanders, a topological notion and a combinatorial notion, to obtain this result. This is the first time that both notions were used together.
In order to describe our strategy we will first have to introduce a generalization of graphs to higher dimensions (called simplicial complexes) as well as both notions of high dimensional expanders:

Simplicial Complexes

A simplicial complex can be thought of as a hypergraph with a closure property, meaning that if F is a hyperedge in the hypergraph then so is every subset of F. We also define the dimension of a hyperedge F to be $|F| - 1$, and denote the set of i-dimensional edges of a complex X as $X(i)$. For example: In a graph, the vertices are considered the 0-dimensional hyperedges, and the edges are considered the 1-dimensional hyperedges. Now that we have defined the dimension of a hyperedge, we can define the dimension of the complex as the dimension of the maximal hyperedge. For example: A 2-dimensional simplicial complex is a simplicial complex that contains 2-dimensional hyperedges, often called the "triangles" (note that these hyperedges contain 3 vertices). Throughout this paper, we will use a standard weighted counting norm denoted as $\|\cdot\|$ (which will be defined in 8). In this work we will be interested in simplicial complexes whose maximal hyperedges are all of the same dimension (which are called "pure simplicial complexes").

As previously discussed, we will use two generalizations of expansion that apply to simplicial complexes: The first will be co-systolic expanders and the second will be colorful expanders. In order to discuss these notions of expansion, it will be useful to reexamine the Cheeger constant in the 1-dimensional case (aka graphs):

$$\min_{S \neq \emptyset, V} \left\{ \frac{\left\| E(S, \bar{S}) \right\|}{\min\left\{ \|S\|, \left\|\bar{S}\right\| \right\}} \right\}$$

In any higher dimensional analogue, we would still like the essence of this constant to hold, every set of hyperedges of some dimension i (in graphs - vertices) has a number of out-going hyperedges of dimension $i+1$ (in graphs - edges) relative to its size. Because we are dealing with multi-dimensional objects we would also like this bound to apply in every dimension. The only question remaining is how to generalize the notion of an out-going edge to higher dimensions.

Co-systolic Expanders

The first notion of expansion we will introduce is the co-systolic expansion, which is the more topological of the two. In this form of expansion, a hyperedge of dimension $i+1$, is said to be going out of a set E of hyperedges of dimension i, if it has an odd number of i-dimensional sub-edges in the set E[1]. We denote the set of hyperedges that are going out of a set E, according to this notion, as δE. Note that the Cheeger constant is normalized over the distance of E from a set that has no neighbors (in the 1-dimensional case the only sets with no neighbors are the empty set and the entire graph). Therefore we normalize our new high-dimensional analogue accordingly and receive the following definition:

$$\epsilon^i(X) = \min_{\substack{S \in \{0,1\}^{X(i)} \\ \delta S \neq \emptyset}} \left\{ \frac{\|\delta S\|}{dist(S, \{Z | \delta Z = \mathbb{0}\})} \right\}$$

A simplicial complex is a co-systolic expander if there exists some ϵ such that in every dimension i: $\epsilon^i(X) \geq \epsilon$. Note that there is another property that a simplicial complex must fulfill in order to be a co-systolic expander. However, it is not required in the proof of this paper and can be found in definition 20.

Colorful Expanders

The other notion of expansion we will introduce is the colorful expansion, which is the more combinatorial of the two. In this form of expansion, a hyperedge of dimension $i+1$, is said to be going out of a set E of hyperedges of dimension i, if it has at least one i-dimensional sub-edge in E and at least one i-dimensional sub-edge outside of E. We denote the set of hyperedges that are going out of a set E, according to this notion, as $c(E)$. Using this definition of out-going edges, we get the following generalization of the Cheeger constant (for the i-th dimension):

$$\sigma^i(X) = \min_{S \neq \emptyset, X(i)} \left\{ \frac{\|c(S)\|}{\min\{\|S\|, \|X(i) \setminus S\|\}} \right\}$$

A simplicial complex is a σ-colorful-expander if in every dimension i: $\sigma^i(X) \geq \sigma$.

1.1 Proof Layout

We will start by defining the property of being a k-direct-sum again, this time using the language of simplicial complexes: Given a simplicial complex X, a function $F : X(k-1) \to \{0,1\}$ is called a k-direct-sum if there exists a function $f : X(0) \to \{0,1\}$ such that for every A in $X(k-1)$: $F(A) = \sum_{a \in A} f(a)$. Note that we define a k-direct-sum to be a function from

[1] In the 1-dimensional case we say that an edge crosses a cut if it has exactly one vertex in the cut. Note that if there is an odd number of vertices in an edge the odd number must be one since edges are of cardinality 2.

the $(k-1)$-dimensional hyperedges of the complex, and not the k-dimensional hyperedges of the complex, because we want the k to represent the size of the set and not the dimension of the face.

We will show that the following algorithm tests whether a given function is a k-direct-sum for odd constant values of k:

Algorithm 1 $T_{assembled-k-direct-sum}$.

pick *one of the following options uniformly:*
  Test whether δF is a $(k+1)$-direct-sum using a known test for even sized sets[a].
  pick $m \in X(k+1)$ *randomly:*
    Check whether $F|_m$ is a k-direct-sum.

[a] Note that $k+1$ is even and, whenever the known test asks to query $\delta F(a)$, the algorithm queries every set in $\binom{a}{k}$ and returns the sum of the results.

In order to analyze this test, it would first be useful to deconstruct F into three functions $F = D + Z + G$ where D is a k-direct-sum, $\delta Z = \mathbb{0}$, and the remainder G.

Bounding the Norm of G Using Co-Systolic expansion

First we will show that the rejection probability of step (2) bounds $\|G\|$ from above. In order to do so we must first consider the following two properties of δ:

- δ is linear.
- If D is a k-direct-sum (and k is odd) then δD is a $(k+1)$-direct-sum (See Lemma 28).

Combining these properties with the fact that the complex is a co-systolic expander, yields an upper bound for G. Specifically: Because $\delta F = \delta D + \delta G$, the test performed in step (2) gives an upper bound to $\|\delta G\|$ (since δD is a direct sum) and co-systolic expansion implies that $\|G\| \leq \epsilon \|\delta G\|$.

Bounding the Norm of Z Using Colorful Expansion

Secondly, we will show how to bound $\|Z\|$ from above. Alas, step (3) does not bound $\|Z\|$ from above unconditionally, but if we assume that $G = \mathbb{0}$, we can bound $\|Z\|$ from above using the rejection probability of step (3). We do that in two steps: The first is noting that $\|Z\|$ is bounded from above by all the $(k+1)$-faces that Z "touches", namely $\{m \in X(k+1) | Z|_m \neq \mathbb{0}\}$ due to a property of the norm (Lemma 10). We then show the following property: Step (3) rejects every $(k+1)$-dimensional face m on which $Z|_m \notin \{\mathbb{0}, \mathbb{1}\}$. In expander graphs, given a set of vertices S, the set of edges that are going out of S bounds from above the edges that connect two vertices within S. Similarly in higher dimensional colorful expanders, the set of edges that stay within a set S is bounded from above by the set of edges that are going out of S. We think of an edge m on which $Z|_m = \mathbb{1}$ as an edge that connects vertices within S, and an edge m on which $Z|_m \notin \{\mathbb{0}, \mathbb{1}\}$ as an edge that is going out of S. Thus by colorful expansion we can bound $\|\{m \in X(k+1) | Z|_m = \mathbb{1}\}\|$ using $\|\{m \in X(k+1) | Z|_m \notin \{\mathbb{0}, \mathbb{1}\}\}\|$. We conclude that $\|Z\|$ can be bounded from above as follows:

$$\begin{aligned}
\|Z\| \leq \|\{m \in X(k+1) | \quad Z|_m \neq \mathbb{0}\}\| = \\
\|\{m \in X(k+1) | \quad Z|_m \notin \{\mathbb{0}, \mathbb{1}\}\}\| + \|\{m \in X(k+1) | \quad Z|_m = \mathbb{1}\}\| \leq \\
\|\{m \in X(k+1) | \quad Z|_m \notin \{\mathbb{0}, \mathbb{1}\}\}\| + c\,\|\{m \in X(k+1) | \quad Z|_m \notin \{\mathbb{0}, \mathbb{1}\}\}\| = \\
(1+c)\,\|\{m \in X(k+1) | \quad Z|_m \notin \{\mathbb{0}, \mathbb{1}\}\}\|
\end{aligned}$$

which is bounded from above by the probability that step (3) rejects.

We end the proof by showing a way to combine both bounds. Note this is not trivial since the bound on $\|Z\|$ is dependent on the fact that $G = \mathbb{0}$. However, we can mitigate for this dependency, since G can be bounded independently of Z.

2 Preliminaries

▶ **Notation 2.** *Given a set S and an integer k denote by $\binom{S}{k} = \{s \subseteq S | \quad |s| = k\}$.*

2.1 Simplicial Complexes

We are now going to provide formal definitions of simplicial complexes and a norm on them:

▶ **Definition 3** (Simplicial complex). *A simplicial complex is a pair $X = (V, E)$ such that: $E \subseteq P(V)$, and if $F \in E$ then every $F' \subseteq F$ is in E as well. Elements in the set E are called the faces of X.*

▶ **Definition 4** (Dimension of a face). *Let m be a face in X. Define the dimension of m to be:*

$$dim(m) := |m| - 1$$

Also, define the set $X(i)$ to be the set of all faces of dimension i (note that $X(-1) = \{\emptyset\}$).

▶ **Notation 5.** *Let X be a d-dimensional simplicial complex, given $-1 \leq i < j \leq d$, a function $F : X(i) \to \{0, 1\}$, and $m \in X(j)$. Denote by $F|_m$ the function $F|_m : \binom{X(j)}{i+1} \to \{0, 1\}$ such that $\forall q \in X(i) : F|_m(q) = F(q)$.*

▶ **Definition 6** (Dimension of a simplicial complex). *Let $X = (V, F)$ be a simplicial complex. Define the dimension of X to be:*

$$dim(X) := \max_{f \in F} dim(f)$$

▶ **Definition 7** (Pure simplicial complex). *A d-dimensional simplicial complex X is called pure if all of its maximal faces are of dimension d.*

▶ **Definition 8** (Norm over the faces). *Let X be a pure simplicial complex of dimension d. Define the weight of the face a to be:*

$$w(a) = \frac{|\{F \in X(d) | a \subseteq F\}|}{\binom{d+1}{|a|} \cdot |X(d)|}$$

and the norm $\|.\| = \|.\|^k : P(X(i)) \to [0, 1]$ to be: $\|A\| := \sum_{a \in A} w(A)$.

We will show in Appendix A that w defines a distribution on every dimension where the probability of a face to be chosen is equal to its norm. For the rest of the paper, when an algorithm chooses a face (unless a distribution is explicitly specified), it chooses a face with the distribution implied by w.

▶ **Definition 9** (Container). *Let X be a d-dimensional simplicial complex, let $-1 \leq i \leq r \leq d$ and let $A \subseteq X(i)$. Define $\Gamma^r(A) := \{a \in X(r) | \exists b \in A : b \subseteq a\}$.*

▶ **Lemma 10.** *Let X be a d-dimensional simplicial complex, and let $-1 \leq i \leq j \leq d$. Then for any $A \subseteq X(i)$:*

$$\|A\| \leq \left\|\Gamma^j(A)\right\| \leq \binom{j+1}{i+1} \|A\|$$

▶ **Lemma 11.** *Let* $A \subseteq X(i)$*:* $\forall j : \left\| \left\{ A' \in X(i+j) \middle| \binom{A'}{i} \subseteq A \right\} \right\| \leq \|A\|$

The proofs of Lemma 10 and Lemma 11 can be found in Appendix B.

▶ **Notation 12.** *Given a complex* X*, and a test* T *whose random choice is some* $m \in X$*, denote the result of the test* T *when testing the function* F *and the random face chosen is* m *from the complex* X *by* $T_X^F(m)$*.*

2.2 Co-systolic Expansion

We will now present the first notion of expansion used in this paper, namely - co-systolic expanders. Co-systolic expansion was introduced by Evra and Kaufman in [7] and is the more topological notion of expansion we will use in this paper. In order to define this notion of expansion we must first define some spaces and operators over simplicial complexes:

▶ **Definition 13** (Co-chains)**.** *Let* X *be a simplicial complex, define the* i*-co-chains of* X *to be* $C^i(X) = \{0,1\}^{X(i)}$*.*

Note that the norm defined in Definition 8 implies a norm on the co-chains by setting the norm of a co-chain to be the norm of set of faces on which it returns 1. Formally:

▶ **Definition 14** (Extension of the norm to co-chains)**.** *For every* $C \in C^i(X)$ *define:*

$$\|C\| := \|\{a \in X(i) | C(a) = 1\}\|$$

Now that we have defined the co-chains and a norm on them, we can also define the distance between co-chains as well as the distance of a co-chain from the k-direct-sums.

▶ **Definition 15** (Distance between co-chains)**.** *Given* $C_1, C_2 \in C^k(X)$*, the distance between* C_1 *and* C_2 *is:*

$$dist(C_1, C_2) = \|C_1 + C_2\|$$

▶ **Definition 16.** *We define the distance of a co-chain* $C \in C^k(X)$ *to the* k*-direct-sum to be:*

$$\min_{D \in \{k\text{-}direct\text{-}sum\}} \{dist(C, D)\}$$

▶ **Definition 17** (Co-boundary operator)**.** *Let* $\delta_i : C^i(X) \to C^{i+1}(X)$ *be the following function:*

$$\delta_i(F)(m) = \sum_{q \in \binom{m}{i-1}} F(q)$$

Note that $F : X(i) \to \{0,1\}$ *and* $m \in X(i+1)$*.*

Lastly we will define two more spaces over the faces of the simplicial complex:

▶ **Definition 18** (Co-cycles and co-boundaries)**.** *Let* X *be a simplicial complex, define the following spaces:*

- *The* i*-co-cycles:* $Z^i(X) = Ker(\delta_i) = \left\{ Z \in C^i(X) \middle| \delta_i Z = \mathbb{0} \right\}$*.*
- *The* i*-co-boundaries:* $B^i(X) = Im(\delta_{i-1}) = \left\{ B \in C^i(X) \middle| \exists B' \in C^{i-1}(X) : B = \delta_{i-1} B' \right\}$*.*

▶ **Fact 19.** *For every dimension* i*:* $B^i(X) \subseteq Z^i(X) \subseteq C^i(X)$*.*

A complex X is an (ϵ, μ)-co-systolic expander if any i-co-chain that is far from being a co-cycle "touches" an odd number of times many $(i+1)$-co-chains. In addition to that, any co-cycle that is not a co-boundary must be large. Formally:

▶ **Definition 20** (Co-systolic expander). *Let X be a d-dimensional simplicial complex and let $\epsilon, \mu > 0$. X is an (ϵ, μ)-co-systolic-expander if for every $i = 0, 1, ..., d-1$:*

$$exp^i(X) = \min\left\{ \frac{\|\delta_i(f)\|}{\min_{z \in Z^i(X)} \{\|f + z\|\}} \middle| f \in C^i(X) \setminus Z^i(X) \right\} \geq \epsilon$$

and

$$syst^i(X) = \min\left\{ \|z\| \middle| z \in Z^i(X) \setminus B^i(X) \right\} \geq \mu$$

Note that $\min_{z \in Z^i(X)} \{\|f + z\|\}$ is the distance of f from being a co-cycle.

This notion of expansion implies that the simplicial complex has the topological overlapping property (which is explained in detail in [7]). In this paper, we will use this definition of expansion in order to estimate the non-co-cyclic part of the difference between the function given to us and its closest k-direct-sum. We will do that by first applying the co-boundary operator to the function given to us, and then test whether the result is a $(k+1)$-direct-sum (we will see why this suffices in section 3).

2.3 Colorful Expansion

The other form of high dimensional expansion we use is a combinatorial one. It was first introduced by Kaufman and Mass in [10]. This notion of expansion considers every face on which the i-co-chain is equal to 1 as if it is colored in one color, and every face on which the i-co-chain is equal to 0 as if is it colored in a different color. Then we look at all the $(i+1)$-faces that are not monochromatic. More formally:

▶ **Definition 21** (Colorful Operator). *Let $c_i : C^i(X) \to C^{i+1}(X)$ be the following function:*

$$c_i(F)(m) = \begin{cases} 1 & \exists a, b \in \binom{m}{k-1} : F(a) = 1 \text{ and } F(b) = 0 \\ 0 & \text{otherwise} \end{cases}$$

Note that $F : X(i) \to \{0, 1\}$ and $m \in X(i+1)$.

A simplicial complex is a colorful expander if every sufficiently small i-co-chain implies a lot of non-monochromatic $(i+1)$-faces. Formally:

▶ **Definition 22** (Colorful Expander). *Let X be a d-dimensional simplicial complex. We say that X is a σ-colorful-expander if for any $W \in C^i(X)$ $(0 \leq i < d)$ such that $\|W\| \leq 0.5$:*

$$\frac{\|c_i(W)\|}{\|W\|} \geq \sigma$$

This notion of expansion deals with random walks - consider the random walk that moves between two i-faces through a common $(i+1)$-face that contains them both. In [10] it was shown that such random walks converge rapidly to the stationary distribution. In this paper we will use this notion of expansion in order to estimate the co-cyclic part of the difference between the function given to us and its closest k-direct-sum (which would be impossible to do using the other notion of expansion).

3 Properties of Direct Sums

We will now present what the k-direct-sums are and show some useful properties of k-direct-sums.

▶ **Definition 23** (k-direct-sum). *A co-chain $D : X(k-1) \to \{0,1\}$ is called a k-direct-sum if there is some function $d : X(0) \to \{0,1\}$ such that $D(a) = \sum_{v \in a} d(v)$ (The sum is performed modulo 2).*

▶ **Definition 24** (Origin function). *Let $D : X(k) \to \{0,1\}$ be a k-direct-sum. An origin function of D is any function $d : X(0) \to \{0,1\}$ such that $D(a) = \sum_{v \in a} d(v)$.*

In the rest of this chapter we will explore properties of the k-direct-sums. We will start by finding a set of functions that spans the k-direct-sums. Then we will use these functions in order to show how direct-sums behave when applying the co-boundary operator to them. We will start by showing that the set of k-direct-sums is linear:

▶ **Lemma 25** (Direct sums are closed under addition). *Let F and G be two k-direct-sums whose origin functions are f and g respectively then $F + G$ is a k-direct-sum and its origin function is $f + g$.*

Proof. We know that $F(a) = \sum_{b \in a} f(b)$ and $G(a) = \sum_{b \in a} g(b)$. It is easy to see that $F + G = \sum_{b \in a} f(b) + \sum_{b \in a} g(b) = \sum_{b \in a} f(b) + g(b) = \sum_{b \in a} (f + g)(b)$. Therefore $F + G$ is a k-direct-sum and $f + g$ is its origin function. ◀

We will now wish to find a set of functions that spans the k-direct-sum so:

▶ **Definition 26** (Spanning set of the k-direct-sums). *Let $u \in X(0)$. Define $H_u^k : X(k-1) \to \{0,1\}$ to be:*

$$H_u^k(a) = \begin{cases} 1 & \text{if } u \in a \\ 0 & \text{otherwise} \end{cases}$$

One can easily check that $\forall k : H_u^k$ is a k-direct-sum whose origin function is:

$$h_u^k(v) = \begin{cases} 1 & \text{if } v = u \\ 0 & \text{otherwise} \end{cases}$$

We can now prove that $\left\{H_u^k\right\}$ spans the set of k-direct-sums:

▶ **Lemma 27.** *The set of k-direct-sums is spanned by $\left\{H_u^k | u \in X(0)\right\}$*

Proof. Let F be a k-direct-sum. By definition there exists $f : X(0) \to \{0,1\}$ such that $F(a) = \sum_{b \in a} f(b)$. Consider the support of f: $\sup(f) = \{u \in X(0) | f(u) = 1\}$, and define $G = \sum_{u \in \sup(f)} H_u^k$. It is easy to see that $F(a) = \sum_{b \in a} f(b) = \sum_{b \in a} H_u^k(b)$ and therefore $F \in span\left\{H_u^k | u \in X(0)\right\}$.
Let $F \in span\left\{H_u^k \middle| u \in X(0)\right\}$ therefore there exists some set $I \subseteq X(0)$ such that $F = \sum_{u \in I} H_u^k$. We know that $\left\{H_u^k\right\}_{u,k}$ are k-direct-sums, therefore F is a sum of k-direct-sums and, due to Lemma 25, F is a k-direct-sum as well. ◀

We will now show a connection between the k-direct-sums in the odd dimensions and the k-direct-sums in the even dimensions:

▶ **Lemma 28.** *For odd values of k: $\delta_k H_u^k = H_u^{k+1}$*

Proof.

$$\delta_k H_u^k(a) = \sum_{\substack{b\subset a \\ |b|=|a|-1}} H_u^k(B) = |\{b|b\subset a, |b| = |a|-1, u\in b\}| = \begin{cases} \binom{k}{k-1} & \text{if } v_i \in a \\ 0 & \text{otherwise} \end{cases} =$$

$$\begin{cases} k & \text{if } u\in a \\ 0 & \text{otherwise} \end{cases} = \begin{cases} 1 & \text{if } u\in A \\ 0 & \text{otherwise} \end{cases} = H_u^{k+1}$$ ◀

▶ **Lemma 29.** *For odd values of k, if F is k-direct-sum then δF is a $(k+1)$-direct-sum.*

Proof. F is a k-direct-sum therefore there exists some $I \subseteq X(0)$ such that $F = \sum_{u\in I} H_u^k$. And thus $\delta F = \delta(\sum_{u\in I} H_u^k) = \sum_{u\in I} \delta H_u^k = \sum_{u\in I} H_u^{k+1} \in span\left\{H_u^{k+1}|u\in X(0)\right\}$ and δF is a $(k+1)$-direct-sum. ◀

▶ **Lemma 30.** *For even values of k, if F is a k-direct-sum then $F \in B^{k+1}(X) \subseteq Z^{k-1}(X)$.*

Proof. F is a k-direct-sum therefore there exists $I \subseteq X(0)$ such that $F = \sum_{u\in I} H_u^k = \sum_{u\in I} \delta H_u^{k-1}$. Finally we get that $F \in B^{k-1}(X) \subseteq Z^{k-1}(X)$. ◀

Note that the previous two Lemmas imply that Lemma 29 is true for any value of k.

4 Definition of Components Appearing in the Tester

In this section, we will provide some definitions that will help us build the test for the k-direct-sum problem.
We would first like to define a relaxed version of the k-direct-sum, namely the k-co-cycle-indifferent-direct-sum:

▶ **Definition 31** (Co-cycle indifferent direct sum)**.** *Define the property of being a k-co-cycle-indifferent-direct-sum to be:*

$$CI = \left\{F = D + Z \middle| D \text{ is a } k\text{-direct-sum and } Z \in Z^{k-1}(X)\right\}$$

In section 6 will show that this property is testable for odd values of k.
We would also want to define a separator which helps in separating k-direct-sums from non k-direct-sums. Unlike tests, in which the rejection probability is linear in the distance from the property, separators reject with (at least) constant probability when their input is not in the property.

▶ **Definition 32** (Direct sum separator)**.** *Let X be a simplicial complex. An algorithm T is called an (n,k,Q,η)-direct-sum-separator if, for the complete complex on $n-1$ nodes (denoted by X_{n-1}), when given $f \in C^{k+1}(X_{n-1})$, the following applies:*

- *If f is a k-direct-sum then $Pr[T^f = 1] = 1$.*
- *If f is not a k-direct-sum then $Pr[T^f = 0] \geq \eta$.*
- *T queries f on at most Q faces in $X_{n-1}(k-1)$.*

In appendix C we will show an explicit separator whose error probability is 0 and queries the entire complex. We will also show how to construct a separator from a test. It is important to note that one can reduce the query complexity of the test presented in this paper by providing a different separator with lower query complexity (using, for example, Lemma 54).

5 Presenting A Test for Being a k-direct-sum

In this section, we will prove the main theorem. But first recall the definition of a (Q, E)-test for being a k-direct-sum:

▶ **Definition 33** ((Q, E)-test for being a k-direct-sum). *A (Q, E)-test for being a k-direct-sum is an algorithm that:*

- *Queries F on Q inputs from X.*
- *Accepts k-direct-sums.*
- *Rejects with probability of at least ξ every function whose distance from the k-direct-sums is at least $E\xi$.*

▶ **Theorem 34** (Main Theorem). *Let X be a d-dimensional pure simplicial complex, and $0 < k \leq d-2$ be an odd constant. Also assume there exists a (Q, E)-test for being a $(k+1)$-direct-sum on X and let $F : X(k-1) \to \{0,1\}$ be a function. Then, if X is an (ϵ, μ)-co-systolic expander and a σ-colorful-expander, there exists a test T such that:*

- *T queries F a maximum of $\max\left\{(k+1)\cdot Q, \binom{k+2}{k}\right\}$ times.*
- *F is a k-direct-sum $\Leftrightarrow \Pr[T \text{ accepts } F] = 1$.*
- *If $\Pr[T \text{ rejects } F] \leq \xi$ then there exists a k-direct-sum F' such that*

$$dist(F, F') \leq \left(\left(1+\frac{1}{\sigma}\right)\left(\binom{k+2}{k}\frac{E}{\epsilon}+1\right)+\frac{E}{\epsilon}\right)\xi.$$

As a corollary we show that the k-direct-sum problem on the complete complex is testable with $O(k^2)$-queries for odd k.

▶ **Corollary 35** (k-direct-sum is testable on the complete complex for odd k's). *On the complete complex there exists a $(O(k^2), E)$-test for being a k-direct-sum where E is constant and k is odd.*

The proof of this corollary will be presented in Appendix C. We also show that:

▶ **Corollary 36.** *For any dimension d, there exists a family of **bounded** degree simplicial complexes X such that the property of k-direct-sum is testable on X.*

Proof. We will show that Ramanujan complexes satisfy the conditions of Theorem 34:

- In [7] it was shown that for any dimension d there exists q_0, such that for any prime power $q > q_0$, there are $\mu = \mu(d)$ and $\epsilon = \epsilon(d, q)$ such that if X is the the d-dimensional complex induced by a q-thick Ramanujan complex then X is an (ϵ, μ)-co-systolic expander.
- In addition to that in [10] it was proven that for any dimension d, there exists a constant $q_0' = q_0'(d)$ such that, if X is a d-dimensional q'-thick Ramanujan complex for $q' > q_0'$, then there are $\sigma = \sigma(d, q')$ such that X is a σ-colorful expander.

We end this proof by noting that it was shown in [11] that there is an explicit construction of Ramanujan complexes (and therefore there is an explicit construction for complexes that are both co-systolic expanders and colorful expanders). ◀

We will prove the main theorem using a (Q_{CI}, ζ)-test for the k-co-cycle-indifferent-direct-sum problem called T_{CI} and a $(k+2, k, Q_{sep}, \eta)$-direct-sum-separator T_{sep}. Specifically, we will prove that the following is a tester for the k-direct-sum problem:

Algorithm 2 $T_{direct-sum}$.

```
pick one of the following options uniformly:
    Run T_CI and return its result.
    pick m ∈ X(k+1) randomly:
        Run T_sep on m with F|_m and return its result.
```

Formally we will prove that:

▶ **Theorem 37** (k-direct-sums are testable). *On any complex that is a σ-colorful-expander and for any constant odd value of k, given:*

- *T_{sep} - A $(k+2, k, Q_{sep}, \eta)$-direct-sum-separator for the complete complex.*
- *T_{CI} - A (Q_{CI}, ζ)-test for the k-co-cycle-indifferent-direct-sum.*

We can construct $T_{direct-sum}$ as shown above such that $T_{direct-sum}$ is a:

$$\left(\max\left\{Q_{CI}, Q_{sep}\right\}, \left(1+\frac{1}{\sigma}\right)\left(\frac{1}{\eta}+\binom{k+2}{k}\zeta\right)+\zeta\right)\text{-test}$$

for the k-direct-sum problem.

In order to understand why the test works, consider a deconstruction of F into three parts: $F = D + Z + G$. In this deconstruction we assume that:

- G is minimal with regards to the k-co-cycle-indifferent-direct-sum.
- Z is the minimal co-cycle with regards to the k-direct-sum problem.
- D is a k-direct-sum.

In sub-section 5.1 we will show that the rejection probability of step (2) bounds from above $\|G\|$. In sub-section 5.2 we will show that, when ignoring G, step (3)'s rejection probability bounds $\|Z\|$ from above. Finally in sub-section 5.3 we will show how the combination of both steps provides a test for being a k-direct-sum. Note that unlike step (2) (in which there is no assumption on Z), the analysis of step (3) assumes that $G = \mathbb{0}$.

5.1 Step (2) of the test estimates the Norm of G

▶ **Lemma 38.** *Let T_{CI} be a (Q_{CI}, ζ)-test for the k-co-cycle-indifferent-direct-sum then:*

$$\|G\| \leq \zeta \cdot \Pr\left[\text{step (2) rejects}\right]$$

Proof. $\|G\| = dist(F, CI) \leq \zeta \cdot \Pr\left[T_{CI} = 0\right] = \zeta \cdot \Pr\left[\text{step (2) rejects}\right]$ The second inequality holds due to the definition of T_{CI}. ◀

5.2 Step (3) of the Test Estimates Norm of Z Assuming That There is No Remainder

In step (3) we pick a $(k+1)$-dimensional face randomly and then check whether the function is a k-direct sum on that specific face. In this section, we will show that the failure probability of doing so bounds $\|Z\|$ from above. We will do that by first observing that given m, a $(k+1)$-dimensional face, either $F|_m$ is not a k-direct-sum or $Z|_m \in \{\mathbb{0}, \mathbb{1}\}$:

▶ **Lemma 39.** *Let $F = D + Z$ such that D is a k-direct-sum and $Z \in Z^{k-1}(X)$ then for every odd value k and $m \in X(k+1)$: If $F|_m$ is a k-direct-sum on m then: $Z|_m \in \{\mathbb{0}, \mathbb{1}\}$.*

Proof. $F|_m$ is a k-direct-sum and, because $G = \mathbb{0}$, so is $Z|_m$ as $Z|_m = F|_m + D|_m$. Assuming that $Z|_m \notin \{\mathbb{0}, \mathbb{1}\}$, let $z : m \to \{0, 1\}$ be an origin function of Z and let $A_i = \{v \in m | z(v) = i\}$. Pick the largest possible set (of up to $k+1$ elements) of odd size out of A_1 (the set is not empty because otherwise $Z|_m = \mathbb{0}$) and name it A. Add to that set $k + 1 - |A|$ items from A_0 (which cannot be empty since $Z|_m \neq \mathbb{1}$) to form a $(k+1)$-face which we will denote as t. It is easy to see that $\delta Z|_m(t) = \sum_{v \in t} z(v) = \sum_{v \in A} z(v) = 1$ (the last equality holds because $|A|$ is odd) which contradicts the fact that $Z \in Z^{k-1}(X)$ ◀

We now observe that the set of $(k+1)$-dimensional faces on which $Z|_m \neq \mathbb{0}$ can be split into two sets:

- The set of all $m \in X(k+1)$ on which $F|_m$ is not a k-direct-sum.
- The set of all $m \in X(k+1)$ such that $Z|_m = \mathbb{1}$ (which we will denote as S).

It is easy to see that the rejection probability of step (3) bounds the first set (since step (3) fails on every face in the set). We will spend the majority of this sub-section proving that $\|S\|$ can also be bounded from above using the rejection probability of step (3). We will end this sub-section by combining the aforementioned bounds.

Before discussing how to bound $\|S\|$ from above, it will be useful to present Lemma 39 again, this time with the new terminology described above:

▶ **Corollary 40.** *Let $m \in X(k+1)$ then:*

$$F|_m \text{ is not a } k\text{-direct-sum} \Leftrightarrow m \in \Gamma^{k+1}(Z) \setminus S$$

Proof. $m \in \Gamma^{k+1}(Z)$ iff $Z|_m \neq \mathbb{0}$ (due to the definition of Γ) and $m \notin S$ iff $Z|_m \neq \mathbb{1}$ (due to the definition of S) therefore:

$$F|_m \text{ is not a } k\text{-direct-sum} \Leftrightarrow Z \notin \{\mathbb{0}, \mathbb{1}\} \Leftrightarrow m \in \Gamma^{k+1}(Z) \setminus S$$ ◀

In order to bound $\|S\|$ we will look at a different function whose norm bounds $\|S\|$ from above, specifically:

▶ **Definition 41.** *Define $E : X(k) \to \{0,1\}$ to be the following function:*

$$E(a) = \begin{cases} 1 & \text{if } Z|_a = \mathbb{1} \\ 0 & \text{otherwise} \end{cases}$$

This function helps in bounding $\|S\|$ from above because every face in S is comprised solely of k-dimensional faces on which E returns 1. Combining this fact with Lemma 11 yields that $\|S\| \leq \|E\|$.

All we have to do now is to bound $\|E\|$. This will be done by first showing that step (3) of the test rejects every non-monochromatic $(k+1)$-face (where E is considered the coloring). We will then show that $\|E\| < 0.5$ which will allow to use the colorful expansion in order to bound $\|E\|$.

▶ **Lemma 42** (Step (3) Fails on the Non-Monochromatic Faces)**.** *Let $m \in Z(k+1)$. If $c(E)(m) = 1$ then $F|_m$ is not a k-direct-sum.*

Proof. $c(E)(m) = 1 \Rightarrow \exists a, b \in \binom{m}{k+1} : E(a) = 1$ and $E(b) = 0$. Using the definition of E we get that:

- $E(b) = 0 \Rightarrow \exists c \in \binom{b}{k} : Z(c) = 0$
- $E(a) = 1 \Rightarrow \forall t \in \binom{a}{k} : Z(t) = 1$

Therefore $Z|_m \notin \{\mathbb{0}, \mathbb{1}\}$ and $F|_m$ is not a k-direct-sum (Lemma 39). ◀

▶ **Lemma 43.** *For every function of the form $F = D + Z + G$ it holds that $\|Z\| \leq 0.5$ (Note that this lemma is true even if $G \neq 0$).*

Proof. It is easy to see that the function $f(v) = 1$ is the origin function of $\mathbb{1}$ and therefore $\mathbb{1}$ is a k-direct-sum. Now, assuming that $\|Z\| > 0.5$ we conclude that $\|\mathbb{1} + Z\| \leq 0.5$ and $(\mathbb{1} + D) + (\mathbb{1} + Z) = D + Z$. Also $(\mathbb{1} + D)$ is a k-direct-sum. We conclude that $\|\mathbb{1} + Z\| < \|Z\|$ and $F + G + (\mathbb{1} + Z) = \mathbb{1} + D$ which is a k-direct-sum. This contradicts the fact that Z is minimal. ◀

▶ **Corollary 44.** $\|E\| \leq 0.5$.

Proof. By the definition of E if $E(a) = 1$ then $\forall a' \in \binom{a}{k} : Z(a') = 1$. Using Lemma 11 yields that $\|E\| \leq \|Z\|$ which finishes the proof. ◀

We are now finally ready to bound E using the colorful expansion of X:

▶ **Lemma 45** (Estimating E)**.** *On every σ-colorful expander X:*

$$\|E\| \leq \frac{1}{\sigma} \|\{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\}\|$$

Proof. X is a colorful expander and $\|E\| \leq 0.5$ therefore $\sigma \leq \frac{\|c(E)\|}{\|E\|}$ which in turn means that:

$$\sigma \|E\| \leq \|c(E)\| \leq \|\{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\}\|$$

(the second inequality is due to Lemma 42) and therefore:

$$\|E\| \leq \frac{1}{\sigma} \|\{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\}\|$$ ◀

▶ **Lemma 46** (Estimating Z)**.** *Let X be a σ-colorful-expander, and let F be a function of the form $F = D + Z$ such that D is a k-direct-sum and Z is a co-cycle then:*

$$\|Z\| \leq \frac{1}{\eta}\left(1 + \frac{1}{\sigma}\right) \Pr\left[\text{step (3) rejects}\right]$$

Proof.

$$\begin{aligned}
\|Z\| \leq \left\|\Gamma^{k+1}(Z)\right\| \leq \left\|\left(\Gamma^{k+1}(Z) \setminus S\right) \cup S\right\| = \left\|\Gamma^{k+1}(Z) \setminus S\right\| + \|S\| \leq \\
\left\|\Gamma^{k+1}(Z) \setminus S\right\| + \|E\| \leq \left\|\Gamma^{k+1}(Z) \setminus S\right\| + \frac{1}{\sigma}\left\|\Gamma^{k+1}(Z) \setminus S\right\| = \\
\left(1 + \frac{1}{\sigma}\right)\left\|\Gamma^{k+1}(Z) \setminus S\right\| = \\
\left(1 + \frac{1}{\sigma}\right)\|\{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\}\|
\end{aligned}$$

Note that the inequality found at the end of the first row is due to Lemma 11, the inequality in the second row is due to Lemma 45 and the last equality is due to Corollary 40.
Note that:

$$\begin{aligned}
&\Pr\left[\text{step (3) rejects}\right] = \\
&\quad \Pr\left[F|_m \text{ is not a } k\text{-direct-sum}\right] \cdot \Pr\left[T_{sep} \text{ rejects} | \quad F|_m \text{ is not a } k\text{-direct-sum}\right] = \\
&\quad \|\{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\}\| \cdot \eta
\end{aligned}$$

All the probabilities are over a choice of $m \in X(k+1)$.
We conclude by noting that this yields that:

$$\|Z\| \leq \frac{1}{\eta}\left(1 + \frac{1}{\sigma}\right) \Pr\left[\text{step (3) rejects}\right]$$ ◀

5.3 Combining the Estimations

Now that we know how to estimate both $\|G\|$ and $\|Z\|$ (with the assumption that $G = \mathbb{0}$), it is finally time to combine both estimations in order to estimate $\|Z + G\|$. Note that our estimation of $\|Z\|$ is dependent on our estimation of $\|G\|$. We will deal with this dependency by bounding the interference of G using our estimation of it. We will then estimate $\|Z\|$ as if wherever G would have interfered, step (3) rejected.

▶ **Lemma 47.** *Let $F = D+Z+G$ such that D is a k-direct-sum, Z is a $(k-1)$-co-cycle and G is the remainder. Then if $\Pr\left[T^F_{direct-sum} \text{ rejects}\right] \leq \xi$ then $\|Z\| \leq \left(1 + \frac{1}{\sigma}\right)\left(\frac{1}{\eta} + \binom{k+2}{k}\zeta\right)\xi$.*

Proof. First note that because $\Pr\left[T^F_{direct-sum} \text{ rejects}\right] \leq \xi$ we know that $\Pr\left[\text{step (2) rejects } F\right] \leq \xi$ and $\Pr\left[\text{step (3) rejects } F\right] \leq \xi$. Also, consider what happens when we run the test on $F' = D + Z$. Note that on F' the bound found in Lemma 46 holds. Also note the the co-cyclic part of F and F' is Z. Therefore if we could bound the rejection probability of step (3) on F' using the rejection probability of steps (2) and (3) on F we would have a bound for $\|Z\|$. We will start by bounding the set of $(k+1)$-faces on which F' is not a k-direct-sum:

$$\begin{aligned}
&\{m \in X(k+1) | \quad F'|_m \text{ is not a } k\text{-direct-sum}\} \subseteq \\
&\quad \{m \in X(k+1) | \quad (F'+G)|_m \text{ is not a } k\text{-direct-sum and } G|_m = \mathbb{0}\} \cup \\
&\quad \{m \in X(k+1) | \quad G|_m \neq \mathbb{0}\} = \\
&\quad \{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\} \cup \{m \in X(k+1) | \quad G|_m \neq \mathbb{0}\} = \\
&\quad \{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\} \cup \Gamma^{k+1}(G)
\end{aligned}$$

Knowing this, we get that:

$$\begin{aligned}
&\Pr\left[\text{step (3) rejects } m \text{ when testing } F'\right] = \\
&\quad \eta \left\|\{m \in X(k+1) | \quad F'|_m \text{ is not a } k\text{-direct-sum}\}\right\| \leq \\
&\quad \eta \left\|\{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\}\right\| + \eta \left\|\Gamma^{k+1}(G)\right\|
\end{aligned}$$

Using Lemma 38, we know that $\|G\| \leq \zeta \cdot \Pr\left[\text{step (2) rejects}\right]$ and therefore, using Lemma 10 we get that $\left\|\Gamma^{k+1}(G)\right\| \leq \binom{k+2}{k}\|G\| \leq \binom{k+2}{k}\zeta \cdot \Pr\left[\text{step (2) rejects}\right]$. Therefore:

$$\begin{aligned}
&\Pr\left[\text{step (3) rejects } m \text{ when testing } F'\right] \leq \\
&\quad \eta \left\|\{m \in X(k+1) | \quad F|_m \text{ is not a } k\text{-direct-sum}\}\right\| + \eta \left\|\Gamma^{k+1}(G)\right\| \leq \\
&\quad \Pr\left[\text{step (3) rejects } m \text{ when testing } F\right] + \binom{k+2}{k}\eta\zeta \cdot \Pr\left[\text{step (2) rejects}\right] \leq \\
&\quad \left(1 + \binom{k+2}{k}\eta\zeta\right)\xi
\end{aligned}$$

We will now use the bound obtained in Lemma 46 on F' which would yield:

$$\eta\frac{1}{1+\frac{1}{\sigma}}\|Z\| \leq \left(1 + \binom{k+2}{k}\eta\zeta\right)\xi \Rightarrow \|Z\| \leq \left(1 + \frac{1}{\sigma}\right)\left(\frac{1}{\eta} + \binom{k+2}{k}\zeta\right)\xi \qquad \blacktriangleleft$$

We are now finally ready to prove Theorem 37:

Proof of Theorem 37. First, consider the number of queries performed by $T_{direct-sum}$. If step (2) is chosen then $T_{direct-sum}$ performs Q_{CI} queries and if step (3) is chosen then $T_{direct-sum}$ performs Q_{sep} queries. Therefore $T_{direct-sum}$ performs, at most, $\max\{Q_{CI}, Q_{sep}\}$ queries.

Suppose that $\Pr\left[T^F_{direct-sum} \text{ rejects}\right] \leq \xi$ then, using Lemma 38 and Lemma 47 we get that:

$$\|Z+G\| \leq \|Z\| + \|G\| \leq \left(1+\frac{1}{\sigma}\right)\left(\frac{1}{\eta} + \binom{k+2}{k}\zeta\right)\xi + \zeta\xi = \left(\left(1+\frac{1}{\sigma}\right)\left(\frac{1}{\eta} + \binom{k+2}{k}\zeta\right) + \zeta\right)\xi$$

Now all that is left to prove is that a k-direct-sum will always pass the test. If step (2) is chosen then, because a k-direct-sum is also a k-co-cycle-indifferent-direct-sum, the test will always accept. Otherwise, if step (3) is chosen then, because the function is a k-direct-sum, it will be a k-direct-sum on any sub-complex of dimension $k+1$ and therefore step (3) will always accept as well. ◀

We can now prove the main theorem using Theorem 37:

Proof of Theorem 34. Combining Lemma 49 and Lemma 53 we get that there exists a $\left(\max\left\{(k+1)\cdot Q, \binom{k+2}{k}\right\}, \left(\left(1+\frac{1}{\sigma}\right)\left(\binom{k+2}{k}\frac{E}{\epsilon}+1\right)+\frac{E}{\epsilon}\right)\right)$-test for k-direct-sum for odd values of k. ◀

6 Providing a Test for Being a k-co-cycle-indifferent-direct-sum

In this section we will show how to obtain a test for being a k-co-cycle-indifferent-direct-sum using a test for being a $(k+1)$-direct-sum. We will do that by considering the expansion of k-direct sums under co-systolic expansion.

▶ **Lemma 48.** *For any function $F = D+Z+G$ (G is minimal) on an ϵ-co-systolic expander: $\|G\| \leq \frac{1}{\epsilon} dist(\delta F, k\text{-direct-sum})$*

Proof. First note that $\delta F = \delta D + \delta Z + \delta G = \delta D + \delta G$. In addition, because the complex is an ϵ-co-systolic expander and G is minimal: $\|G\| \leq \frac{1}{\epsilon}\|\delta G\|$. Also $\|\delta G\| = dist(\delta F, k\text{-direct-sum})$ and therefore $\|G\| \leq \frac{1}{\epsilon} dist(\delta F, k\text{-direct-sum})$. ◀

We are now ready to provide the actual test:

▶ **Lemma 49.** *Let X be an (ϵ,μ)-co-systolic-expander. If there is a (Q,ξ)-test for being a $(k+1)$-direct-sum (denoted by T) on X, then there is also a $((k+1)\cdot Q, \frac{\xi}{\epsilon})$-test for being a k-co-cycle-indifferent-direct-sum on X.*

Proof. Consider the following test:

Algorithm 3 T_{CI}.

1 Return the result of T on δF (whenever T queries δF, calculate it and send the result).

It is easy to see that:

$$dist(F, k\text{-co-cycle-indifferent-direct-sum}) = \|G\| \leq \frac{1}{\epsilon} dist(\delta F, k\text{-direct-sum}) \leq \frac{\xi}{\epsilon}\Pr\left[T_{CI}=0\right]$$

Also, F is a k-co-cycle-indifferent-direct-sum $\Leftrightarrow G = \mathbb{0} \Leftrightarrow \Pr\left[T_{CI} \text{ accepts } F\right] = 1$
For any query T makes, T_{CI} makes $\binom{k+1}{k} = (k+1)$ queries and therefore T_{CI} performs at most $(k+1)\cdot Q$ queries. ◀

References

1 Mihir Bellare, Don Coppersmith, JOHAN Hastad, Marcos Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996.

2 Eli Ben-Sasson, Madhu Sudan, Salil Vadhan, and Avi Wigderson. Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 612–621. ACM, 2003.

3 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of computer and system sciences*, 47(3):549–595, 1993.

4 Roee David, Irit Dinur, Elazar Goldenberg, Guy Kindler, and Igor Shinkar. Direct sum testing. *SIAM Journal on Computing*, 46(4):1336–1369, 2017.

5 Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.

6 Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. *Electronic Colloquium on Computational Complexity (ECCC)*, 2017.

7 Shai Evra and Tali Kaufman. Bounded degree cosystolic expanders of every dimension. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 36–48. ACM, 2016.

8 Eldar Fischer. The art of uninformed decisions: A primer to property testing. In *Current Trends in Theoretical Computer Science: The Challenge of the New Century Vol 1: Algorithms and Complexity Vol 2: Formal Models and Semantics*, pages 229–263. World Scientific, 2004.

9 Tali Kaufman and Alexander Lubotzky. High dimensional expanders and property testing. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 501–506. ACM, 2014.

10 Tali Kaufman and David Mass. High Dimensional Combinatorial Random Walks and Colorful Expansion. In *ITCS*, 2017.

11 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Ramanujan complexes of typeÃ$_d$. *Israel Journal of Mathematics*, 149(1):267–299, 2005.

A Sampling According to the Norm

In this section we will show how to pick a face with probability that equals to its norm. Consider the following sampling algorithm:

Algorithm 4 *Sample*(l, r).

1 **pick** *uniformly (using the random bits from r)* $m \in X(d)$.
2 **while** $|m| > l + 1$ **do**
3 | **pick** *uniformly (using the random bits from r)* $v \in m$.
4 | $m \leftarrow m \setminus \{v\}$.
5 **end**
6 return m.

Note that steps 3 and 4 are equivalent to choosing a sub-face of m of dimension $\dim(m)-1$. We are going to requite a way to denote a specific value of m during the run of the sampling algorithm:

▶ **Definition 50.** *Given a single run of Sample*(l, r)*, define for every* $l + 1 < i < d + 1$*:* M_i^r *to be the value of* m *when* $|m| = i$ *and the random bits chosen by the algorithm are* r*.*

It is easy to see that these sets satisfy the following properties:

- $\forall r \forall i : M_i^r \subset M_{i+1}^r$.
- $\forall i \forall a \in X(i-1) : \Pr[M_i^r = a] = \Pr[Sample(i-1, r) = a]$.

▶ **Lemma 51.** *Let X be a simplicial complex of dimension d and let $-1 \leq l \leq d$ then:*

$$\forall a \in X(l) : \Pr\left[Sample(l,r) = a\right] = w(a)$$

And also:

$$\forall A \in P(X(i)) : \|A\| = \Pr\left[Sample(i,r) \in A\right]$$

Where Sample is the algorithm 4.

Proof. We will prove this lemma using induction:
Base case: $l = d$: Notice that $\forall a \in X(d) : w(a) = \frac{1}{\binom{d+1}{d+1}|X(d)|}$.
The lemma holds because $Sample(d+1,r)$ simply chooses a face of dimension d uniformly.
Assuming that $\forall a \in X(l+1) : \Pr\left[Sample(l+1,r) = a\right] = w(a)$ we will now prove that $\forall a \in X(l) : \Pr\left[Sample(l,r) = a\right] = w(a)$ (where M_i^r are the values defined in definition 50):

$$\begin{aligned}
&\forall a \in X(l) : \Pr\left[Sample(l,r) = a\right] = \\
&\quad \sum_{b \in \{b \in X(l+1) | a \subseteq b\}} \Pr\left[Sample(l,r) = a \middle| M_{l+1}^r = b\right] \cdot \Pr\left[M_{l+1}^r = b\right] = \\
&\quad \sum_{b \in \{b \in X(l+1) | a \subseteq b\}} \frac{1}{l+2} w(b) = \sum_{b \in \{b \in X(l+1) | a \subseteq b\}} \frac{1}{l+2} \frac{|\{q \in X(d) | b \subseteq q\}|}{\binom{d+1}{|b|} \cdot |X(d)|} = \\
&\quad \sum_{b \in \{b \in X(l+1) | a \subseteq b\}} \frac{1}{d-l} \frac{|\{q \in X(d) | b \subseteq q\}|}{\binom{d+1}{|b|-1} \cdot |X(d)|} = \\
&\quad \frac{1}{(d-l)\binom{d+1}{|a|} \cdot |X(d)|} \sum_{b \in \{b \in X(l+1) | a \subseteq b\}} |\{q \in X(d) | b \subseteq q\}| \\
&\quad = \frac{|\{q \in X(d) | b \subseteq q\}|}{\binom{d+1}{|a|} \cdot |X(d)|} = w(a)
\end{aligned}$$

The fourth equation holds because:

$$\begin{aligned}
(l+2)\binom{d+1}{l+2} &= (l+2)\frac{(d+1)!}{(l+2)!(d-l-1)!} = \frac{(d+1)!}{(l+1)!(d-l-1)!} \\
&= (d-l)\frac{(d+1)!}{(l+1)!(d-l)!} = (d-l)\binom{d+1}{l+1}
\end{aligned}$$

The sixth equation holds because every maximal face that contains a is counted $\binom{d+1-(l+1)}{1} = d-l$ times. Finally we can see that:

$$\begin{aligned}
&\Pr\left[Sample(i,r) \in A\right] = \Pr\left[\bigvee_{a \in A} Sample(i,r) = a\right] = \sum_{a \in A} \Pr\left[Sample(i,r) = a\right] = \\
&\quad \sum_{a \in A} w(\{a\}) = \|A\|
\end{aligned}$$

◀

B Proofs of Bounds on the Norm

Proof of Lemma 10. First consider how a single face behaves under Γ^j:

$$\begin{aligned}\forall a \in X(i) : \left\|\Gamma^j(\{a\})\right\| &= \sum_{\substack{b\in X(j)\\ a\subseteq b}} w(b) = \sum_{\substack{b\in X(j)\\ a\subseteq b}} \frac{|\{q \in X(d) | b \subseteq q\}|}{\binom{d+1}{|b|}\cdot |X(d)|} \\ &= \sum_{\substack{b\in X(j)\\ a\subseteq b}} \sum_{\substack{q\in X(d)\\ b\subseteq q}} \frac{1}{\binom{d+1}{j+1}\cdot |X(d)|} = \sum_{\substack{q\in X(d)\\ a\subseteq q}} \sum_{\substack{b\in X(j)\\ a\subseteq b\subseteq q}} \frac{1}{\binom{d+1}{j+1}\cdot |X(d)|} \\ &= \sum_{\substack{q\in X(d)\\ a\subseteq q}} \frac{\binom{d-i}{j-i}}{\binom{d+1}{j+1}\cdot |X(d)|} = \frac{\binom{d-i}{j-i}\cdot |\{q \in X(d) | a \subseteq q\}|}{\binom{d+1}{j+1}\cdot |X(d)|} = \frac{\binom{d-i}{j-i}\cdot \binom{d+1}{i+1}}{\binom{d+1}{j+1}} w(\{a\}) \\ &= \binom{j+1}{i+1} w(\{a\})\end{aligned}$$

Note that the last equation holds because:

$$\frac{\binom{d-i}{j-i}\cdot \binom{d+1}{i+1}}{\binom{d+1}{j+1}} = \frac{\frac{(d+1)!(d-i)!}{(j-i)!(d-j)!(i+1)!(d-i)!}}{\frac{(d+1)!}{(d-j)!(j+1)!}} = \frac{(j+1)!}{(j-i)!(i+1)!} = \binom{j+1}{i+1}$$

Now one can easily check that:

$$\begin{aligned}\forall A \subseteq X(i) : \left\|\Gamma^j(A)\right\| &= \left\|\bigcup_{a\in A} \Gamma^j(\{a\})\right\| \leq \sum_{a\in A} \left\|\Gamma^j(\{a\})\right\| = \sum_{a\in A} \binom{j+1}{i+1} w(\{a\}) \\ &= \binom{j+1}{i+1} \sum_{a\in A} w(\{a\}) = \binom{j+1}{i+1} \|A\|\end{aligned}$$

The other direction can be achieved by looking at the algorithm presented in Lemma 51: Consider the set of values M_i^r defined in definition 50. Note that for every co-chain A: If $M_{i+1}^r \in A$ then $M_{j+1}^r \in \Gamma^j(A)$ (because $M_{i+1}^r \subseteq M_{j+1}^r$). Now we can see that:

$$\begin{aligned}\forall A \subseteq X(i) : \|A\| &= \Pr\left[Sample(i,r) \in A\right] = \Pr\left[M_{i+1}^r \in A\right] \leq \Pr\left[M_{j+1}^r \in \Gamma^j(A)\right] \\ &= \Pr\left[Sample(j,r) \in \Gamma^j(A)\right] = \left\|\Gamma^j(A)\right\|\end{aligned}$$ ◀

Proof of Lemma 11. First denote $U = \left\{A' \in X(i+j) \middle| \binom{A'}{i} \subseteq A\right\}$ and let M_i^r be the values defined in definition 50. Due to Lemma 51 we know that:

$$\begin{aligned}&\|A\| = \Pr\left[Sample(i,r) \in A\right] = \Pr\left[M_{i+1}^r \in A\right] = \\ &\Pr\left[M_{i+1}^r \in A \middle| M_{i+j}^r \in U\right] \cdot \Pr\left[M_{i+j}^r \in U\right] + \\ &\Pr\left[M_{i+1}^r \in A \middle| M_{i+j}^r \notin U\right] \cdot \Pr\left[M_{i+j}^r \notin U\right] \geq \\ &\Pr\left[M_{i+1}^r \in A \middle| M_{i+j}^r \in U\right] \cdot \Pr\left[M_{i+j}^r \in U\right] = \\ &\Pr\left[M_{i+j}^r \in U\right] = \Pr\left[Sample(i+j,r) \in U\right] = \|U\|\end{aligned}$$ ◀

C Direct Sum Separators

In this section we will provide two direct sum separators: One using reconstructing the origin-function of F, and the other using a test for being a k-direct-sum. The first method provided here yields a separator that separates a k-direct-sum from other functions with probability 1. The other method, allows reducing the query complexity while increasing the error margin.

C.1 Direct Sum Separator Using Reconstruction

In this section we will provide a simple direct sum separator that, given F, attempts to reconstruct the origin function of F and accepts whenever it succeeds.

▶ **Lemma 52.** *Let X_{k+2} be a the complete simplicial complex on $k+2$ nodes and $F : X_{k+2}(k-1) \to \{0,1\}$. Define f to be a function that, given $v \in X_{k+2}(0)$, picks a $q \in X_{k+2}(k-1)$ such that $v \notin q$ and returns $\sum_{w \in \binom{q}{k-1}} F(w \cup \{v\})$. We will show that: F is a k-direct-sum $\Leftrightarrow$ f is an origin function of F.*

Proof. $\Rightarrow$ F is a k-direct-sum therefore there exists an origin function to F denoted by f'.

$$\forall q : f(v) = \sum_{w \in \binom{q}{k-1}} F(w \cup \{v\}) = \sum_{w \in \binom{q}{k-1}} \left(f'(v) + \sum_{v' \in w} f'(v') \right) =$$
$$\binom{k}{k-1} f'(v) + \sum_{v' \in q} \binom{k-1}{k-2} f'(v') = k \cdot f'(v) + \sum_{v' \in q} (k-1) \cdot f'(v') = f'(v)$$

And therefore f is an origin function of F.
$\Leftarrow$ F has an origin function and therefore it is a k-direct-sum. ◀

This lemma allows us to create the following $\left(k+2, k, \binom{k+2}{k}, 1\right)$-separator:

▶ **Lemma 53** (Direct Sum Separator for Odd Values of k)**.** *The following is a $\left(k+2, k, \binom{k+2}{k}, 1\right)$-direct-sum-separator (given a function $F \in C^k(X)$ on a simplicial complex X):*

Algorithm 5 T_{sep}.

```
foreach node v ∈ X(0) do
    Calculate f(v).
end
foreach face q ∈ X(k) do
    Check whether F(q) = Σ_{e∈q} f(e), if it is not return 0.
end
Return 1
```

Proof. The algorithm returns 1 $\Leftrightarrow$ F is a k-direct-sum on X due to Lemma 52.
It is easy to see that the separator queries the entire function (Therefore it uses $\binom{k+2}{k}$ queries). ◀

C.2 Obtaining a Direct Sum Separator From Test

In this section we will show how to construct a separator out of a test for the k-direct-sums over a $k+1$ dimensional complex. This will help reduce query complexity.

▶ **Lemma 54** (Separator from Test)**.** *If there is a (Q, E)-test (denoted by T) for being a k-direct-sum on a $k+1$ dimensional complex then there is a $(k+2, k, Q, \rho)$-direct-sum-separator such that $\rho = min_{F \in C^{k-1}(X) \setminus \{k\text{-direct-sums}\}} \left\{ Pr[T_X^G = 0] \right\}$ where X is the complete $(k+1)$-dimensional complex.*

Proof. Consider the following tester:

Algorithm 6 T'_{sep}.

1 Run T on F and return its output.

It is east to see that the algorithm queries F exactly Q times.
All we have to prove is that if F is not a k-direct-sum than the algorithm returns false with probability of at least ρ. $F \in C^{k-1}(X) \setminus \{k\text{-direct-sums}\}$ and therefore:

$$Pr[T_X^G = 0] \geq min_{G \in C^{k-1}(X) \setminus \{k\text{-direct-sums}\}} \{Pr[T_X^G = 0]\} = \rho$$

Note that $\rho > 0$ because if $\rho = 0$ then there would exist a function F' such that $F' \in C^{k-1}(X) \setminus \{k\text{-direct-sums}\}$ and $Pr[T_X^{F'} = 0] = 0$. Note that T is a test and therefore if $Pr[T_X^{F'} = 0] = 0$ then F' is a k-direct-sum which contradicts the assumption about F'. ◀

Lastly, we can prove corollary 35:

Proof of Corollary 35. Combining the second test provided in [4] and Lemma 54 we get a $(k+2, k, O(k), \rho)$-separator. From the first test provided in [4] and Lemma 49 we get a $(3k+3, E')$-test for being a k-co-cycle-indifferent-direct-sum. Combining both of these results with Theorem 37 yields the desired result. ◀

A Lower Bound for Sampling Disjoint Sets

Mika Göös
Institute for Advanced Study, Princeton, NJ, USA
mika@ias.edu

Thomas Watson
University of Memphis, TN, USA
Thomas.Watson@memphis.edu

Abstract

Suppose Alice and Bob each start with private randomness and no other input, and they wish to engage in a protocol in which Alice ends up with a set $x \subseteq [n]$ and Bob ends up with a set $y \subseteq [n]$, such that (x, y) is uniformly distributed over all pairs of disjoint sets. We prove that for some constant $\beta < 1$, this requires $\Omega(n)$ communication even to get within statistical distance $1 - \beta^n$ of the target distribution. Previously, Ambainis, Schulman, Ta-Shma, Vazirani, and Wigderson (FOCS 1998) proved that $\Omega(\sqrt{n})$ communication is required to get within some constant statistical distance $\varepsilon > 0$ of the uniform distribution over all pairs of disjoint sets of size $\sqrt{n}$.

2012 ACM Subject Classification Theory of computation → Communication complexity

Keywords and phrases Communication complexity, set disjointness, sampling

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.51

Category RANDOM

Related Version A full version of the paper is available at `https://eccc.weizmann.ac.il/report/2019/066/`.

Funding *Mika Göös*: Supported by NSF grant CCF-1412958.
Thomas Watson: Supported by NSF grant CCF-1657377.

1 Introduction

In most traditional computational problems, the goal is to take an input and produce the "correct" output, or produce one of a set of acceptable outputs. In a *sampling* problem, on the other hand, the goal is to generate a random sample from a specified probability distribution D, or at least from a distribution that is close to D. There has been a surge of interest in studying sampling problems from a complexity theory perspective [7, 36, 73, 1, 58, 32, 74, 13, 72, 47, 77, 15, 78, 75, 79, 76]. Unlike more traditional computational problems, sampling problems do not necessarily need to have any real input, besides the uniformly random bits fed into a sampling algorithm.

One commonly studied type of target distribution is "input–output pairs" of a function f, i.e., $(D, f(D))$ where D is perhaps the uniform distribution over inputs to f. Using an algorithm for computing f, one can sample $(D, f(D))$ by first sampling from D, then evaluating f on that input. However, for some functions f, generating an input jointly with the corresponding output may be computationally easier than evaluating f on an adversarially-chosen input. Thus in general, sampling lower bounds tend to be more challenging to prove than lower bounds for functions.

Many of the above-cited works focus on concrete computational models such as low-depth circuits. We consider the model of 2-party communication complexity, for which comparatively less is known about sampling. Which problem should we study? Well, the single most important function in communication complexity is Set-Disjointness, in which Alice gets a set


© Mika Göös and Thomas Watson;
licensed under Creative Commons License CC-BY

$x \subseteq [n]$, Bob gets a set $y \subseteq [n]$, and the goal is to determine whether $x \cap y = \emptyset$. Identifying the sets with their characteristic bit strings, this can be viewed as $\textsc{Disj}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ where $\textsc{Disj}(x,y) = 1$ iff $x \wedge y = 0^n$. The applications of communication bounds for Set-Disjointness are far too numerous to list, but they span areas such as streaming, circuit complexity, proof complexity, data structures, property testing, combinatorial optimization, fine-grained complexity, cryptography, and game theory. Because of its central role, Set-Disjointness has become the de facto testbed for proving new types of communication bounds. This function has been studied in the contexts of randomized [9, 49, 62, 10, 17] and quantum [25, 43, 63, 2, 66, 70] protocols; multi-party number-in-hand [6, 10, 27, 41, 48, 18, 22] and number-on-forehead [40, 71, 12, 66, 28, 57, 11, 69, 68, 61, 60] models; Merlin–Arthur and related models [50, 3, 35, 39, 38, 4, 64, 29]; with a bounded number of rounds of interaction [52, 46, 80, 19, 23]; with bounds on the sizes of the sets [42, 56, 59, 31, 26, 65]; very precise relationships between communication and error probability [20, 21, 39, 33, 30]; when the goal is to find the intersection [24, 34, 79, 8]; in space-bounded, online, and streaming models [53, 16, 5]; and direct product theorems [54, 12, 14, 45, 51, 67, 69, 68]. We contribute one more result to this thorough assault on Set-Disjointness.

Here is the definition of our 2-party sampling model: Let D be a probability distribution over $\{0,1\}^n \times \{0,1\}^n$; we also think of D as a matrix with rows and columns both indexed by $\{0,1\}^n$ where $D_{x,y}$ is the probability of outcome (x,y). We define $\mathsf{Samp}(D)$ as the minimum communication cost of any protocol where Alice and Bob each start with private randomness and no other input, and at the end Alice outputs some $x \in \{0,1\}^n$ and Bob outputs some $y \in \{0,1\}^n$ such that (x,y) is distributed according to D. Note that $\mathsf{Samp}(D) = 0$ iff D is a product distribution (x and y are independent), and $\mathsf{Samp}(D) \leq n$ for all D (since Alice can privately sample (x,y) and send y to Bob). Allowing public randomness would not make sense since Alice and Bob could read a properly-distributed (x,y) off of the randomness without communicating. We define $\mathsf{Samp}_\varepsilon(D)$ as the minimum of $\mathsf{Samp}(D')$ over all distributions D' with $\Delta(D,D') \leq \varepsilon$, where Δ denotes statistical (total variation) distance, defined as

$$\Delta(D,D') \coloneqq \max_{\text{event } E} \big|\mathbb{P}_D[E] - \mathbb{P}_{D'}[E]\big| = \max_{\text{event } E} \big(\mathbb{P}_D[E] - \mathbb{P}_{D'}[E]\big) = \tfrac{1}{2} \sum_{\text{outcome } o} \big|\mathbb{P}_D[o] - \mathbb{P}_{D'}[o]\big|.$$

1.1 A story

Our story begins with [7], which proved that $\mathsf{Samp}_\varepsilon\big((D, \textsc{Disj}(D))\big) \geq \Omega(\sqrt{n})$ for some constant $\varepsilon > 0$, where D is uniform over the set of all pairs of sets of size $\sqrt{n}$ (note that this D is a product distribution and is approximately balanced between 0-inputs and 1-inputs of $\textsc{Disj}$); here it does not matter which party is responsible for outputting the bit $\textsc{Disj}(D)$. The main tool in the proof was a lemma that was originally employed in [9] to prove an $\Omega(\sqrt{n})$ bound on the randomized communication complexity of *computing* $\textsc{Disj}$. The latter bound was improved to $\Omega(n)$ via several different proofs [49, 62, 10], which leads to a natural question: Can we improve the sampling bound of [7] to $\Omega(n)$ by using the techniques of [49, 62, 10] instead of [9]?

For starters, the answer is "no" for the particular D considered in [7] – there is a trivial exact protocol with $O(\sqrt{n} \log n)$ communication since it only takes that many bits to specify a set of size $\sqrt{n}$. What about other interesting distributions D? The following illuminates the situation.

▷ Observation 1. For any D and constants $\varepsilon > \delta > 0$, if $\mathsf{Samp}_\varepsilon\big((D, \textsc{Disj}(D))\big) \geq \omega(\sqrt{n})$ then $\mathsf{Samp}_\delta(D) \geq \Omega\big(\mathsf{Samp}_\varepsilon\big((D, \textsc{Disj}(D))\big)\big)$.

Proof. It suffices to show $\mathsf{Samp}_\varepsilon\big((D, \textsc{Disj}(D))\big) \leq \mathsf{Samp}_\delta(D) + O(\sqrt{n})$. First, note that for any sampling protocol, if we condition on a particular transcript then the output distribution becomes product (Alice and Bob are independent after they stop communicating). Second, [17] proved that for every product distribution and every constant $\gamma > 0$, there exists a deterministic protocol that uses $O(\sqrt{n})$ bits of communication and computes Disj with error probability $\leq \gamma$ on a random input from the distribution. Now to ε-sample $(D, \textsc{Disj}(D))$, Alice and Bob can δ-sample D to obtain (x, y), and then conditioned on that sampler's transcript, they can run the average-case protocol from [17] for the corresponding product distribution with error $\varepsilon - \delta$. A simple calculation shows this indeed gives statistical distance ε. ◁

The upshot is that to get an improved bound, *the hardness of sampling* $(D, \textsc{Disj}(D))$ *would come entirely from the hardness of just sampling* D. Thus such a result would not really be "about" the Set-Disjointness function, it would be about the distribution on inputs. Instead of abandoning this line of inquiry, we realize that if D itself is somehow defined in terms of Disj, then a bound for sampling D would still be saying something about the complexity of Set-Disjointness. In fact, the proof in [7] actually shows something stronger than the previously-stated result: If D is instead defined as the uniform distribution over pairs of *disjoint* sets of size $\sqrt{n}$ (which are 1-inputs of Disj), then $\mathsf{Samp}_\varepsilon(D) \geq \Omega(\sqrt{n})$. After this pivot, we are now facing a direction in which we can hope for an improvement. We prove that by removing the restriction on the sizes of the sets, the sampling problem becomes maximally hard. Our result holds for error $\varepsilon < 1$ that is exponentially close to 1, but the result is already new and interesting for constant $\varepsilon > 0$.

▶ **Theorem 1.** *Let U be the uniform distribution over the set of all $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ with $x \wedge y = 0^n$. There exists a constant $\beta < 1$ such that $\mathsf{Samp}_{1-\beta^n}(U) = \Omega(n)$.*

The proof from [7] was a relatively short application of the technique from [9], but for Theorem 1, harnessing known techniques for proving linear communication lower bounds turns out to be more involved.

For calibration, the uniform distribution over *all* (x, y) achieves statistical distance $1 - 0.75^n$ from U since there are 4^n inputs and 3^n disjoint inputs. We can do a little better: Suppose for each coordinate independently, Alice picks 0 with probability $\sqrt{1/3}$ and picks 1 with probability $1 - \sqrt{1/3}$, and Bob does the same. This again involves no communication, and it achieves statistical distance $1 - \big(2\sqrt{1/3} - 1/3\big)^n \leq 1 - 0.82^n$ from U. Theorem 1 shows that the constant 0.82 cannot be improved arbitrarily close to 1 without a lot of communication. (In the setting of lower bounds for circuit samplers, significant effort has gone into handling statistical distances exponentially close to the maximum possible [32, 13, 76].)

1.2 Interpreting the result

We first observe that our sampling model is equivalent to two other models. One of these we call (for lack of a better word) "synthesizing" the distribution D: Alice and Bob get inputs $x, y \in \{0,1\}^n$ respectively, in addition to their private randomness, and their goal is to accept with probability exactly $D_{x,y}$. We let $\mathsf{Synth}(D)$ denote the minimum communication cost of any synthesizing protocol for D, and $\mathsf{Synth}_\varepsilon(D)$ denote the minimum of $\mathsf{Synth}(D')$ over all D' with $\Delta(D, D') \leq \varepsilon$. The other model is the nonnegative rank of a matrix: $\mathrm{rank}_+(D)$ is defined as the minimum k for which D can be written as a sum of k many nonnegative rank-1 matrices.

▷ Observation 2. For every distribution D, the following are all within $\pm O(1)$ of each other:

$$\mathsf{Samp}(D), \quad \mathsf{Synth}(D), \quad \log \mathrm{rank}_+(D).$$

Proof. $\mathsf{Synth}(D) \leq \mathsf{Samp}(D) + 2$ since a synthesizing protocol can just run a sampling protocol and accept iff the result equals the given input (x, y).

$\log \mathrm{rank}_+(D) \leq \mathsf{Synth}(D)$ since for each transcript of a synthesizing protocol, the matrix that records the probability of getting that transcript on each particular input has rank 1; summing these matrices over all accepting transcripts yields a nonnegative rank decomposition of D.

To see that $\mathsf{Samp}(D) \leq \lceil \log \mathrm{rank}_+(D) \rceil$, suppose $D = M^{(1)} + M^{(2)} + \cdots + M^{(k)}$ is a sum of nonnegative rank-1 matrices. For each i, by scaling we can write $M^{(i)}_{x,y} = p_i \, u^{(i)}_x v^{(i)}_y$ for some distributions $u^{(i)}$ and $v^{(i)}$ over $\{0,1\}^n$, where p_i is the sum of all entries of $M^{(i)}$. Since D is a distribution, $p := (p_1, \ldots, p_k)$ is a distribution over $[k]$. To sample from D, Alice can privately sample $i \sim p$ and send it to Bob using $\lceil \log k \rceil$ bits, then Alice can sample $x \sim u^{(i)}$ and Bob can independently sample $y \sim v^{(i)}$ with no further communication. ◁

By this characterization, Theorem 1 can be viewed as a lower bound on the approximate nonnegative rank of the Disj matrix, where the approximation is in ℓ_1 (which has an average-case flavor). In the recent literature, "approximate nonnegative rank" generally refers to approximation in ℓ_∞ (which is a worst-case requirement), and this model is equivalent to the so-called smooth rectangle bound and WAPP communication complexity [44, 55, 37].

2 Proof

2.1 Overview

Our proof of Theorem 1 is by a black-box reduction to the well-known *corruption lemma* for Set-Disjointness due to Razborov [62]. We start with a high-level overview.

For notation: Let $|z|$ denote the Hamming weight of a string $z \in \{0,1\}^n$. For $\ell \in \mathbb{N}$, let U^ℓ be the uniform distribution over all $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ with $|x \wedge y| = \ell$. Note that $U = U^0$. For a distribution D over $\{0,1\}^n \times \{0,1\}^n$ and an event $E \subseteq \{0,1\}^n \times \{0,1\}^n$, let $D_E := \sum_{(x,y)\in E} D_{x,y}$. For a randomized protocol Π, let $\mathrm{acc}_\Pi(x, y)$ denote the probability that Π accepts (x, y).

Step I: Uniform corruption

The corruption lemma states that if a rectangle $R \subseteq \{0,1\}^n \times \{0,1\}^n$ contains a noticeable fraction of *disjoint* pairs, then it must contain about as large a fraction of *uniquely intersecting* pairs. More quantitatively, there exist a constant $C > 0$ and two distributions D^ℓ, $\ell = 0, 1$, defined over disjoint ($\ell = 0$) and uniquely intersecting pairs ($\ell = 1$) such that for every rectangle R,

$$\text{if} \quad D^0_R \geq 2^{-o(n)} \quad \text{then} \quad D^1_R \geq C \cdot D^0_R.$$

The original proof [62] defined D^ℓ as the uniform distribution over all pairs (x, y) with fixed sizes $|x| = |y| = \lceil n/4 \rceil$ and $|x \wedge y| = \ell$. For our purpose, we need the corruption lemma to hold relative to the aforementioned distributions U^ℓ, $\ell = 0, 1$, which have no restrictions on set sizes. We derive in Subsection 2.2 a corruption lemma for U^ℓ from the original lemma for D^ℓ. To do this, we exhibit a reduction that uses public randomness and no communication to transform a sample from D^ℓ into a sample from a distribution that is close to U^ℓ in a suitable sense, for $\ell = 0, 1$.

Step II: Truncate and scale

For simplicity, let us think about proving Theorem 1 for a small error $\varepsilon > 0$. Assume for contradiction there is some distribution D, $\Delta(U, D) \leq \varepsilon$, such that $\mathsf{Synth}(D) \leq o(n)$ as witnessed by a private-randomness synthesizing protocol Π' with $\mathrm{acc}_{\Pi'}(x, y) = D_{x,y}$. Note that the total acceptance probability over disjoint inputs is close to 1:

$$\sum_{x,y\,:\,|x\wedge y|=0} \mathrm{acc}_{\Pi'}(x,y) \;\geq\; 1-\varepsilon \quad \text{and thus} \quad \mathbb{E}_{(x,y)\sim U^0}[\mathrm{acc}_{\Pi'}(x,y)] \;\geq\; (1-\varepsilon)3^{-n}.$$

Our eventual goal (in Step III) is to apply our corruption lemma to the transcript rectangles, but the above threshold $(1-\varepsilon)3^{-n}$ is too low for this. To raise the threshold to $2^{-o(n)}$ as needed for corruption, we would like to scale up all the acceptance probabilities accordingly. To "make room" for the scaling, we first carry out a certain truncation step. Specifically, in Subsection 2.3 we transform Π' into a public-randomness protocol Π:

1. First, we **truncate** (using a *truncation lemma* [37]) the values $\mathrm{acc}_{\Pi'}(x, y)$, which has the effect of decreasing some of them, but any $\mathrm{acc}_{\Pi'}(x, y)$ that is under 3^{-n} remains approximately the same. This results in an intermediate protocol Π'' that still satisfies $\mathbb{E}_{(x,y)\sim U^0}[\mathrm{acc}_{\Pi''}(x, y))] \geq \Omega((1-\varepsilon)3^{-n})$ (using the assumption that $\Delta(U, D) \leq \varepsilon$).
2. Second, we **scale** (using the low cost of Π'') the truncated probabilities up by a large factor $3^n 2^{-o(n)}$. This results in a protocol Π with large typical acceptance probabilities:

$$\mathbb{E}_{(x,y)\sim U^0}[\mathrm{acc}_{\Pi}(x,y)] \;\geq\; 2^{-o(n)}. \tag{1}$$

Step III: Iterate corruption

Because Π has such large acceptance probabilities (Equation 1), our corruption lemma can be applied: there is some constant $C' > 0$ such that

$$\mathbb{E}_{(x,y)\sim U^1}[\mathrm{acc}_{\Pi}(x,y)] \;\geq\; C' \cdot \mathbb{E}_{(x,y)\sim U^0}[\mathrm{acc}_{\Pi}(x,y)]. \tag{2}$$

Since Π is a truncated-and-scaled version of Π', this allows us to infer that

$$\mathbb{E}_{(x,y)\sim U^1}[\mathrm{acc}_{\Pi'}(x,y)] \;\geq\; \Omega((1-\varepsilon)3^{-n}) \quad \text{and thus} \quad \sum_{x,y\,:\,|x\wedge y|=1} \mathrm{acc}_{\Pi'}(x,y) \;\geq\; \Omega((1-\varepsilon)n)$$

using the fact that $|\mathrm{supp}(U^1)| = n3^{n-1} = (n/3) \cdot |\mathrm{supp}(U^0)|$. Thus for $\varepsilon = 1 - \omega(1/n)$, this means Π' must have placed a total probability mass > 1 on uniquely intersecting inputs, which is the sought contradiction.

To prove Theorem 1 for very large error $\varepsilon = 1 - \beta^n$, in Subsection 2.4 we iterate the above argument for U^ℓ over $0 \leq \ell \leq o(n)$. Namely, analogously to Equation 2, we show that the average acceptance probability of Π over $U^{\ell+1}$ is at least a constant times the average over U^ℓ. Meanwhile, the support sizes increase as $|\mathrm{supp}(U^{\ell+1})| \geq \omega(1) \cdot |\mathrm{supp}(U^\ell)|$ for $\ell \leq o(n)$. These facts together imply a large constant factor increase in the total probability mass that Π' places on $\mathrm{supp}(U^{\ell+1})$ as compared to $\mathrm{supp}(U^\ell)$. Starting with even a tiny probability mass over $\mathrm{supp}(U^0)$, this iteration will eventually lead to a contradiction.

2.2 Step I: Uniform corruption

The goal of this step is to derive Lemma 3 from Lemma 2.

▶ **Lemma 2** (Corruption [62]). *For every rectangle $R \subseteq \{0,1\}^n \times \{0,1\}^n$ we have $D^1_R \geq \frac{1}{45} D^0_R - 2^{-0.017n}$ where, assuming $n = 4k - 1$, D^ℓ is the uniform distribution over all (x, y) with $|x| = |y| = k$ and $|x \wedge y| = \ell$.*

▶ **Lemma 3** (Uniform Corruption). *For every rectangle $R \subseteq \{0,1\}^n \times \{0,1\}^n$ we have $U^1_R \geq \frac{1}{765} U^0_R - 2^{-0.008n}$.*

Proof. Assume for convenience that $n/2$ has the form $4k-1$ (otherwise use the nearest such number instead of $n/2$ throughout). We prove that Lemma 2 for $n/2$ implies Lemma 3 for n by the contrapositive. Thus, D^0 and D^1 are distributions over $\{0,1\}^{n/2} \times \{0,1\}^{n/2}$ while U^0 and U^1 are distributions over $\{0,1\}^n \times \{0,1\}^n$. Assume there exists a rectangle $R \subseteq \{0,1\}^n \times \{0,1\}^n$ such that $U^1_R < \frac{1}{765} U^0_R - 2^{-0.008n}$. We exhibit a distribution over rectangles $Q \subseteq \{0,1\}^{n/2} \times \{0,1\}^{n/2}$ such that $\mathbb{E}[D^1_Q] < \frac{1}{45}\mathbb{E}[D^0_Q] - 2^{-0.017n/2}$; by linearity of expectation this implies that there exists such a Q with $D^1_Q < \frac{1}{45} D^0_Q - 2^{-0.017n/2}$.

To this end, we define a distribution F over functions $f\colon \{0,1\}^{n/2} \times \{0,1\}^{n/2} \to \{0,1\}^n \times \{0,1\}^n$ of the form $f(x,y) = (f_1(x), f_2(y))$ and then let Q_f be the rectangle $f^{-1}(R) \coloneqq \{(x,y) \,:\, f(x,y) \in R\}$. Let H be the distribution over $\{(v,w) \in \mathbb{N} \times \mathbb{N} \,:\, v+w \leq n\}$ obtained by sampling $(x,y) \sim U^0$ and outputting $(|x|,|y|)$; i.e., $H_{v,w} \coloneqq \frac{n!}{v!\,w!\,(n-v-w)!} \cdot 3^{-n}$. To sample $f \sim F$:

1. Sample (v,w) from H conditioned on $v \geq k$, $w \geq k$, and $v+w \leq 2k+n/2$.
2. Sample a uniformly random permutation π of $[n]$.
3. Given $(x,y) \in \{0,1\}^{n/2} \times \{0,1\}^{n/2}$, define $(x',y') \in \{0,1\}^n \times \{0,1\}^n$ by letting

$$x'_i y'_i \coloneqq \begin{cases} x_i y_i & \text{for the first } n/2 \text{ coordinates } i; \\ 10 & \text{for the next } v-k \text{ coordinates } i; \\ 01 & \text{for the next } w-k \text{ coordinates } i; \\ 00 & \text{for the remaining } n/2-(v-k)-(w-k) \geq 0 \text{ coordinates } i. \end{cases}$$

4. Let $f(x,y) \coloneqq (\pi(x'), \pi(y'))$ (i.e., permute the coordinates according to π).

For $\ell \in \{0,1\}$ let $F(D^\ell)$ denote the distribution obtained by sampling $(x,y) \sim D^\ell$ and $f \sim F$ and outputting $f(x,y)$, and note that $F(D^\ell)_R = \mathbb{E}_F[D^\ell_{Q_F}]$. Now we claim that $F(D^\ell)$ and U^ℓ are close, in the following senses:

(1) For every event E, $F(D^0)_E \geq U^0_E - 2^{-0.01n}$.

(2) For every event E, $F(D^1)_E \leq U^1_E \cdot 17$.

Using R as the event E, we have

$$\begin{aligned} F(D^1)_R &\leq U^1_R \cdot 17 \\ &< 17\big(\tfrac{1}{765} U^0_R - 2^{-0.008n}\big) \\ &\leq 17\big(\tfrac{1}{765}(F(D^0)_R + 2^{-0.01n}) - 2^{-0.008n}\big) \\ &\leq \tfrac{1}{45} F(D^0)_R - 2^{-0.017n/2} \end{aligned}$$

as desired. To see *(1)*, note that $F(D^0)$ is precisely U^0 conditioned on $v \geq k$, $w \geq k$, and $v+w \leq 2k+n/2$, and this conditioning event has probability $\geq 1-2^{-0.01n}$ by Chernoff bounds:

$$\begin{aligned} \mathbb{P}[v<k] \;=\; \mathbb{P}[w<k] \;&=\; \mathbb{P}[\mathrm{Bin}(n,1/3) < n/8 + 1/4] \;\leq\; 2^{-0.12n} \\ \mathbb{P}[v+w > 2k+n/2] \;&=\; \mathbb{P}[\mathrm{Bin}(n,2/3) > 3n/4 + 1/2] \;\leq\; 2^{-0.02n} \end{aligned}$$

Thus letting C be the complement of the conditioning event, we have $F(D^0)_E \geq U^0_{E \setminus C} \geq U^0_E - U^0_C \geq U^0_E - 2^{-0.01n}$. To see *(2)*, consider any outcome $(x,y) \in \{0,1\}^n \times \{0,1\}^n$ with $|x \wedge y| = 1$. We have $U^1_{x,y} = 1/(n3^{n-1})$. Abbreviating $a \coloneqq |x|$ and $b \coloneqq |y|$, assume $a \geq k$, $b \geq k$, and $a+b \leq 2k+n/2$ since otherwise $F(D^1)_{x,y} = 0$ and there would be nothing to prove. Henceforth consider the probability space with the randomness of D^1 and of F. Let I be the event that $F_1(D^1) \wedge F_2(D^1) = x \wedge y$, i.e., that the intersecting coordinate of $F(D^1)$ is the same as for (x,y). We have

$$F(D^1)_{x,y} \;=\; \underbrace{\mathbb{P}[I]}_{(*)} \cdot \underbrace{\mathbb{P}[v=a \text{ and } w=b]}_{(**)} \cdot \underbrace{\mathbb{P}\big[F(D^1) = (x,y) \,\big|\, I \text{ and } v=a \text{ and } w=b\big]}_{(***)}.$$

For the three terms on the right side, we have

$$(*) = \tfrac{1}{n}, \quad (**) \le H_{a,b}/(1-2^{-0.01n}) \le \tfrac{n!}{a!\,b!\,(n-a-b)!} \cdot 3^{-n} \cdot 1.01, \quad (***) = 1/\tfrac{(n-1)!}{(a-1)!\,(b-1)!\,(n-a-b+1)!}.$$

We have

$$\tfrac{n!}{a!\,b!\,(n-a-b)!} \,/\, \tfrac{(n-1)!}{(a-1)!\,(b-1)!\,(n-a-b+1)!} \;=\; \tfrac{n\cdot(n-a-b+1)}{a\cdot b} \;\le\; \tfrac{n\cdot(n-2k+1)}{k\cdot k} \;\le\; \tfrac{n\cdot(n-2n/8+1)}{(n/8)\cdot(n/8)} \;=\; (\tfrac{3}{4}+\tfrac{1}{n})\cdot 64.$$

Combining, we get

$$F(D^1)_{x,y} \,/\, U^1_{x,y} \;=\; (*)\cdot(**)\cdot(***)\cdot n3^{n-1} \;\le\; \tfrac{1.01}{3}\cdot(\tfrac{3}{4}+\tfrac{1}{n})\cdot 64 \;\le\; 17. \qquad \blacktriangleleft$$

2.3 Step II: Truncate and scale

The goal of this step is to construct a truncated-and-scaled protocol Π from any given low-cost Π' that synthesizes a distribution close to U.

For a nonnegative matrix M, we define its *truncation* $\overline{M}$ to be the same matrix but where each entry > 1 is replaced with 1.

▶ **Lemma 4** (Truncation Lemma [37]). *For every $2^n \times 2^n$ nonnegative rank-1 matrix M and every d there exists a $O(d+\log n)$-communication public-randomness protocol Π such that for every (x,y) we have $\mathrm{acc}_\Pi(x,y) \in \overline{M}_{x,y} \pm 2^{-d}$.*

Let $c \ge 1$ be the hidden constant in the big O in Lemma 4, and let $\delta := 0.00005/c$. Toward proving Theorem 1, suppose for contradiction $\mathsf{Samp}(D) \le \delta n$ for some distribution D with $\Delta(U,D) \le 1-2^{-\delta n}$ (so $\beta := 2^{-\delta}$ in Theorem 1) and thus $\sum_{x,y\,:\,|x\wedge y|=0} \min(3^{-n}, D_{x,y}) \ge 2^{-\delta n}$. By Observation 2, $\mathsf{Synth}(D) \le \delta n + 2$, so consider a synthesizing protocol Π' for D with communication cost $\le \delta n + 2$. Let A be the set of all accepting transcripts of Π'. For each $\tau \in A$ let N^τ be the nonnegative rank-1 matrix such that $N^\tau_{x,y}$ is the probability Π' generates τ on input (x,y); thus $D_{x,y} = \sum_{\tau\in A} N^\tau_{x,y}$. Let Π^τ be the public-randomness protocol from Lemma 4 applied to $M^\tau := 3^n N^\tau$ and $d := 15\delta n$. Let Π be the public-randomness protocol that picks a uniformly random $\tau \in A$ and then runs Π^τ. The communication cost of Π is $\le c\cdot(d+\log n) \le 0.001n$.

▷ Claim 5. For every input (x,y) we have $\frac{3^n}{|A|}\min(3^{-n}, D_{x,y}) - 2^{-d} \le \mathrm{acc}_\Pi(x,y) \le \frac{3^n}{|A|} D_{x,y} + 2^{-d}$.

Proof. We have

$$\begin{aligned}
\mathrm{acc}_\Pi(x,y) &= \tfrac{1}{|A|}\textstyle\sum_{\tau\in A} \mathrm{acc}_{\Pi^\tau}(x,y) \\
&\in \tfrac{1}{|A|}\textstyle\sum_{\tau\in A} (\overline{M}^\tau_{x,y} \pm 2^{-d}) \\
&\subseteq \tfrac{1}{|A|}\textstyle\sum_{\tau\in A} \min(1, 3^n N^\tau_{x,y}) \pm 2^{-d} \\
&= \tfrac{3^n}{|A|}\textstyle\sum_{\tau\in A} \min(3^{-n}, N^\tau_{x,y}) \pm 2^{-d}.
\end{aligned}$$

From this it follows that:

$$\begin{aligned}
\mathrm{acc}_\Pi(x,y) &\ge \tfrac{3^n}{|A|}\min\big(3^{-n}, \textstyle\sum_{\tau\in A} N^\tau_{x,y}\big) - 2^{-d} &&= \tfrac{3^n}{|A|}\min(3^{-n}, D_{x,y}) - 2^{-d} \\
\mathrm{acc}_\Pi(x,y) &\le \tfrac{3^n}{|A|}\textstyle\sum_{\tau\in A} N^\tau_{x,y} + 2^{-d} &&= \tfrac{3^n}{|A|} D_{x,y} + 2^{-d}.
\end{aligned}$$ ◁

We can now formally state the large typical acceptance probability property (Equation 1 from the overview): writing $U_\Pi \coloneqq \mathbb{E}_{(x,y)\sim U}[\mathrm{acc}_\Pi(x,y)]$ (and similarly for other input distributions),

$$\begin{aligned} U_\Pi &\geq \tfrac{1}{3^n}\textstyle\sum_{x,y\,:\,|x\wedge y|=0}\big(\frac{3^n}{|A|}\min(3^{-n}, D_{x,y}) - 2^{-d}\big) && \text{(by Claim 5)}\\ &= \tfrac{1}{|A|}\textstyle\sum_{x,y\,:\,|x\wedge y|=0}\min(3^{-n}, D_{x,y}) - 2^{-d}\\ &\geq \tfrac{1}{|A|}2^{-\delta n} - 2^{-15\delta n}\\ &\geq \tfrac{1}{|A|}2^{-\delta n-1} && (3) \end{aligned}$$

where the last line follows because $|A| \leq 2^{\delta n+2}$ and $2^{-2\delta n-2}$ is at least twice $2^{-15\delta n}$.

2.4 Step III: Iterate corruption

Here we derive the final contradiction: Π' places an acceptance probability mass exceeding 1 on $\mathrm{supp}(U^{\delta n})$. This is achieved by iterating our corruption lemma, starting with Equation 3 as the base case.

For $z \in \{0,1\}^n$ let U^z be the uniform distribution over all $(x,y) \in \{0,1\}^n \times \{0,1\}^n$ with $x \wedge y = z$ (so U^ℓ is the uniform mixture of all U^z with $|z| = \ell$; in particular, $U^0 = U^{0^n}$), and if $|z| < n$ then let $\widehat{U}^z$ be the uniform mixture of $U^{z'}$ over all z' that can be obtained from z by flipping a single 0 to 1 (so $U^{\ell+1}$ is the uniform mixture of all $\widehat{U}^z$ with $|z| = \ell$; in particular, $U^1 = \widehat{U}^{0^n}$).

▷ Claim 6. For every $z \in \{0,1\}^n$ with $|z| \leq n/2$ we have $\widehat{U}^z_\Pi \geq \frac{1}{765}U^z_\Pi - 2^{-0.003n}$.

Proof. Since all relevant inputs (x,y) have $x_i y_i = 11$ for all i such that $z_i = 1$, we can ignore those coordinates and think of $\widehat{U}^z$ and U^z as U^1 and U^0 respectively, but defined on the remaining $n - |z| \geq n/2$ coordinates (instead of on all n coordinates). Thus by Lemma 3, for every outcome of the public randomness of Π and every accepting transcript, say corresponding to rectangle R, we have $\widehat{U}^z_R \geq \frac{1}{765}U^z_R - 2^{-0.008n/2}$. Summing over all the (at most $2^{0.001n}$ many) accepting transcripts, and then taking the expectation over the public randomness, yields the claim since $2^{0.001n} \cdot 2^{-0.008n/2} \leq 2^{-0.003n}$. ◁

▷ Claim 7. For every $\ell = 0, \ldots, \delta n$ we have $U^\ell_\Pi \geq \frac{1}{|A|}2^{-\delta n-1-11\ell}$.

Proof. We prove this by induction on ℓ. The base case $\ell = 0$ is Equation 3. For the inductive step, assume the claim is true for ℓ. Since $U^{\ell+1}$ and U^ℓ are the uniform mixtures of $\widehat{U}^z$ and U^z respectively over all z with $|z| = \ell$ (so $U^{\ell+1}_\Pi = \mathbb{E}_z[\widehat{U}^z_\Pi]$ and $U^\ell_\Pi = \mathbb{E}_z[U^z_\Pi]$), by linearity of expectation Claim 6 implies

$$U^{\ell+1}_\Pi \geq \tfrac{1}{765}U^\ell_\Pi - 2^{-0.003n} \geq \tfrac{1}{|A|}2^{-\delta n-1-11\ell-\log_2(765)} - 2^{-0.003n} \geq \tfrac{1}{|A|}2^{-\delta n-1-11(\ell+1)}$$

where the last inequality follows because $|A| \leq 2^{\delta n+2}$ and $2^{-\delta n-2-\delta n-1-11\delta n-\log_2(765)} \geq 2^{-14\delta n}$ is at least twice $2^{-0.003n}$, which gives $U^{\ell+1}_\Pi \geq \frac{1}{|A|}2^{-\delta n-1-11\ell-\log_2(765)-1}$, and $\log_2(765) + 1 \leq 11$. ◁

Choosing $\ell = \delta n$ we have

$$U^\ell_\Pi - 2^{-d} \geq \tfrac{1}{|A|}2^{-\delta n-1-11\ell} - 2^{-15\delta n} \geq \tfrac{1}{|A|}2^{-\delta n-2-11\ell} \qquad (4)$$

because $|A| \leq 2^{\delta n+2}$ and $2^{-\delta n-2-\delta n-1-11\delta n} \geq 2^{-14\delta n}$ is at least twice $2^{-15\delta n}$. Thus, for $\ell = \delta n$,

$$\begin{aligned}
\textstyle\sum_{x,y} D_{x,y} &\geq \textstyle\sum_{x,y\,:\,|x\wedge y|=\ell} D_{x,y} \\
&\geq \textstyle\sum_{x,y\,:\,|x\wedge y|=\ell} \frac{|A|}{3^n}(\mathrm{acc}_\Pi(x,y) - 2^{-d}) && \text{(by Claim 5)} \\
&= \tfrac{|A|}{3^n}\binom{n}{\ell}3^{n-\ell}(U_\Pi^\ell - 2^{-d}) \\
&\geq \tfrac{|A|}{3^n}(\tfrac{n}{\ell})^\ell 3^{n-\ell}\tfrac{1}{|A|}2^{-\delta n-2-11\ell} && \text{(using Equation 4)} \\
&= (\tfrac{n}{\ell\cdot 3\cdot 2^{11}})^\ell 2^{-\delta n-2} \\
&= (\tfrac{1}{\delta\cdot 3\cdot 2^{11}\cdot 2})^{\delta n}/4 \\
&\geq 1.6^{\delta n} \\
&> 1,
\end{aligned}$$

contradicting the fact that D is a distribution.

References

1. Scott Aaronson. The Equivalence of Sampling and Searching. *Theory of Computing Systems*, 55(2):281–298, 2014. doi:10.1007/s00224-013-9527-3.
2. Scott Aaronson and Andris Ambainis. Quantum Search of Spatial Regions. *Theory of Computing*, 1(1):47–79, 2005. doi:10.4086/toc.2005.v001a004.
3. Scott Aaronson and Avi Wigderson. Algebrization: A New Barrier in Complexity Theory. *ACM Transactions on Computation Theory*, 1(1):2:1–2:54, 2009. doi:10.1145/1490270.1490272.
4. Amir Abboud, Aviad Rubinstein, and Ryan Williams. Distributed PCP Theorems for Hardness of Approximation in P. In *Proceedings of the 58th Symposium on Foundations of Computer Science (FOCS)*, pages 25–36. IEEE, 2017. doi:10.1109/FOCS.2017.12.
5. Josh Alman, Joshua Wang, and Huacheng Yu. Cell-Probe Lower Bounds from Online Communication Complexity. In *Proceedings of the 50th Symposium on Theory of Computing (STOC)*, pages 1003–1012. ACM, 2018. doi:10.1145/3188745.3188862.
6. Noga Alon, Yossi Matias, and Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999. doi:10.1006/jcss.1997.1545.
7. Andris Ambainis, Leonard Schulman, Amnon Ta-Shma, Umesh Vazirani, and Avi Wigderson. The Quantum Communication Complexity of Sampling. *SIAM Journal on Computing*, 32(6):1570–1585, 2003. doi:10.1137/S009753979935476.
8. Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Polynomial Pass Lower Bounds for Graph Streaming Algorithms. In *Proceedings of the 51st Symposium on Theory of Computing (STOC)*, pages 265–276. ACM, 2019. doi:10.1145/3313276.3316361.
9. László Babai, Peter Frankl, and Janos Simon. Complexity Classes in Communication Complexity Theory. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 337–347. IEEE, 1986. doi:10.1109/SFCS.1986.15.
10. Ziv Bar-Yossef, T.S. Jayram, Ravi Kumar, and D. Sivakumar. An Information Statistics Approach to Data Stream and Communication Complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004. doi:10.1016/j.jcss.2003.11.006.
11. Paul Beame and Dang-Trinh Huynh-Ngoc. Multiparty Communication Complexity and Threshold Circuit Size of AC^0. In *Proceedings of the 50th Symposium on Foundations of Computer Science (FOCS)*, pages 53–62. IEEE, 2009. doi:10.1109/FOCS.2009.12.
12. Paul Beame, Toniann Pitassi, Nathan Segerlind, and Avi Wigderson. A Strong Direct Product Theorem for Corruption and the Multiparty Communication Complexity of Disjointness. *Computational Complexity*, 15(4):391–432, 2006. doi:10.1007/s00037-007-0220-2.

13 Christopher Beck, Russell Impagliazzo, and Shachar Lovett. Large Deviation Bounds for Decision Trees and Sampling Lower Bounds for AC^0-Circuits. In *Proceedings of the 53rd Symposium on Foundations of Computer Science (FOCS)*, pages 101–110. IEEE, 2012. doi:10.1109/FOCS.2012.82.

14 Avraham Ben-Aroya, Oded Regev, and Ronald de Wolf. A Hypercontractive Inequality for Matrix-Valued Functions with Applications to Quantum Computing and LDCs. In *Proceedings of the 49th Symposium on Foundations of Computer Science (FOCS)*, pages 477–486. IEEE, 2008. doi:10.1109/FOCS.2008.45.

15 Itai Benjamini, Gil Cohen, and Igor Shinkar. Bi-Lipschitz Bijection Between the Boolean Cube and the Hamming Ball. In *Proceedings of the 55th Symposium on Foundations of Computer Science (FOCS)*, pages 81–89. IEEE, 2014. doi:10.1109/FOCS.2014.17.

16 Lucas Boczkowski, Iordanis Kerenidis, and Frédéric Magniez. Streaming Communication Protocols. *ACM Transactions on Computation Theory*, 10(4):19:1–19:21, 2018. doi:10.1145/3276748.

17 Ralph Bottesch, Dmitry Gavinsky, and Hartmut Klauck. Correlation in Hard Distributions in Communication Complexity. In *Proceedings of the 19th International Workshop on Randomization and Computation (RANDOM)*, pages 544–572. Schloss Dagstuhl, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.544.

18 Mark Braverman, Faith Ellen, Rotem Oshman, Toniann Pitassi, and Vinod Vaikuntanathan. A Tight Bound for Set Disjointness in the Message-Passing Model. In *Proceedings of the 54th Symposium on Foundations of Computer Science (FOCS)*, pages 668–677. IEEE, 2013. doi:10.1109/FOCS.2013.77.

19 Mark Braverman, Ankit Garg, Young Kun-Ko, Jieming Mao, and Dave Touchette. Near-Optimal Bounds on the Bounded-Round Quantum Communication Complexity of Disjointness. *SIAM Journal on Computing*, 47(6):2277–2314, 2018. doi:10.1137/16M1061400.

20 Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From Information to Exact Communication. In *Proceedings of the 45th Symposium on Theory of Computing (STOC)*, pages 151–160. ACM, 2013. doi:10.1145/2488608.2488628.

21 Mark Braverman and Ankur Moitra. An Information Complexity Approach to Extended Formulations. In *Proceedings of the 45th Symposium on Theory of Computing (STOC)*, pages 161–170. ACM, 2013. doi:10.1145/2488608.2488629.

22 Mark Braverman and Rotem Oshman. On Information Complexity in the Broadcast Model. In *Proceedings of the 34th Symposium on Principles of Distributed Computing (PODC)*, pages 355–364. ACM, 2015. doi:10.1145/2767386.2767425.

23 Mark Braverman and Rotem Oshman. A Rounds vs. Communication Tradeoff for Multi-Party Set Disjointness. In *Proceedings of the 58th Symposium on Foundations of Computer Science (FOCS)*, pages 144–155. IEEE, 2017. doi:10.1109/FOCS.2017.22.

24 Joshua Brody, Amit Chakrabarti, Ranganath Kondapally, David Woodruff, and Grigory Yaroslavtsev. Beyond Set Disjointness: The Communication Complexity of Finding the Intersection. In *Proceedings of the 33rd Symposium on Principles of Distributed Computing (PODC)*, pages 106–113. ACM, 2014. doi:10.1145/2611462.2611501.

25 Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. Classical Communication and Computation. In *Proceedings of the 30th Symposium on Theory of Computing (STOC)*, pages 63–68. ACM, 1998. doi:10.1145/276698.276713.

26 Harry Buhrman, David Garcia-Soriano, Arie Matsliah, and Ronald de Wolf. The Non-Adaptive Query Complexity of Testing k-Parities. *Chicago Journal of Theoretical Computer Science*, 2013(6):1–11, 2013. doi:10.4086/cjtcs.2013.006.

27 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-Optimal Lower Bounds on the Multi-Party Communication Complexity of Set Disjointness. In *Proceedings of the 18th Conference on Computational Complexity*, pages 107–117. IEEE, 2003. doi:10.1109/CCC.2003.1214414.

28 Arkadev Chattopadhyay and Anil Ada. Multiparty Communication Complexity of Disjointness. Technical Report TR08-002, Electronic Colloquium on Computational Complexity (ECCC), 2008. URL: https://eccc.weizmann.ac.il//eccc-reports/2008/TR08-002/.

29 Lijie Chen. On The Hardness of Approximate and Exact (Bichromatic) Maximum Inner Product. In *Proceedings of the 33rd Computational Complexity Conference (CCC)*, pages 14:1–14:45. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.CCC.2018.14.

30 Yuval Dagan, Yuval Filmus, Hamed Hatami, and Yaqiao Li. Trading Information Complexity for Error. *Theory of Computing*, 14(1):1–73, 2018. doi:10.4086/toc.2018.v014a006.

31 Anirban Dasgupta, Ravi Kumar, and D. Sivakumar. Sparse and Lopsided Set Disjointness via Information Theory. In *Proceedings of the 16th International Workshop on Randomization and Computation (RANDOM)*, pages 517–528. Springer, 2012. doi:10.1007/978-3-642-32512-0_44.

32 Anindya De and Thomas Watson. Extractors and Lower Bounds for Locally Samplable Sources. *ACM Transactions on Computation Theory*, 4(1):3:1–3:21, 2012. doi:10.1145/2141938.2141941.

33 Yuval Filmus, Hamed Hatami, Yaqiao Li, and Suzin You. Information Complexity of the AND Function in the Two-Party and Multi-Party Settings. In *Proceedings of the 23rd International Computing and Combinatorics Conference (COCOON)*, pages 200–211. Springer, 2017. doi:10.1007/978-3-319-62389-4_17.

34 Dmitry Gavinsky. Communication Complexity of Inevitable Intersection. Technical Report abs/1611.08842, arXiv, 2016. arXiv:1611.08842.

35 Dmitry Gavinsky and Alexander Sherstov. A Separation of NP and coNP in Multiparty Communication Complexity. *Theory of Computing*, 6(1):227–245, 2010. doi:10.4086/toc.2010.v006a010.

36 Oded Goldreich, Shafi Goldwasser, and Asaf Nussboim. On the Implementation of Huge Random Objects. *SIAM Journal on Computing*, 39(7):2761–2822, 2010. doi:10.1137/080722771.

37 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles Are Nonnegative Juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016. doi:10.1137/15M103145X.

38 Mika Göös, Toniann Pitassi, and Thomas Watson. Zero-Information Protocols and Unambiguity in Arthur–Merlin Communication. *Algorithmica*, 76(3):684–719, 2016. doi:10.1007/s00453-015-0104-9.

39 Mika Göös and Thomas Watson. Communication Complexity of Set-Disjointness for All Probabilities. *Theory of Computing*, 12(9):1–23, 2016. doi:10.4086/toc.2016.v012a009.

40 Vince Grolmusz. The BNS Lower Bound for Multi-Party Protocols Is Nearly Optimal. *Information and Computation*, 112(1):51–54, 1994. doi:10.1006/inco.1994.1051.

41 André Gronemeier. Asymptotically Optimal Lower Bounds on the NIH-Multi-Party Information Complexity of the AND-Function and Disjointness. In *Proceedings of the 26th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 505–516. Schloss Dagstuhl, 2009. doi:10.4230/LIPIcs.STACS.2009.1846.

42 Johan Håstad and Avi Wigderson. The Randomized Communication Complexity of Set Disjointness. *Theory of Computing*, 3(1):211–219, 2007. doi:10.4086/toc.2007.v003a011.

43 Peter Høyer and Ronald de Wolf. Improved Quantum Communication Complexity Bounds for Disjointness and Equality. In *Proceedings of the 19th Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 299–310. Springer, 2002. doi:10.1007/3-540-45841-7_24.

44 Rahul Jain and Hartmut Klauck. The Partition Bound for Classical Communication Complexity and Query Complexity. In *Proceedings of the 25th Conference on Computational Complexity (CCC)*, pages 247–258. IEEE, 2010. doi:10.1109/CCC.2010.31.

45 Rahul Jain, Hartmut Klauck, and Ashwin Nayak. Direct Product Theorems for Classical Communication Complexity via Subdistribution Bounds. In *Proceedings of the 40th Symposium on Theory of Computing (STOC)*, pages 599–608. ACM, 2008. doi:10.1145/1374376.1374462.

46 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A Lower Bound for the Bounded Round Quantum Communication Complexity of Set Disjointness. In *Proceedings of the 44th Symposium on Foundations of Computer Science (FOCS)*, pages 220–229. IEEE, 2003. doi:10.1109/SFCS.2003.1238196.

47 Rahul Jain, Yaoyun Shi, Zhaohui Wei, and Shengyu Zhang. Efficient Protocols for Generating Bipartite Classical Distributions and Quantum States. *IEEE Transactions on Information Theory*, 59(8):5171–5178, 2013. doi:10.1109/TIT.2013.2258372.

48 T.S. Jayram. Hellinger strikes back: A note on the multi-party information complexity of AND. In *Proceedings of the 13th International Workshop on Randomization and Computation (RANDOM)*, pages 562–573. Springer, 2009. doi:10.1007/978-3-642-03685-9_42.

49 Bala Kalyanasundaram and Georg Schnitger. The Probabilistic Communication Complexity of Set Intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992. doi:10.1137/0405044.

50 Hartmut Klauck. Rectangle Size Bounds and Threshold Covers in Communication Complexity. In *Proceedings of the 18th Conference on Computational Complexity (CCC)*, pages 118–134. IEEE, 2003. doi:10.1109/CCC.2003.1214415.

51 Hartmut Klauck. A Strong Direct Product Theorem for Disjointness. In *Proceedings of the 42nd Symposium on Theory of Computing (STOC)*, pages 77–86. ACM, 2010. doi:10.1145/1806689.1806702.

52 Hartmut Klauck, Ashwin Nayak, Amnon Ta-Shma, and David Zuckerman. Interaction in Quantum Communication. *IEEE Transactions on Information Theory*, 53(6):1970–1982, 2007. doi:10.1109/TIT.2007.896888.

53 Hartmut Klauck and Supartha Podder. New Bounds for the Garden-Hose Model. In *Proceedings of the 34th International Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 481–492. Schloss Dagstuhl, 2014. doi:10.4230/LIPIcs.FSTTCS.2014.481.

54 Hartmut Klauck, Robert Spalek, and Ronald de Wolf. Quantum and Classical Strong Direct Product Theorems and Optimal Time-Space Tradeoffs. *SIAM Journal on Computing*, 36(5):1472–1493, 2007. doi:10.1137/05063235X.

55 Gillat Kol, Shay Moran, Amir Shpilka, and Amir Yehudayoff. Approximate Nonnegative Rank Is Equivalent to the Smooth Rectangle Bound. *Computational Complexity*, 28(1):1–25, 2019. doi:10.1007/s00037-018-0176-4.

56 Eyal Kushilevitz and Enav Weinreb. The Communication Complexity of Set-Disjointness with Small Sets and 0-1 Intersection. In *Proceedings of the 50th Symposium on Foundations of Computer Science (FOCS)*, pages 63–72. IEEE, 2009. doi:10.1109/FOCS.2009.15.

57 Troy Lee and Adi Shraibman. Disjointness is Hard in the Multiparty Number-on-the-Forehead Model. *Computational Complexity*, 18(2):309–336, 2009. doi:10.1007/s00037-009-0276-2.

58 Shachar Lovett and Emanuele Viola. Bounded-Depth Circuits Cannot Sample Good Codes. *Computational Complexity*, 21(2):245–266, 2012. doi:10.1007/s00037-012-0039-3.

59 Mihai Patrascu. Unifying the Landscape of Cell-Probe Lower Bounds. *SIAM Journal on Computing*, 40(3):827–847, 2011. doi:10.1137/09075336X.

60 Vladimir Podolskii and Alexander Sherstov. Inner Product and Set Disjointness: Beyond Logarithmically Many Parties. Technical Report abs/1711.10661, arXiv, 2017. arXiv:1711.10661.

61 Anup Rao and Amir Yehudayoff. Simplified Lower Bounds on the Multiparty Communication Complexity of Disjointness. In *Proceedings of the 30th Computational Complexity Conference (CCC)*, pages 88–101. Schloss Dagstuhl, 2015. doi:10.4230/LIPIcs.CCC.2015.88.

62 Alexander Razborov. On the Distributional Complexity of Disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992. doi:10.1016/0304-3975(92)90260-M.

63 Alexander Razborov. Quantum Communication Complexity of Symmetric Predicates. *Izvestiya: Mathematics*, 67(1):145–159, 2003. doi:10.1070/IM2003v067n01ABEH000422.

64 Aviad Rubinstein. Hardness of Approximate Nearest Neighbor Search. In *Proceedings of the 50th Symposium on Theory of Computing (STOC)*, pages 1260–1268. ACM, 2018. doi:10.1145/3188745.3188916.

65 Mert Saglam and Gábor Tardos. On the Communication Complexity of Sparse Set Disjointness and Exists-Equal Problems. In *Proceedings of the 54th Symposium on Foundations of Computer Science (FOCS)*, pages 678–687. IEEE, 2013. doi:10.1109/FOCS.2013.78.

66 Alexander Sherstov. The Pattern Matrix Method. *SIAM Journal on Computing*, 40(6):1969–2000, 2011. doi:10.1137/080733644.

67 Alexander Sherstov. Strong Direct Product Theorems for Quantum Communication and Query Complexity. *SIAM Journal on Computing*, 41(5):1122–1165, 2012. doi:10.1137/110842661.

68 Alexander Sherstov. Communication Lower Bounds Using Directional Derivatives. *Journal of the ACM*, 61(6):1–71, 2014. doi:10.1145/2629334.

69 Alexander Sherstov. The Multiparty Communication Complexity of Set Disjointness. *SIAM Journal on Computing*, 45(4):1450–1489, 2016. doi:10.1137/120891587.

70 Yaoyun Shi and Yufan Zhu. Quantum Communication Complexity of Block-Composed Functions. *Quantum Information and Computation*, 9(5–6):444–460, 2009.

71 Pascal Tesson. *Computational Complexity Questions Related to Finite Monoids and Semigroups*. PhD thesis, McGill University, 2003.

72 Emanuele Viola. Extractors for Turing-Machine Sources. In *Proceedings of the 16th International Workshop on Randomization and Computation (RANDOM)*, pages 663–671. Springer, 2012. doi:10.1007/978-3-642-32512-0_56.

73 Emanuele Viola. The Complexity of Distributions. *SIAM Journal on Computing*, 41(1):191–218, 2012. doi:10.1137/100814998.

74 Emanuele Viola. Extractors for Circuit Sources. *SIAM Journal on Computing*, 43(2):655–672, 2014. doi:10.1137/11085983X.

75 Emanuele Viola. Quadratic Maps Are Hard to Sample. *ACM Transactions on Computation Theory*, 8(4):18:1–18:4, 2016. doi:10.1145/2934308.

76 Emanuele Viola. Sampling Lower Bounds: Boolean Average-Case and Permutations. Technical Report TR18-060, Electronic Colloquium on Computational Complexity (ECCC), 2018. URL: https://eccc.weizmann.ac.il/report/2018/060.

77 Thomas Watson. Time Hierarchies for Sampling Distributions. *SIAM Journal on Computing*, 43(5):1709–1727, 2014. doi:10.1137/120898553.

78 Thomas Watson. Nonnegative Rank vs. Binary Rank. *Chicago Journal of Theoretical Computer Science*, 2016(2):1–13, 2016. doi:10.4086/cjtcs.2016.002.

79 Thomas Watson. Communication Complexity with Small Advantage. In *Proceedings of the 33rd Computational Complexity Conference (CCC)*, pages 9:1–9:17. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.CCC.2018.9.

80 Omri Weinstein and David Woodruff. The Simultaneous Communication of Disjointness with Applications to Data Streams. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 1082–1093. Springer, 2015. doi:10.1007/978-3-662-47672-7_88.

Approximating the Noise Sensitivity of a Monotone Boolean Function

Ronitt Rubinfeld
CSAIL at MIT, Cambridge, MA, USA
Blavatnik School of Computer Science at Tel Aviv University, Israel
https://people.csail.mit.edu/ronitt/
ronitt@csail.mit.edu

Arsen Vasilyan
CSAIL at MIT, Cambridge, MA, USA
vasilyan@mit.edu

Abstract

The *noise sensitivity* of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ is one of its fundamental properties. For noise parameter δ, the noise sensitivity is denoted as $NS_\delta[f]$. This quantity is defined as follows: First, pick $x = (x_1, \ldots, x_n)$ uniformly at random from $\{0,1\}^n$, then pick z by flipping each x_i independently with probability δ. $NS_\delta[f]$ is defined to equal $\Pr[f(x) \neq f(z)]$. Much of the existing literature on noise sensitivity explores the following two directions: (1) Showing that functions with low noise-sensitivity are structured in certain ways. (2) Mathematically showing that certain classes of functions have low noise sensitivity. Combined, these two research directions show that certain classes of functions have low noise sensitivity and therefore have useful structure.

The fundamental importance of noise sensitivity, together with this wealth of structural results, motivates the algorithmic question of approximating $NS_\delta[f]$ given an oracle access to the function f. We show that the standard sampling approach is essentially optimal for general Boolean functions. Therefore, we focus on estimating the noise sensitivity of *monotone* functions, which form an important subclass of Boolean functions, since many functions of interest are either monotone or can be simply transformed into a monotone function (for example the class of *unate* functions consists of all the functions that can be made monotone by reorienting some of their coordinates [21]).

Specifically, we study the algorithmic problem of approximating $NS_\delta[f]$ for monotone f, given the promise that $NS_\delta[f] \geq 1/n^C$ for constant C, and for δ in the range $1/n \leq \delta \leq 1/2$. For such f and δ, we give a randomized algorithm performing $O\left(\frac{\min(1,\sqrt{n}\delta \log^{1.5} n)}{NS_\delta[f]} \text{poly}\left(\frac{1}{\epsilon}\right)\right)$ queries and approximating $NS_\delta[f]$ to within a multiplicative factor of $(1 \pm \epsilon)$. Given the same constraints on f and δ, we also prove a lower bound of $\Omega\left(\frac{\min(1,\sqrt{n}\delta)}{NS_\delta[f] \cdot n^\xi}\right)$ on the query complexity of any algorithm that approximates $NS_\delta[f]$ to within any constant factor, where ξ can be any positive constant. Thus, our algorithm's query complexity is close to optimal in terms of its dependence on n.

We introduce a novel *descending-ascending view* of noise sensitivity, and use it as a central tool for the analysis of our algorithm. To prove lower bounds on query complexity, we develop a technique that reduces computational questions about query complexity to combinatorial questions about the existence of "thin" functions with certain properties. The existence of such "thin" functions is proved using the probabilistic method. These techniques also yield new lower bounds on the query complexity of approximating other fundamental properties of Boolean functions: the *total influence* and the *bias*.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation

Keywords and phrases Monotone Boolean functions, noise sensitivity, influence

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.52

Category RANDOM

Related Version Full version: https://arxiv.org/abs/1904.06745



© Ronitt Rubinfeld and Arsen Vasilyan;
licensed under Creative Commons License CC-BY

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh; Article No. 52; pp. 52:1–52:17



Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Funding *Ronitt Rubinfeld*: NSF grants CCF-1650733, CCF-1733808, IIS-1741137 and CCF-1740751 *Arsen Vasilyan*: NSF grant IIS-1741137, EECS SuperUROP program, the MIT Summer UROP program and the DeFlorez Endowment Fund

Acknowledgements We are grateful to the anonymous referees, Daniel Grier and MIT EECS Communication Lab for helpful comments and suggestions.

1 Introduction

Noise sensitivity is a property of any Boolean function $f : \{0,1\}^n \to \{0,1\}$ defined as follows: First, pick $x = (x_1, \ldots, x_n)$ uniformly at random from $\{0,1\}^n$, then pick z by flipping each x_i independently with probability δ. Here δ, the noise parameter, is a given positive constant no greater than $1/2$ (and at least $1/n$ in the interesting cases). With the above distributions on x and z, the noise sensitivity of f, denoted as $NS_\delta[f]$, is defined as follows:

$$NS_\delta[f] \stackrel{\text{def}}{=} \Pr[f(x) \neq f(z)] \tag{1}$$

Noise sensitivity was first explicitly defined by Benjamini, Kalai and Schramm in [3], and has been the focus of multiple papers: e.g. [3, 7, 8, 10, 12, 17, 22]. It has been applied to learning theory [4, 7, 8, 9, 10, 11, 15], property testing [1, 2], hardness of approximation [13, 16], hardness amplification [19], combinatorics [3, 12], distributed computing [18] and differential privacy [7]. Multiple properties and applications of noise sensitivity are summarized in [20] and [21]. Much of the existing literature on noise sensitivity explores the following directions: (1) Showing that functions with low noise-sensitivity are structured in certain ways. (2) Mathematically showing that certain classes of functions have low noise sensitivity. Combined, these two research directions show that certain classes of functions have low noise sensitivity and therefore have useful structure.

The fundamental importance of noise sensitivity, together with this wealth of structural results, motivates the algorithmic question of approximating $NS_\delta[f]$ given an oracle access to the function f. It can be shown that standard sampling techniques require $O\left(\frac{1}{NS_\delta[f]\epsilon^2}\right)$ queries to get a $(1+\epsilon)$-multiplicative approximation for $NS_\delta[f]$. In the full version of the paper, we show that this is optimal for a wide range of parameters of the problem. Specifically, it cannot be improved by more than a constant when ϵ is a sufficiently small constant, δ satisfies $1/n \leq \delta \leq 1/2$ and $NS_\delta[f]$ satisfies $\Omega\left(\frac{1}{2^n}\right) \leq NS_\delta[f] \leq O(1)$.

It is often the case that data possesses a known underlying structural property which makes the computational problem significantly easier to solve. A natural first such property to investigate is that of monotonicity, as a number of natural function families are made up of functions that are either monotone or can be simply transformed into a monotone function (for example the class of *unate* functions consists of all the functions that can be made monotone by reorienting some of their coordinates [21]). Therefore, we focus on estimating the noise sensitivity of monotone functions.

The approximation of the related quantity of total influence (henceforth just influence) of a monotone Boolean function in this model was previously studied by [24, 23][1]. Influence, denoted by $I[f]$, is defined as n times the probability that a random edge of the Boolean cube (x, y) is *influential*, which means that $f(x) \neq f(y)$. (This latter probability is sometimes referred to as the *average sensitivity*). It was shown in [24, 23] that one can approximate the influence of a monotone function f with only $\tilde{O}\left(\frac{\sqrt{n}}{I[f]\text{poly}(\epsilon)}\right)$ queries, which for constant ϵ beats the standard sampling algorithm by a factor of $\sqrt{n}$, ignoring logarithmic factors.

[1] [23] is the journal version of [24] and contains a different algorithm that yields sharper results. However, our algorithmic techniques build on the conference version [24].

Despite the fact that the noise sensitivity is closely connected to the influence [20, 21], the noise sensitivity of a function can be quite different from its influence. For instance, for the parity function of all n bits, the influence is n, but the noise sensitivity is $\frac{1}{2}(1-(1-2\delta)^n)$ (such disparities also hold for monotone functions, see for example the discussion of influence and noise sensitivity of the majority function in [21]). Therefore, approximating the influence by itself does not give one a good approximation to the noise sensitivity.

The techniques in [24, 23] also do not immediately generalize to the case of noise sensitivity. The result in [24, 23] is based on the observation that given a descending[2] path on the Boolean cube, at most one edge in it can be influential. Thus, to check if a descending path of any length contains an influential edge, it suffices to check the function values at the endpoints of the path. By sampling random descending paths, [24, 23] show that one can estimate the fraction of influential edges, which is proportional to the influence.

The most natural attempt to relate these path-based techniques with the noise sensitivity is to view it in the context of the following process: first one samples x randomly, then one obtains z by taking a random walk from x by going through all the indices in an arbitrary order and deciding whether to flip each with probability δ. The intermediate values in this process give us a natural path connecting x to z. However, this path is in general not descending, so it can, for example, cross an even number of influential edges, and then the function will have the same value on the two endpoints of this path. This prevents one from immediately applying the techniques from [24, 23].

We overcome this difficulty by introducing our main conceptual contribution: the *descending-ascending view* of noise sensitivity. In the process above, instead of going through all the indices in an arbitrary order, we first go through the indices i for which $x_i = 1$ and only then through the ones for which $x_i = 0$. This forms a path between x and z that has first a descending component and then an ascending component. Although this random walk is more amenable to an analysis using the path-based techniques of [24, 23], there are still non-trivial sampling questions involved in the design and analysis of our algorithm.

An immediate corollary of our result is a query complexity upper bound on estimating the gap between the noise stability of a Boolean function and one. The noise stability of a Boolean function f depends on a parameter ρ and is denoted by $\mathrm{Stab}_\rho[f]$ (for more information about noise stability, see [21]). One way $\mathrm{Stab}_\rho[f]$ can be defined is as the unique quantity satisfying the functional relation $\frac{1}{2}(1-\mathrm{Stab}_{1-2\delta}[f]) = NS_\delta[f]$ for all δ. This implies that by obtaining an approximation for $NS_\delta[f]$, one also achieves an approximation for $1-\mathrm{Stab}_{1-2\delta}[f]$.

1.1 Results

Our main algorithmic result is the following:

▶ **Theorem 1.** *Let δ be a parameter satisfying:*

$$\frac{1}{n} \leq \delta \leq \frac{1}{\sqrt{n}\log^{1.5} n}$$

Suppose, $f : \{0,1\}^n \to \{0,1\}$ is a monotone function and $NS_\delta[f] \geq \frac{1}{n^C}$ for some constant C.

Then, there is an algorithm that outputs an approximation to $NS_\delta[f]$ to within a multiplicative factor of $(1 \pm \epsilon)$, with success probability at least $2/3$. In expectation, the algorithm makes $O\left(\frac{\sqrt{n}\delta\log^{1.5} n}{NS_\delta[f]\epsilon^3}\right)$ queries to the function. Additionally, it runs in time polynomial in n.

[2] A path is *descending* if each subsequent vertex in it is dominated by all the previous ones in the natural partial order on the Boolean cube.

Note that computing noise-sensitivity using standard sampling[3] requires $O\left(\frac{1}{NS_\delta[f]\epsilon^2}\right)$ samples. Therefore, for a constant ϵ, we have the most dramatic improvement if $\delta = \frac{1}{n}$, in which case, ignoring constant and logarithmic factors, our algorithm outperforms standard sampling by a factor of $\sqrt{n}$.

As in [24], our algorithm requires that the noise sensitivity of the input function f is larger than a specific threshold $1/n^C$. Our algorithm is not sensitive to the value of C as long as it is a constant, and we think of $1/n^C$ as a rough initial lower bound known in advance.

We next give lower bounds for approximating three different parameters of monotone Boolean functions: the bias, the influence and the noise sensitivity. A priori, it is not clear what kind of lower bounds one could hope for. Indeed, determining whether a given function is the all-zeros function requires $\Omega(2^n)$ queries in the general function setting, but only 1 query (of the all-ones input), if the function is promised to be monotone. Nevertheless, we show that such a dramatic improvement for approximating these quantities is not possible.

For monotone functions, we are not aware of previous lower bounds on approximating the bias or noise sensitivity. Our lower bound on approximating influence is not comparable to the lower bounds in [24, 23], as we will elaborate shortly.

We now state our lower bound for approximating the noise sensitivity. Here and everywhere else, to "reliably distinguish" means to distinguish with probability at least 2/3.

▶ **Theorem 2.** *For all constants C_1 and C_2 satisfying $C_1 - 1 > C_2 \geq 0$, for an infinite number of values of n the following is true: For all δ satisfying $1/n \leq \delta \leq 1/2$, given a monotone function $f : \{0,1\}^n \to \{0,1\}$, one needs at least $\Omega\left(\frac{n^{C_2}}{e^{\sqrt{C_1 \log n/2}}}\right)$ queries to reliably distinguish between the following two cases: (i) f has noise sensitivity between $\Omega(1/n^{C_1+1})$ and $O(1/n^{C_1})$ and (ii) f has noise sensitivity larger than $\Omega(\min(1, \delta\sqrt{n})/n^{C_2})$.*

▶ Remark 3. For any positive constant ξ, we have that $e^{\sqrt{C_1 \log n/2}} \leq n^\xi$.

▶ Remark 4. The range of the parameter δ can be divided into two regions of interest. In the region $1/n \leq \delta \leq 1/(\sqrt{n}\log n)$, the algorithm from Theorem 1 can distinguish the two cases above with only $\tilde{O}(n^{C_2})$ queries. Therefore its query complexity is optimal up to a factor of $\tilde{O}(e^{\sqrt{C_1 \log n/2}})$. Similarly, in the region $1/(\sqrt{n}\log n) \leq \delta \leq 1/2$, the standard sampling algorithm can distinguish the two distributions above with only $\tilde{O}(n^{C_2})$ queries. Therefore in this region of interest, standard sampling is optimal up to a factor of $\tilde{O}(e^{\sqrt{C_1 \log n/2}})$.

We define the *bias* of a Boolean function as $B[f] \stackrel{\text{def}}{=} \Pr[f(x) = 1]$, where x is chosen uniformly at random from $\{0,1\}^n$. It is arguably the most basic property of a Boolean function, so we consider the question of how quickly it can be approximated for monotone functions. To approximate the bias up to a multiplicative factor of $(1 \pm \epsilon)$ using standard sampling, one needs $O(1/(B[f]\epsilon^2))$ queries. We obtain a lower bound for this task similar to the previous theorem:

▶ **Theorem 5.** *For all constants C_1 and C_2 satisfying $C_1 - 1 > C_2 \geq 0$, for an infinite number of values of n the following is true: Given a monotone function $f : \{0,1\}^n \to \{0,1\}$, one needs at least $\Omega\left(\frac{n^{C_2}}{e^{\sqrt{C_1 \log n/2}}}\right)$ queries to reliably distinguish between the following two cases: (i) f has bias of $\Theta(1/n^{C_1})$ (ii) f has bias larger than $\Omega(1/n^{C_2})$.*

[3] Standard sampling refers to the algorithm that picks $O\left(\frac{1}{NS_\delta[f]\epsilon^2}\right)$ pairs x and z as in the definition of noise sensitivity and computes the fraction of pairs for which $f(x) \neq f(z)$.

Finally, we prove a lower bound for approximating influence:

▶ **Theorem 6.** *For all constants C_1 and C_2 satisfying $C_1 - 1 > C_2 \geq 0$, for an infinite number of values of n the following is true: Given a monotone function $f : \{0,1\}^n \to \{0,1\}$, one needs at least $\Omega\left(\frac{n^{C_2}}{e^{\sqrt{C_1 \log n/2}}}\right)$ queries to reliably distinguish between the following two cases: (i) f has influence between $\Omega(1/n^{C_1})$ and $O(n/n^{C_1})$ (ii) f has influence larger than $\Omega(\sqrt{n}/n^{C_2})$.*

This gives us a new sense in which the algorithm family in [24, 23] is close to optimal, because for a function f with influence $\Omega(\sqrt{n}/n^{C_2})$ this algorithm makes $\tilde{O}(n^{C_2})$ queries to estimate the influence up to any constant factor.

Our lower bound is incomparable to the lower bound in [24], which makes the stronger requirement that $I[f] \geq \Omega(1)$, but gives a bound that is only a polylogarithmic factor smaller than the runtime of the algorithm in [24, 23]. There are many possibilities for algorithmic bounds that were compatible with the lower bound in [24, 23], but are eliminated with our lower bound. For instance, prior to this work, it was conceivable that an algorithm making as little as $O(\sqrt{n})$ queries could give a constant factor approximation to the influence of **any** monotone input function whatsoever. Our lower bound shows that not only is this impossible, no algorithm that makes $O(n^{C_2})$ queries for any constant C_2 can accomplish this either.

1.2 Algorithm overview

Here, we give the algorithm in Theorem 1 together with the subroutines it uses. Additionally, we give an informal overview of the proof of correctness and the analysis of running time and query complexity, which are presented in more detail in Section 3.

First of all, recall that $NS_\delta[f] = \Pr[f(x) \neq f(z)]$ by Equation 1. Using a standard pairing argument, we argue that $NS_\delta[f] = 2 \cdot \Pr[f(x) = 1 \wedge f(z) = 0]$. In other words, we can focus only on the case when the value of the function flips from one to zero.

We introduce the *descending-ascending view of noise sensitivity* (described more formally in Subsection 3.1), which, roughly speaking, views the noise process as decomposed into a first phase that operates only on the locations in x that are 1, and a second phase that operates only on the locations in x that are set to 0. Formally, we define the noise process D in Algorithm 1.

This process gives us a path from x to z that can be decomposed into two segments, such that the first part, P_1, descends in the hypercube, and the second part P_2 ascends in the hypercube.

Algorithm 1 Process D.

1. Pick x uniformly at random from $\{0,1\}^n$. Let S_0 be the set of indexes i for which $x_i = 0$, and conversely let S_1 be the rest of indexes.
2. **Phase 1:** go through all the indexes in S_1 in a random order, and flip each with probability δ. Form the descending path P_1 from all the intermediate results. Call the endpoint y.
3. **Phase 2:** start at y, and flip each index in S_0 with probability δ. As before, all the intermediate results form an ascending path P_2, which ends in z.
4. Output P_1, P_2, x, y and z.

Since f is monotone, for $f(x) = 1$ and $f(z) = 0$ to be the case, it is necessary, though not sufficient, that $f(x) = 1$ and $f(y) = 0$, which happens whenever P_1 hits an influential edge. Therefore we break the task of estimating the probability of $f(x) \neq f(z)$ into computing the product of:

- The probability that P_1 hits an influential edge, specifically, the probability that $f(x) = 1$ and $f(y) = 0$, which we refer to as p_A.
- The probability that P_2 does not hit any influential edge, given that P_1 hits an influential edge: specifically, the probability that given $f(x) = 1$ and $f(y) = 0$, it is the case that $f(z) = 0$. We refer to this probability as p_B.

The above informal definitions of p_A and p_B ignore some technical complications. Specifically, the impact of certain "bad events" is considered in our analysis. We redefine p_A and p_B precisely in Subsection 3.2.1.

To define those bad events, we use the following two values, which we reference in our algorithms: t_1 and t_2. Informally, t_1 and t_2 have the following meaning. A typical vertex x of the hypercube has Hamming weight $L(x)$ between $n/2 - t_1$ and $n/2 + t_1$. A typical Phase 1 path from process D will have length at most t_2. To achieve this, we assign $t_1 \stackrel{\text{def}}{=} \eta_1\sqrt{n \log n}$ and $t_2 \stackrel{\text{def}}{=} n\delta(1 + 3\eta_2 \log n)$, where η_1 and η_2 are certain constants.

We also define M to be the set of edges $e = (v_1, v_2)$, for which both $L(v_1)$ and $L(v_2)$ are between and $n/2 - t_1$ and $n/2 + t_1$. Most of the edges in the hypercube are in M, which is used by our algorithm and the run-time analysis.

Our analysis requires that only $\delta \leq 1/(\sqrt{n}\log^{1.5} n)$ as in the statement of Theorem 1, however the utility of the ascending-descending view can be most clearly motivated when $\delta \leq 1/(\sqrt{n}\log^2 n)$. Specifically, given that $\delta \leq 1/(\sqrt{n}\log^2 n)$, it is the case that t_2 will be shorter than $O(\sqrt{n}/\log n)$. Therefore, typically, the path P_1 is also shorter than $O(\sqrt{n}/\log n)$. Similar short descending paths on the hypercube have been studied before: In [24], paths of such lengths were used to estimate the number of influential edges by analyzing the probability that a path would hit such an edge. One useful insight given by [24] is that the probability of hitting almost every single influential edge is roughly the same.

However, the results in [24] cannot be immediately applied to analyze P_1, because (i) P_1 does not have a fixed length, but rather its lengths form a probability distribution, (ii) this probability distribution also depends on the starting point x of P_1. We build upon the techniques in [24] to overcome these difficulties, and prove that again, roughly speaking, for almost every single influential edge, the probability that P_1 hits it depends very little on the location of the edge, and our proof also computes this probability. This allows us to prove that $p_A \approx \delta I[f]/2$. Then, using the algorithm in [24] to estimate $I[f]$, we thereby estimate p_A.

Regarding p_B, we estimate it by approximately sampling paths P_1 and P_2 that would arise from process D, conditioned on that P_1 hits an influential edge. To that end, we first sample an influential edge e that P_1 hits. Since P_1 hits almost every single influential edge with roughly the same probability, we do it by sampling e approximately uniformly from among influential edges. For the latter task, we build upon the result in [24] as follows: As we have already mentioned, the algorithm in [24] samples descending paths of a fixed length to estimate the influence. For those paths that start at an x for which $f(x) = 1$ and end at a z for which $f(z) = 0$, we add a binary search step in order to locate the influential edge e that was hit by the path.

Thus, we have the following algorithm $\mathcal{A}$ (see Algorithm 2), which takes oracle access to a function f and an approximation parameter ϵ as input. In the case of success, it outputs an influential edge that is roughly uniformly distributed.

Algorithm 2 Algorithm $\mathcal{A}$ (given oracle access to a monotone function $f : \{0,1\}^n \to \{0,1\}$ and a parameter ϵ).

1. Assign $w = \frac{\epsilon}{3100\eta_1}\sqrt{\frac{n}{\log n}}$
2. Pick x uniformly at random from $\{0,1\}^n$.
3. Perform a descending walk P_1 downwards in the hypercube starting at x. Stop at a vertex y either after w steps, or if you hit the all-zeros vertex. Query the value of f only at the endpoints x and y of this path.
4. If $f(x) = f(y)$ output FAIL.
5. If $f(x) \neq f(y)$ perform a binary search on the path P_1 and find an influential edge e_{inf}.
6. If $e_{inf} \in M$ return e_{inf}. Otherwise output FAIL.

Finally, once we have obtained a roughly uniformly random influential edge e, we sample a path P_1 from among those that hit it. An obvious way to try to quickly sample such a path is to perform two random walks of lengths w_1 and w_2 in opposite directions from the endpoints of the edge, and then concatenate them into one path. However, to do this, one needs to somehow sample the lengths w_1 and w_2. This problem is not trivial, since longer descending paths are more likely to hit an influential edge, which biases the distribution of the path lengths towards longer ones.

To generate w_1 and w_2 according to the proper distribution, we first sample a path P_1 hitting any edge at the same *layer*[4] Λ_e as e. We accomplish this by designing an algorithm that uses rejection sampling. The algorithm samples short descending paths from some conveniently chosen distribution, until it gets a path hitting the desired layer.

We now describe the algorithm in more detail. Recall that we use $L(x)$ to denote the Hamming weight (which we also call the *level*) of x, which equals the number of indices i on which $x_i = 1$, and we use the symbol Λ_e to denote the whole *layer* of edges that have the same endpoint levels as e. The algorithm $\mathcal{W}$ described in Algorithm 3 takes an influential edge e as an input and samples the lengths w_1 and w_2.

Algorithm 3 Algorithm $\mathcal{W}$ (given an edge $e \stackrel{\text{def}}{=} (v_1, v_2)$ so $v_2 \preceq v_1$).

1. Pick an integer l uniformly at random among the integers in $[L(v_1), L(v_1) + t_2 - 1]$. Pick a vertex x randomly at level l.
2. As in phase 1 of the noise sensitivity process, traverse in random order through the indices of x and for each index that equals to one, flip it with probability δ. The intermediate results form a path P_1, and we call its endpoint y.
3. If P_1 does not intersect Λ_e go to step 1.
4. Otherwise, output $w_1 = L(x) - L(v_1)$ and $w_2 = L(v_2) - L(y)$.

Recall that t_2 has a technical role and is defined to be equal $n\delta(1 + 3\eta_2 \log n)$, where η_2 is a certain constant. t_2 is chosen to be long enough that it is longer than most paths P_1, but short enough to make the sampling in $\mathcal{W}$ efficient. Since the algorithm involves short descending paths, we analyze this algorithm building upon the techniques we used to approximate p_A.

[4] We say that edges e_1 and e_2 are on the same layer if and only if their endpoints have the same Hamming weights. We denote the layer an edge e belongs to as Λ_e.

After obtaining a random path going through the same layer as e, we show how to transform it, using the symmetries of the hypercube, into a a random path P_1 going through e itself. Additionally, given the endpoint of P_1, we sample the path P_2 just as in the process D.

Formally, the algorithm $\mathcal{B}$ (see Algorithm 4) takes an influential edge e and returns a descending path P_1 that goes through e and an adjacent ascending path P_2, together with the endpoints of these paths.

Algorithm 4 Algorithm $\mathcal{B}$ (given an influential edge $e \stackrel{\text{def}}{=} (v_1, v_2)$ so $v_2 \preceq v_1$).

1. Use $\mathcal{W}(e)$ to sample w_1 and w_2.
2. Perform an ascending random walk of length w_1 starting at v_1 and call its endpoint x. Similarly, perform a descending random walk starting at v_2 of length w_2, call its endpoint y.
3. Define P_1 as the descending path that results between x and y by concatenating the two paths from above, oriented appropriately, and edge e.
4. Define P_2 just as in phase 2 of our process starting at y. Consider in random order all the zero indices y has in common with x and flip each with probability δ.
5. Return P_1 ,P_2, x, y and z.

We then use sampling to estimate which fraction of the paths P_2 continuing these P_1 paths does not hit an influential edge. This allows us to estimate p_B, which, combined with our estimate for p_A, gives us an approximation for $NS_\delta[f]$.

Formally, we put all the previously defined subroutines together into the randomized Algorithm 5 that takes oracle access to a function f together with an approximation parameter ϵ and outputs an approximation to $NS_\delta[f]$:

Algorithm 5 Algorithm for estimating noise sensitivity. (given oracle access to a monotone function $f : \{0,1\}^n \to \{0,1\}$, and a parameter ϵ).

1. Using the algorithm from [24] as described in Theorem 14, compute an approximation to the influence of f to within a multiplicative factor of $(1 \pm \epsilon/33)$. This gives us $\tilde{I}$.
2. Compute $\tilde{p}_A := \delta\tilde{I}/2$.
3. Initialize $\alpha := 0$ and $\beta := 0$. Repeat the following until $\alpha = \frac{768 \ln 200}{\epsilon^2}$.
   - Use algorithm $\mathcal{A}$ from Lemma 20 repeatedly to successfully sample an edge e.
   - From Lemma 25 use the algorithm $\mathcal{B}$, giving it e as input, and sample P_1, P_2, x, y and z.
   - If it is the case that $f(x) = 1$ and $f(z) = 0$, then $\alpha := \alpha + 1$.
   - $\beta := \beta + 1$.
4. Set $\tilde{p}_B = \frac{\alpha}{\beta}$.
5. Return $2\tilde{p}_A\tilde{p}_B$.

1.3 Lower bound techniques

We use the same technique to lower bound the query complexity of approximating any of the following three quantities: the noise sensitivity, influence and bias.

For concreteness, let us first focus on approximating the bias. Recall that one can distinguish the case where the bias is 0 from the bias being $1/2^n$ using a single query. Nevertheless, we show that for the most part, no algorithm for estimating the bias can do much better than the random sampling approach.

We construct two probability distributions D_1^B and D_2^B that are relatively hard to distinguish but have drastically different biases. To create them, we fix some threshold l_0 and then construct a special monotone function F^B, which has the following two properties: (1) It has a high bias. (2) It equals to one on only a relatively small fraction of points on the level l_0. We refer to functions satisfying (2) as "thin" functions. We will explain later how to obtain such a function F^B. We pick a function from D_2^B by taking F^B, randomly permuting the indices of its input, and finally "truncating" it by setting it to one on all values of x, which have Hamming weight greater than l_0.

We form D_1^B even more simply. We take the all-zeros function and truncate it at the same threshold l_0. The threshold l_0 is chosen in a way that this function in D_1^B has a sufficiently small bias. Thus D_1^B consists of only a single function.

The purpose of truncation is to prevent a distinguisher from gaining information by accessing the values of the function on the high-level vertices of the hypercube. Indeed, if there was no truncation, one could tell whether they have access to the all-zeros function by simply querying it on the all-ones input. Since F^B is monotone, if it equals to one on at least one input, then it has to equal one on the all-ones input.

The proof has two main lemmas: The first one is computational and says that if F^B is "thin" then D_1^B and D_2^B are hard to reliably distinguish. To prove the first lemma, we show that one could transform any adaptive algorithm for distinguishing D_1^B from D_2^B into an algorithm that is just as effective, is non-adaptive and queries points only on the layer l_0.

To show this, we observe that, because of truncation, distinguishing a function in D_2^B from a function in D_1^B is in a certain sense equivalent to finding a point with level at most l_0 on which the given function evaluates to one. We argue that for this setting, adaptivity does not help. Additionally, if $x \preceq y$ and both of them have levels at most l_0 then, since f is monotone, $f(x) = 1$ implies that $f(y) = 1$ (but not necessarily the other way around). Therefore, for finding a point on which the function evaluates to one, it is never more useful to query x instead of y.

Once we prove that no algorithm can do better than a non-adaptive algorithm that only queries points on the level l_0, we use a simple union bound to show that any such algorithm cannot be very effective for distinguishing our distributions.

Finally, to construct F^B, we need to show that there exist functions that are "thin" and simultaneously have a high bias. This is a purely combinatorial question and is proven in our second main lemma. We build upon Talagrand random functions that were first introduced in [25]. In [17] it was shown that they are very sensitive to noise, which was applied for property testing lower bounds [2]. A Talagrand random DNF consists of $2^{\sqrt{n}}$ clauses of $\sqrt{n}$ indices chosen randomly with replacement. We modify this construction by picking the indices without replacement and generalize it by picking $2^{\sqrt{n}}/n^{C_2}$ clauses, where C_2 is a non-negative constant. We show that these functions are "thin", so they are appropriate for our lower bound technique.

"Thinness" allows us to conclude that D_1^B and D_2^B are hard to distinguish from each other. We then prove that they have drastically different biases. We do the latter by employing the probabilistic method and showing that in expectation our random function has a large enough bias. We handle influence and noise sensitivity analogously, specifically by showing that that as we pick fewer clauses, the expected influence and noise sensitivity decrease proportionally. We prove this by dividing the points, where one of these random functions equals to one,

into two regions: (i) the region where only one clause is true and (ii) a region where more than one clause is true. Roughly speaking, we show that the contribution from the points in (i) is sufficient to obtain a good lower bound on the influence and noise sensitivity.

1.4 Possibilities of improvement?

In [23] (which is the journal version of [24]), it was shown that using the chain decomposition of the hypercube, one can improve the run-time of the algorithm to $O\left(\frac{\sqrt{n}}{\epsilon^2 I[f]}\right)$ and also improve the required lower bound on $I[f]$ to be $I[f] \geq \exp(-c_1\epsilon^2 n + c_2 \log(n/\epsilon))$ for some constant c_1 and c_2 (it was $I[f] \geq 1/n^C$ for any constant C in [24]). Additionally, the algorithm itself was considerably simplified.

A hope is that techniques based on the chain decomposition could help improve the algorithm in Theorem 1. However, it is not clear how to generalize our approach to use these techniques, since the ascending-descending view is a natural way to express noise sensitivity in terms of random walks, and it is not obvious whether one can replace these walks with chains of the hypercube.

2 Preliminaries

2.1 Definitions

2.1.1 Fundamental definitions and lemmas pertaining to the hypercube

▶ **Definition 7.** *We refer to the poset over $\{0,1\}^n$ as the **n-dimensional hypercube**, viewing the domain as vertices of a graph, in which two vertices are connected by an edge if and only if the corresponding elements of $\{0,1\}^n$ differ in precisely one index. For $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ in $\{0,1\}^n$, we say that $x \preceq y$ if and only if for all i in $[n]$ it is the case that $x_i \leq y_i$.*

▶ **Definition 8.** *The **level of a vertex** x on the hypercube is the hamming weight of x, or in other words number of 1-s in x. We denote it by $L(x)$.*

We define the set of edges that are in the same "layer" of the hypercube as a given edge:

▶ **Definition 9.** *For an arbitrary edge e suppose $e = (v_1, v_2)$ and $v_2 \preceq v_1$. We denote Λ_e to be the set of all edges $e' = (v_1', v_2')$, so that $L(v_1) = L(v_1')$ and $L(v_2) = L(v_2')$.*

The size of Λ_e is $L(v_1)\binom{n}{L(v_1)}$. The concept of Λ_e will be useful because we will deal with paths that are symmetric with respect to change of coordinates, and these have an equal probability of hitting any edge in Λ_e.

As we view the hypercube as a graph, we will often refer to paths on it. By referring to a path P we will, depending on the context, refer to its set of vertices or edges.

▶ **Definition 10.** *We call a path **descending** if for every pair of consecutive vertices v_i and v_{i+1}, it is the case that $v_{i+1} \prec v_i$. Conversely, if the opposite holds and $v_i \prec v_{i+1}$, we call a path **ascending**. We consider an empty path to be vacuously both ascending and descending. We define the length of a path to be the number of edges in it, and denote it by $|P|$. We say we **take a descending random walk of length w starting at x**, if we pick a uniformly random descending path of length w starting at x.*

Descending random walks over the hyper-cube were used in an essential way in [24] and were central for the recent advances in monotonicity testing algorithms [5, 6, 14].

▶ **Lemma 11** (Hypercube Continuity Lemma). *Suppose n is a sufficiently large positive integer, C_1 is a constant and we are given l_1 and l_2 satisfying $\frac{n}{2} - \sqrt{C_1 n \log(n)} \leq l_1 \leq l_2 \leq \frac{n}{2} + \sqrt{C_1 n \log(n)}$. If we denote $C_2 \stackrel{\text{def}}{=} \frac{1}{10\sqrt{C_1}}$, then for any ξ satisfying $0 \leq \xi \leq 1$, if it is the case that $l_2 - l_1 \leq C_2 \xi \sqrt{\frac{n}{\log(n)}}$, then, for large enough n, it is the case that $1 - \xi \leq \frac{\binom{n}{l_1}}{\binom{n}{l_2}} \leq 1 + \xi$*

Proof. See the full version of the paper. ◀

2.1.2 Fundamental definitions pertaining to Boolean functions

▶ **Definition 12.** *Let δ be a parameter and let x be selected uniformly at random from $\{0,1\}^n$. Let $z \in \{0,1\}^n$ be defined as follows:*

$$z_i = \begin{cases} x_i & \text{with probability } 1-\delta \\ 1 - x_i & \text{with probability } \delta \end{cases}$$

*We denote this distribution of x and z by T_δ. Then we define the **noise sensitivity** of f as $NS_\delta[f] \stackrel{\text{def}}{=} \Pr_{(x,z)\in_R T_\delta}[f(x) \neq f(z)]$.*

▶ **Observation 13.** *For every pair of vertices a and b, the probability that for a pair x, z drawn from T_δ, it is the case that $(x,z) = (a,b)$, is equal to the probability that $(x,z) = (b,a)$.*

Therefore, $\Pr[f(x) = 0 \wedge f(z) = 1] = \Pr[f(x) = 1 \wedge f(z) = 0]$. Hence:

$$NS_\delta[f] = 2 \cdot \Pr[f(x) = 1 \wedge f(z) = 0]$$

2.1.3 Influence estimation

To estimate the influence, standard sampling would require $O\left(\frac{n}{I[f]\epsilon^2}\right)$ samples. However, from [24] we have:

▶ **Theorem 14.** *There is an algorithm that approximates $I[f]$ to within a multiplicative factor of $(1 \pm \epsilon)$ for a monotone $f : \{0,1\}^n \to \{0,1\}$. The algorithm requires that $I[f] \geq 1/n^{C'}$ for a constant C' that is given to the algorithm. It outputs a good approximation with probability at least 0.99 and in expectation requires $O\left(\frac{\sqrt{n}\log(n/\epsilon)}{I[f]\epsilon^3}\right)$ queries. Additionally, it runs in time polynomial in n.*

2.1.4 Bounds for the error parameter and the influence

The following observation allows us to assume that without loss of generality ϵ is not too small. A similar technique was also used in [24].

▶ **Observation 15.** *When $\epsilon < O(\sqrt{n}\delta \log^{1.5}(n))$ there is a simple algorithm that accomplishes the desired query complexity of $O\left(\frac{\sqrt{n}\delta \log^{1.5}(n)}{NS_\delta[f]\epsilon^3}\right)$. Namely, this can be done by the standard sampling algorithm that requires only $O\left(\frac{1}{NS_\delta[f]\epsilon^2}\right)$ samples. Thus, since we can handle the case when $\epsilon < O(\sqrt{n}\delta \log^{1.5}(n))$, we focus on the case when $\epsilon \geq H\sqrt{n}\delta \log^{1.5}(n) \geq H\frac{\log^{1.5} n}{\sqrt{n}}$, for any constant H.*

Additionally, throughout the paper whenever we need it, we will without loss of generality assume that ϵ is smaller than a sufficiently small positive constant.

We will also need a known lower bound on influence:

▶ **Observation 16.** *For any function* $f : \{0,1\}^n \to \{0,1\}$ *and* $\delta \leq 1/2$ *it is the case that* $NS_\delta[f] \leq \delta I[f]$*. Therefore it is the case that* $I[f] \geq \frac{1}{n^C}$*.*

A very similar statement is proved in [17] and for completeness we prove it in the full version of the paper.

3 An improved algorithm for small values of the noise parameter

In this section we give a more in-depth motivation for the analysis of our algorithm, together with the statements of the main lemmas. For all proofs, the reader is referred to the full version.

3.1 Descending-ascending framework

The descending-ascending process

It will be useful to view noise sensitivity in the context of the noise process D (see Algorithm 1 for the definition). By inspection, x and z are distributed identically in D as in T_δ. Therefore from Observation 13:

$$NS_\delta[f] = 2 \cdot \Pr_D[f(x) = 1 \wedge f(z) = 0]$$

▶ **Observation 17.** *Since the function is monotone, if* $f(x) = 1$ *and* $f(z) = 0$*, then it has to be that* $f(y) = 0$*.*

3.2 Review of algorithm

Roughly speaking, our algorithm will break the task of estimating $NS_\delta[f]$ into estimating the probabilities[5] $\Pr_D[f(x) = 1 \wedge f(y) = 0]$ and $\Pr_D[f(z) = 0 | f(x) = 1 \wedge f(y) = 0]$. To estimate the former, in Lemma 23 we will advantage of the fact that δ is small, so the path P_1 is typically short, and hence the same types of techniques can be applied as in the analysis of the influence estimation algorithm.

The situation here is different from that of influence estimation because (i) the length of the path is random (ii) this probability distribution of lengths depends on the starting vertex. However, we will prove there exists a value, call it p_1, so that most influential edges are hit with probability close to p_1. It depends on δ and $I[f]$, and we can estimate it quite efficiently by estimating $I[f]$.

In order to estimate $\Pr_D[f(z) = 0 | f(x) = 1 \wedge f(y) = 0]$ we will approximate the distribution D conditioned on $f(x) = 1 \wedge f(y) = 0$. To that end, we will first sample an influential edge e that P_1 goes through, and then among all the downwards paths going through e we will sample P_1 itself. The algorithm that samples an influential edge approximately uniformly is inspired by the algorithm that estimates influence.

3.2.1 Defining bad events and technical notation

In this section, we give the parameters that we use to determine the lengths of our walks, as well as the "middle" of the hypercube. Additionally, in this section we define some notation we use.

[5] In the precise analysis there will be some bad events to take care of. For the sake of simplicity, we do not talk about them right now.

Define the following values:

$$t_1 \stackrel{\text{def}}{=} \eta_1 \sqrt{n \log n} \qquad\qquad t_2 \stackrel{\text{def}}{=} n\delta(1 + 3\eta_2 \log n)$$

Here η_1 and η_2 are large enough constants. Taking $\eta_1 = \sqrt{C} + 4$ and $\eta_2 = C + 2$ is sufficient for our purposes (recall that we were promised that $NS_\delta[f] \geq 1/n^C$ for a constant C).

Informally, t_1 and t_2 have the following intuitive meaning. A typical vertex x of the hypercube has $L(x)$ between $n/2 - t_1$ and $n/2 + t_1$. A typical Phase 1 path from process D will have length at most t_2.

We define the "middle edges" M as the following set of edges:

$$M \stackrel{\text{def}}{=} \{e = (v_1, v_2) : \frac{n}{2} - t_1 \leq L(v_2) \leq L(v_1) \leq \frac{n}{2} + t_1\}$$

Denote by $\overline{M}$ the rest of the edges.

We define two bad events in context of D, such that when neither of these events happen, we can show that the output has certain properties. The first one happens roughly when P_1 (from x to y, as defined by Process D) is much longer than it should be in expectation, and the second one happens when P_1 crosses one of the edges that are too far from the middle of the hypercube, which could happen because P_1 is long or because of a starting point that is far from the middle. More specifically:

- E_1 happens when both of the following hold (i) P_1 crosses an edge $e \in E_I$ and (ii) denoting $e = (v_1, v_2)$, so that $v_2 \preceq v_1$, it is the case that $L(x) - L(v_1) \geq t_2$.
- E_2 happens when P_1 contains an edge in $E_I \cap \overline{M}$.

While defining E_1 we want two things from it. First of all, we want its probability to be upper-bounded easily. Secondly, we want it not to complicate the sampling of paths in Lemma 24. There exists a tension between these two requirements, and as a result the definition of E_1 is somewhat convoluted.

We will approximate the noise sensitivity as the product of the following two quantities:

$$p_A \stackrel{\text{def}}{=} \Pr_D[f(x) = 1 \wedge f(y) = 0 \wedge \overline{E_1} \wedge \overline{E_2}]$$

$$p_B \stackrel{\text{def}}{=} \Pr_D[f(z) = 0 | f(x) = 1 \wedge f(y) = 0 \wedge \overline{E_1} \wedge \overline{E_2}]$$

Ignoring the bad events, P_A is the probability that P_1 hits an influential edge, and P_B is the probability that given that P_1 hits an influential edge P_2 does not hit an influential edge. From Observation (17), if and only if these two things happen, it is the case that $f(x) = 1$ and $f(z) = 0$. From this fact and the laws of conditional probabilities we have:

$$\Pr_D[f(x) = 1 \wedge f(z) = 0 \wedge \overline{E_1} \wedge \overline{E_2}] = \Pr_D[f(x) = 1 \wedge f(y) = 0 \wedge f(z) = 0 \wedge \overline{E_1} \wedge \overline{E_2}] = p_A p_B \quad (2)$$

We can consider for every individual edge e in $M \cap E_I$ the probabilities:

$$p_e \stackrel{\text{def}}{=} \Pr_D[e \in P_1 \wedge \overline{E_1} \wedge \overline{E_2}]$$

$$q_e \stackrel{\text{def}}{=} \Pr_D[f(x) = 1 \wedge f(z) = 0 | e \in P_1 \wedge \overline{E_1} \wedge \overline{E_2}] = \Pr_D[f(z) = 0 | e \in P_1 \wedge \overline{E_1} \wedge \overline{E_2}]$$

The last equality is true because $e \in P_1$ already implies $f(x) = 1$. Informally and ignoring the bad events again, p_e is the probability that $f(x) = 1$ and $f(y) = 0$ **because** P_1 hits e and not some other influential edge. Similarly, q_e is the probability $f(x) = 1$ and $f(z) = 0$ given that P_1 hits specifically e.

Since f is monotone, P_1 can hit at most one influential edge. Therefore, the events of P_1 hitting different influential edges are disjoint. Using this, Equation (2) and the laws of conditional probabilities we can write:

$$p_A = \sum_{e \in E_I \cap M} p_e \tag{3}$$

Furthermore, the events that P_1 hits a given influential edge and then P_2 does not hit any are also disjoint for different influential edges. Therefore, analogous to the previous equation we can write:

$$p_A p_B = \Pr_D[(f(x) = 1) \wedge (f(z) = 0) \wedge \overline{E_1} \wedge \overline{E_2}] = \sum_{e \in E_I \cap M} p_e q_e \tag{4}$$

3.2.2 Bad events can be "ignored"

In the following, we will need to consider probability distributions in which bad events do not happen. For the most part, conditioning on the fact that bad events do not happen changes little in the calculations. In this subsection, we present two relatively simple lemmas that allow us to formalize these claims.

The following observation suggests that almost all influential edges are in M.

▶ **Observation 18.** *It is the case that:*

$$\left(1 - \frac{\epsilon}{310}\right) |E_I| \leq |M \cap E_I| \leq |E_I|$$

Proof. This is the case, because:

$$|\overline{M} \cap E_I| \leq |\overline{M}| \leq 2^n n \cdot 2\exp(-2\eta_1^2 \log(n)) = 2^{n-1} \cdot 4/n^{2\eta_1^2 - 1} \leq 2^{n-1} I[f]/n = |E_I|/n \leq \frac{\epsilon}{310}|E_I| \tag{5}$$

The second inequality is the Hoeffding bound, then we used Observations 16 and 15. ◀

We now assert that ignoring these bad events does not distort our estimate for $NS_\delta[f]$.

▶ **Lemma 19.** *It is the case that:*

$$p_A p_B \leq \frac{1}{2} NS_\delta[f] \leq \left(1 + \frac{\epsilon}{5}\right) p_A p_B$$

Proof. See the full version of the paper. ◀

3.3 Main lemmas

Having rigorously defined our technical language, now we can state our main algorithmic lemmas, together with their motivation and our approach to proving them. We refer the reader to the full version of the paper for the proofs of these lemmas, the proof of the Theorem 1 using these lemmas, as well as the derivation of the lower bounds in Theorems 5, 6 and 2.

The first two lemmas allow the estimation of the probability that a certain descending random walk hits an influential edge. As we mentioned in the introduction, except for the binary search step, the algorithm in Lemma 20 is similar to the algorithm in [24]. In principle, we could have carried out much of the analysis of the algorithm in Lemma 20 by referencing

an equation in [24]. However, for subsequent lemmas, including Lemma 23, we build on the application of the Hypercube Continuity Lemma to the analysis of random walks on the hypercube. In the full version of the paper, we give a full analysis of the algorithm in Lemma 20, in order to demonstrate how the Hypercube Continuity Lemma (Lemma 11) can be used to analyze random walks on the hypercube, before handling the more complicated subsequent lemmas, including Lemma 23.

▶ **Lemma 20.** *There exists an algorithm $\mathcal{A}$ (see Algorithm 2) that samples edges from $M \cap E_I$ so that for every two edges e_1 and e_2 in $M \cap E_I$:*

$$\left(1 - \frac{\epsilon}{70}\right) \Pr_{e \in_R \mathcal{A}}[e = e_2] \leq \Pr_{e \in_R \mathcal{A}}[e = e_1] \leq \left(1 + \frac{\epsilon}{70}\right) \Pr_{e \in_R \mathcal{A}}[e = e_2]$$

The probability that the algorithm succeeds is at least $\frac{1}{O(\sqrt{n}\log^{1.5} n/I[f]\epsilon)}$. If it succeeds, the algorithm makes $O(\log n)$ queries, and if it fails, it makes only $O(1)$ queries. In either case, it runs in time polynomial in n.

▶ Remark 21. Through the standard repetition technique, the probability of error can be decreased to an arbitrarily small constant, at the cost of $O(\frac{\sqrt{n}\log^{1.5} n}{I[f]\epsilon})$ queries. Then, the run-time still stays polynomial in n, since $I[f] \geq 1/n^C$.

▶ Remark 22. The distribution $\mathcal{A}$ outputs is point-wise close to the uniform distribution over $M \cap E_I$. We will also obtain such approximations to other distributions in further lemmas. Note that this requirement is stronger than closeness in L_1 norm.

The following lemma, roughly speaking, shows that just as in previous lemma, the probability that P_1 in D hits an influential edge e does not depend on where exactly e is, as long as it is in $M \cap E_I$. The techniques we use are similar to the ones in the previous lemma and it follows the same outline. However here we encounter additional difficulties for two reasons: first of all, the length of P_1 is not fixed, but it is drawn from a probability distribution. Secondly, this probability distribution depends on the starting point of P_1.

▶ **Lemma 23.** *For any edge $e \in M \cap E_I$ it is the case that:*

$$\left(1 - \frac{\epsilon}{310}\right) \frac{\delta}{2^n} \leq p_e \leq \left(1 + \frac{\epsilon}{310}\right) \frac{\delta}{2^n}$$

While we will use Lemma 23 in order to estimate p_A, the next two lemmas are for estimating p_B. To that end, we will need to sample from a distribution of descending and ascending paths going through a given edge. Informally, the requirement on the distribution is that it should be close to the conditional distribution of such paths P_1 that would arise from process D, conditioned on going through e and satisfying $\bar{E}_1$ and $\bar{E}_2$.

A first approach to sampling such P_1 would be to take random walks in opposite directions from the endpoints of the edge e and then concatenate them together. This is in fact what we do. However, difficulty comes from determining the appropriate lengths of the walks for the following reason. If P_1 is longer, it is more likely to hit the influential edge e. This biases the distribution of the descending paths hitting e towards the longer descending paths. In order to accommodate for this fact we used the following two-step approach:

1. Sample only the levels of the starting and ending points of the path P_1. This is equivalent to sampling the length of the segment of P_1 before the edge e and after it. This requires careful use of rejection sampling together with the techniques we used to prove Lemmas 20 and 23. Roughly speaking, we use the fact that P_1 is distributed symmetrically with respect to the change of indices in order to reduce a question about the edge e to a question about the layer Λ_e. Then, we use the Lemma 11 to answer questions about random walks hitting a given layer. This is handled in Lemma 24.

2. Sample a path P_1 that has the given starting and ending levels and passes through an influential edge e. This part is relatively straightforward. We prove that all the paths satisfying these criteria are equally likely. We sample one of them randomly by performing two random walks in opposite directions starting at the endpoints of e. This all is handled in Lemma 25.

▶ **Lemma 24.** *There is an algorithm $\mathcal{W}$ (see Algorithm 3) that takes as input an edge $e = (v_1, v_2)$ in $M \cap E_I$, so that $v_2 \preceq v_1$, and samples two non-negative numbers w_1 and w_2, so that for any two non-negative w_1' and w_2':*

$$\left(1 - \frac{\epsilon}{70}\right) \Pr_{\mathcal{W}(e)}[(w_1 = w_1') \wedge (w_2 = w_2')]$$
$$\leq \Pr_D[(L(x) - L(v_1) = w_1') \wedge (L(v_2) - L(y) = w_2') | (e \in P_1) \wedge \overline{E_1} \wedge \overline{E_2}]$$
$$\leq \left(1 + \frac{\epsilon}{70}\right) \Pr_{\mathcal{W}(e)}[(w_1 = w_1') \wedge (w_2 = w_2')] \quad (6)$$

The algorithm requires no queries to f and runs in time polynomial in n.

▶ **Lemma 25.** *There exists an algorithm $\mathcal{B}$ (see Algorithm 4) with the following properties. It takes as input an edge $e = (v_1, v_2)$ in $M \cap E_I$, so that $v_2 \preceq v_1$ and outputs paths P_1 and P_2 together with hypercube vertices x, y and z. It is the case that x is the starting vertex of P_1, y is both the starting vertex of P_2 and the last vertex of P_1, and z is the last vertex of P_2. Additionally, P_1 is descending and P_2 is ascending. Furthermore, for any pair of paths P_1' and P_2' we have:*

$$\left| \Pr_{\mathcal{B}(e)}[(P_1 = P_1') \wedge (P_2 = P_2')] - \Pr_D[(P_1 = P_1') \wedge (P_2 = P_2') | (e \in P_1) \wedge \overline{E_1} \wedge \overline{E_2}] \right|$$
$$\leq \frac{\epsilon}{70} \Pr_{\mathcal{B}(e)}[(P_1 = P_1') \wedge (P_2 = P_2')] \quad (7)$$

It requires no queries to the function and takes computation time polynomial in n to draw one sample.

In the full version of this paper, we analyze the query complexity and the run-time of the Algorithm 5, thus proving Theorem 1. This is shown to be a relatively straightforward application of the four main technical lemmas we presented and discussed in this section.

References

1. Maria-Florina Balcan, Eric Blais, Avrim Blum, and Liu Yang. Active property testing. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 21–30. IEEE, 2012.
2. Aleksandrs Belovs and Eric Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 1021–1032. ACM, 2016.
3. Itai Benjamini, Gil Kalai, and Oded Schramm. Noise sensitivity of Boolean functions and applications to percolation. *Publications Mathématiques de l'Institut des Hautes Études Scientifiques*, 90(1):5–43, 1999.
4. Eric Blais, Ryan O'Donnell, and Karl Wimmer. Polynomial regression under arbitrary product distributions. *Machine learning*, 80(2-3):273–294, 2010.
5. Deeparnab Chakrabarty and Comandur Seshadhri. An o(n) Monotonicity Tester for Boolean Functions over the Hypercube. *SIAM Journal on Computing*, 45(2):461–472, 2016.

6 Xi Chen, Rocco A Servedio, and Li-Yang Tan. New algorithms and lower bounds for monotonicity testing. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 286–295. IEEE, 2014.

7 Mahdi Cheraghchi, Adam Klivans, Pravesh Kothari, and Homin K Lee. Submodular functions are noise stable. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1586–1592. Society for Industrial and Applied Mathematics, 2012.

8 Ilias Diakonikolas, Prasad Raghavendra, Rocco A. Servedio, and Li-Yang Tan. Average Sensitivity and Noise Sensitivity of Polynomial Threshold Functions. *SIAM J. Comput.*, 43(1):231–253, 2014. doi:10.1137/110855223.

9 Adam Tauman Kalai, Adam R Klivans, Yishay Mansour, and Rocco A Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008.

10 Daniel M. Kane. The Gaussian Surface Area and Noise Sensitivity of Degree-d Polynomial Threshold Functions. *Computational Complexity*, 20(2):389–412, 2011. doi:10.1007/s00037-011-0012-6.

11 Daniel M. Kane. The average sensitivity of an intersection of half spaces. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 437–440, 2014. doi:10.1145/2591796.2591798.

12 Nathan Keller and Guy Kindler. Quantitative relation between noise sensitivity and influences. *Combinatorica*, 33(1):45–71, 2013.

13 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable CSPs? *SIAM Journal on Computing*, 37(1):319–357, 2007.

14 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and boolean isoperimetric type theorems. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 52–58. IEEE, 2015.

15 Adam R Klivans, Ryan O'Donnell, and Rocco A Servedio. Learning intersections and thresholds of halfspaces. *Journal of Computer and System Sciences*, 68(4):808–840, 2004.

16 Rajsekar Manokaran, Joseph Seffi Naor, Prasad Raghavendra, and Roy Schwartz. SDP gaps and UGC hardness for multiway cut, 0-extension, and metric labeling. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 11–20. ACM, 2008.

17 Elchanan Mossel and Ryan O'Donnell. On the noise sensitivity of monotone functions. *Random Structures & Algorithms*, 23(3):333–350, 2003.

18 Elchanan Mossel and Ryan O'Donnell. Coin flipping from a cosmic source: On error correction of truly random bits. *Random Structures & Algorithms*, 26(4):418–436, 2005.

19 Ryan O'Donnell. Hardness amplification within NP. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 751–760. ACM, 2002.

20 Ryan O'Donnell. *Computational applications of noise sensitivity*. PhD thesis, Massachusetts Institute of Technology, 2003.

21 Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

22 Yuval Peres. Noise stability of weighted majority. *arXiv preprint*, 2004. arXiv:math/0412377.

23 Dana Ron, Ronitt Rubinfeld, Muli Safra, Alex Samorodnitsky, and Omri Weinstein. Approximating the Influence of Monotone Boolean Functions in $O(\sqrt{n})$ Query Complexity. *TOCT*, 4(4):11:1–11:12, 2012. doi:10.1145/2382559.2382562.

24 Dana Ron, Ronitt Rubinfeld, Muli Safra, and Omri Weinstein. Approximating the Influence of Monotone Boolean Functions in $O(\sqrt{n})$ Query Complexity. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 664–675. Springer, 2011.

25 Michel Talagrand. How Much Are Increasing Sets Positively Correlated? *Combinatorica*, 16(2):243–258, 1996.

Connectivity of Random Annulus Graphs and the Geometric Block Model

Sainyam Galhotra
University of Massachusetts Amherst, USA
sainyam@cs.umass.edu

Arya Mazumdar
University of Massachusetts Amherst, USA
arya@cs.umass.edu

Soumyabrata Pal
University of Massachusetts Amherst, USA
spal@cs.umass.edu

Barna Saha
University of California, Berkeley, USA
barnas@berkeley.edu

Abstract

Random geometric graph (Gilbert, 1961) is a basic model of random graphs for spatial networks proposed shortly after the introduction of the Erdős-Rényi random graphs. The *geometric block model* (GBM) is a probabilistic model for community detection defined over random geometric graphs (RGG) similar in spirit to the popular *stochastic block model* which is defined over Erdős-Rényi random graphs. The GBM naturally inherits many desirable properties of RGGs such as transitivity ("friends having common friends') and has been shown to model many real-world networks better than the stochastic block model. Analyzing the properties of a GBM requires new tools and perspectives to handle correlation in edge formation. In this paper, we study the necessary and sufficient conditions for community recovery over GBM in the connectivity regime. We provide efficient algorithms that recover the communities exactly with high probability and match the lower bound within a small constant factor. This requires us to prove new connectivity results for *vertex-random graphs* or *random annulus graphs* which are natural generalizations of random geometric graphs.

A vertex-random graph is a model of random graphs where the randomness lies in the vertices as opposed to an Erdős-Rényi random graph where the randomness lies in the edges. A vertex-random graph $G(n, [r_1, r_2]), 0 \leq r_1 < r_2 \leq 1$ with n vertices is defined by assigning a real number in $[0, 1]$ randomly and uniformly to each vertices and adding an edge between two vertices if the "distance" between the corresponding two random numbers is between r_1 and r_2. For the special case of $r_1 = 0$, this corresponds to random geometric graph in one dimension. We can extend this model naturally to higher dimensions; these higher dimensional counterparts are referred to as *random annulus graphs*. Random annulus graphs appear naturally whenever the well-known Goldilocks principle ("not too close, not too far') holds in a network. In this paper, we study the connectivity properties of such graphs, providing both necessary and sufficient conditions. We show a surprising *long edge phenomena* for vertex-random graphs: the minimum gap for connectivity between r_1 and r_2 is significantly less when $r_1 > 0$ vs when $r_1 = 0$ (RGG). We then extend the connectivity results to high dimensions. These results play a crucial role in analyzing the GBM.

2012 ACM Subject Classification Mathematics of computing → Random graphs

Keywords and phrases random graphs, geometric graphs, community detection, block model

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.53

Category RANDOM

Related Version A full version of the paper is available at [16], `https://arxiv.org/abs/1804.05013`.

Funding S. Galhotra and B. Saha are supported in part by NSF 1652303, a Google award and a Sloan fellowship. A. Mazumdar and S. Pal are supported in part by NSF 1642658 and 1642550.



© Sainyam Galhotra, Arya Mazumdar, Soumyabrata Pal, and Barna Saha;
licensed under Creative Commons License CC-BY



1 Introduction

Models of random graphs are ubiquitous with Erdős-Rényi graphs [12, 17] at the forefront. Studies of the properties of random graphs have led to many fundamental theoretical observations as well as many engineering applications. In an Erdős-Rényi graph $G(n,p), n \in \mathbb{Z}_+, p \in [0,1]$, the randomness lies in how the edges are chosen: each possible pair of vertices forms an edge independently with probability p. It is also possible to consider models of graphs where randomness lies in the vertices.

Keeping up with the simplicity of the Erdős-Rényi model, one can define a vertex-random graph (VRG) in the following way. Given two reals $0 \leq r_1 \leq r_2 \leq 1/2$, the vertex-random graph $\text{VRG}(n,[r_1,r_2])$ is a random graph with n vertices. Each vertex u is assigned a random point X_u selected uniformly from the circumference of a circle of perimeter 1. Two vertices u and v are connected by an edge, if and only if the distance of the corresponding points on the circle (the geodesic distance) is between r_1 and r_2. This definition is by no means new. For the case of $r_1 = 0$, this is the random geometric graphs (RGG) in one dimension. Random Geometric graphs were defined first by [18] and constitute the first and simplest model of spatial networks. Since then, they have found wide-spread applications in modeling wireless (ad-hoc) communication networks [9, 19], information propagation in social networks [13, 31] etc., and have been studied extensively [4, 5, 6]. The definition of VRG has been previously mentioned in [9]. The interval $[r_1,r_2]$ is called the connectivity interval in VRGs.

Vertex random graphs inherit many desirable properties of RGGs such as vertices with high modularity and the degree associativity property (high degree nodes tend to connect), which in turn led to the popularity of RGGs [13, 31]. In addition, VRGs naturally arise whenever the Goldilocks principle (not too close, not too far) is applicable in networks. For example, in a co-purchase network, a person who bought a bike may buy similar products like a helmet along with it, but not another bike [15]. Understanding connectivity properties of VRGs can shed light in co-purchasing behavior and product recommendation. Interestingly, the connectivity properties of VRGs turn out to be crucial to develop and analyze community detection algorithms for the *geometric block model* [15].

Connectivity of Vertex Random Graph (VRG). Threshold properties of Erdős-Rényi graphs have been at the center of much interest, and in particular it is known that many graph properties exhibit sharp phase transition phenomena [14]. Random geometric graphs also exhibit similar threshold properties [26]. Our first contribution in this work is to identify such connectivity threshold for VRGs. Consider a $\text{VRG}(n,[0,r])$ defined above with $r = \frac{a \ln n}{n}$. It is known that $\text{VRG}(n,[0,r])$ is connected with high probability if and only if $a > 1$ (I.e., $\text{VRG}(n,[0,\frac{(1+\epsilon)\ln n}{n}])$ is connected for any $\epsilon > 0$. We will ignore this ϵ and just mention connectivity threshold as $\frac{\ln n}{n}$). Now let us consider the graph $\text{VRG}(n,[\frac{b\ln n}{n},\frac{\ln n}{n}])$, $b > 0$. Clearly this graph has less edges than $\text{VRG}(n,[0,\frac{\ln n}{n}])$. **Is this graph still connected?** Surprisingly, we show that the modified graph remains connected as long as $b \leq 0.5$. Therefore, $\text{VRG}(n,[\frac{0.5\ln n}{n},\frac{\ln n}{n}])$ is connected, but $\text{VRG}(n,[0,\frac{(1-\epsilon)\ln n}{n}])$ is not $\forall \epsilon > 0$.

Can we explain this striking shift in connectivity interval, when one goes from $b = 0$ to $b > 0$? Note that the $\text{VRG}(n,[\frac{0.50\ln n}{n},\frac{\ln n}{n}])$ is obtained from the $\text{VRG}(n,[0,\frac{\ln n}{n}])$ by deleting all "short-distance" edges. It turns out the "long-distance" edges are sufficient to maintain connectivity, because they can connect points over multiple hops in the graph. Another possible explanation is that connectivity threshold for VRG is not dictated by isolated nodes as is the case in Erdős-Rényi graphs. Thus, after the connectivity threshold has been achieved, removing short edges still retains connectivity.

The Geometric Block Model. We are motivated to study the threshold phenomena of vertex-random graphs, because it appears naturally in the analysis of the geometric block model (GBM) [15]. The geometric block model is a probabilistic generative model of communities and is a spatial analogue to the popular stochastic block model (SBM) [22, 10, 8, 2, 1, 20, 7, 24]. The SBM generalizes the Erdős-Rényi graphs in the following way. Consider a graph $G(V, E)$, where $V = V_1 \sqcup V_2 \sqcup \cdots \sqcup V_k$ is a disjoint union of k clusters denoted by $V_1, \ldots, V_k$. The edges of the graph are drawn randomly: there is an edge between $u \in V_i$ and $v \in V_j$ with probability $q_{i,j}, 1 \leq i, j \leq k$. Given the adjacency matrix of such a graph, the task is to find the partition $V_1 \sqcup V_2 \sqcup \cdots \sqcup V_k$ of V.

This model has been incredibly popular both in theoretical and practical domains of community detection. Recent theoretical works focus on characterizing sharp threshold of recovering the partition in the SBM. For example, when there are only two communities of exactly equal sizes, and the inter-cluster edge probability is $\frac{b \ln n}{n}$ and intra-cluster edge probability is $\frac{a \ln n}{n}$, it is known that exact recovery is possible if and only if $\sqrt{a} - \sqrt{b} > \sqrt{2}$ [1, 24]. The regime of the probabilities being $\Theta\left(\frac{\ln n}{n}\right)$ has been put forward as one of most interesting ones, because in an Erdős-Rényi random graph, this is the threshold for graph connectivity [4]. Note that the results are not only of theoretical interest, many real-world networks exhibit a "sparsely connected" community feature [23], and any efficient recovery algorithm for sparse SBM has many potential applications.

While the SBM is a popular model (because of its apparent simplicity), there are many aspects of real social networks, such as "transitivity rule" ("friends having common friends') and other community structures that are not accounted for in SBM. Defining a block model over a random geometric graph, the geometric block model (GBM), circumvents this since GBMs naturally inherit the transitivity property of random geometric graphs. In a previous work [15], we showed GBMs model community structures better than an SBM in many real world networks (e.g. DBLP collaboration network, Amazon co-purchase network etc.). The GBM depends on the basic definition of the random geometric graph in the same way the SBM depends on Erdős-Rényi graphs. The two-cluster GBM with vertex set $V = V_1 \sqcup V_2$, $V_1 = V_2$ is a random graph defined in the following way. Suppose, $0 \leq r_d < r_s \leq 1/2$ be two real numbers. For each vertex $u \in V$ randomly and independently choose a point X_u from the circumference of a circle of unit perimeter. There will be an edge between u and v if and only if,

$$d_L(X_u, X_v) \leq r_s \text{ when } u, v \in V_1 \text{ or } u, v \in V_2$$
$$d_L(X_u, X_v) \leq r_d \text{ when } u \in V_1, v \in V_2 \text{ or } u \in V_2, v \in V_1,$$

where d_L denotes the geodesic distance. Let us denote this random graph as GBM(r_s, r_d). Given this graph GBM(r_s, r_d), the main problem of community detection is to recover the partition (i.e., V_1 and V_2). The GBM provides a systematic way to introduce correlation during edge formation, an important aspect in real networks that often renders a problem theoretically intractable. The tool set needed to recover communities under a GBM thus differs significantly than what has been used to analyze the SBM.

Motivated by the SBM literature, we here also look at the GBM in the connectivity regime, i.e., when $r_s = \frac{a \ln n}{n}, r_d = \frac{b \ln n}{n}$. Our first contribution in this part is to provide a lower bound that shows that it is impossible to recover the parts from GBM$(\frac{a \ln n}{n}, \frac{b \ln n}{n})$ when $a - b < 1/2$. No lower bound for recovery was known before. We also derive a relation between a and b that defines a sufficient condition of recovery in GBM$(\frac{a \ln n}{n}, \frac{b \ln n}{n})$, closely matching the lower bound. The analysis crucially exploits the connectivity properties of vertex-random graphs.

It is possible to generalize the GBM to include different distributions, different metric spaces and multiple parts. It is also possible to construct other type of spatial block models such as the one very recently being put forward in [28] which rely on the random dot product graphs [30]. In [28], edges are drawn between vertices randomly and independently as a function of the distance between the corresponding vertex random variables. In contrast, in GBM edges are drawn deterministically given the vertex random variables, and edges are dependent unconditionally. Moreover [28] only considers the recovery scenario where in addition to the graph, values of the vertex random variables are provided. Note that in GBM, we only observe the graph. In particular, it will be later clear that if we are given the corresponding random variables (locations) to the vertices in addition to the graph, then recovery of the partitions in $\text{GBM}(\frac{a\ln n}{n}, \frac{b\ln n}{n})$ is possible if and only if $a-b>0.5$ and $a>1$, that is we can identify the recovery threshold exactly.

VRG in Higher Dimension: The Random Annulus Graphs. It is natural to ask similar question of connectivity for VRGs in higher dimension. In a VRG at dimension t, we may assign t-dimensional random vectors to each of the vertices, and use a standard metric such as the Euclidean distance to decide whether there should be an edge between two vertices. Formally, let us define the t-dimensional sphere as $S^t \equiv \{x \in \mathbb{R}^{t+1} \mid ||x||_2 = 1\}$. Given two reals $0 \le r_1 \le r_2 \le 2$, the random annulus graph $\text{RAG}_t(n, [r_1, r_2])$ is a random graph with n vertices. Each vertex u is assigned a random vector X_u selected randomly and uniformly from S^t. Two vertices u and v are connected by an edge, if and only if $r_1 \le d(u,v) \equiv \|X_u - X_v\|_2 \le r_2$. Note that for $t=1$ an $\text{RAG}_1(n,[r_1,r_2])$ is nothing but a VRG as defined above, where we need to convert the Euclidean distance to the geodesic distance and scale the probabilities by a factor of 2π. The $\text{RAG}_t(n,[0,r])$ gives the standard definition of random geometric graphs in t dimensions (for example, see [6] or [26]).

We refer to high-dimensional VRGs as the random annulus graph (RAG) since here two vertices are connected iff one is within an "annulus" centered at the other. For the random annulus graphs, we extend our connectivity results of $t=1$ to general t. In particular, we show that there exists an isolated vertex in the $\text{RAG}_t(n, [b(\frac{\ln n}{n})^{\frac{1}{t}}, a(\frac{\ln n}{n})^{\frac{1}{t}}])$ with high probability if and only if

$$a^t - b^t < \frac{\sqrt{\pi}(t+1)\Gamma(\frac{t+2}{2})}{\Gamma(\frac{t+3}{2})} \equiv \psi(t),$$

where $\Gamma(\cdot)$ is the gamma function. Computing the connectivity threshold of RAG exactly is highly challenging, and we have to use several approximations of high dimensional geometry. Our arguments crucially rely on VC dimensions of sets of geometric objects such as intersections of high dimensional annuluses and hyperplanes. Overall we find that the $\text{RAG}_t(n, [b(\frac{\ln n}{n})^{\frac{1}{t}}, a(\frac{\ln n}{n})^{\frac{1}{t}}])$ is connected with high probability if

$$(a/2)^t - b^t \ge 8(t+1)\psi(t) \text{ and } a > 2b.$$

Using the connectivity result for RAG_t, the results for the geometric block model can be extended to high dimensions. The latent feature space of nodes in most networks are high-dimensional. For example, road networks are two-dimensional whereas the number of features used in a social network may have much higher dimensions. In a "high-dimensional" GBM: for any $t>1$, instead of assigning a random variable from $[0,1]$ we assign a random vector $X_u \in S^t$ to each vertex u; and two vertices in the same part is connected if and only if their Euclidean distance is less than r_s, whereas two vertices from different parts are connected if and only if their distance is less than r_d. We show the algorithm developed for one dimension extends to higher dimensions with nearly tight lower and upper bounds.

In this paper, we consistently refer to the $t = 1$ case for RAG as the vertex-random graph.

The paper is organized as follows. In Section 2, we provide the formal definitions and the main results of the paper. In Section 3, the sharp connectivity phase transition results for vertex-random graphs are proven. In Section 4, the connectivity results are proven for high dimensional random annulus graphs (details in full version [16]). Finally, in Section 5, a lower bound for the geometric block model as well as the main recovery algorithm are presented (details for the high-dimensional case in full version [16]).

2 Main Results

We formally define the random graph models, and state our results here.

▶ **Definition 1** (Vertex-Random Graph). *A vertex-random graph* $\mathrm{VRG}(n, [r_1, r_2])$ *on* n *vertices has parameters* n*, and a pair of real numbers* $r_1, r_2 \in [0, 1/2], r_1 \leq r_2$*. It is defined by assigning a number* $X_i \in \mathbb{R}$ *to vertex* $i, 1 \leq i \leq n$*, where* X_i*'s are independent and identical random variables uniformly distributed in* $[0, 1]$*. There will be an edge between vertices* i *and* $j, i \neq j$*, if and only if* $r_1 \leq d_L(X_i, X_j) \leq r_2$ *where* $d_L(X_i, X_j) \equiv \min\{|X_i - X_j|, 1 - |X_i - X_j|\}$.

We choose $d_L(X_i, X_j) = \min\{|X_i - X_j|, 1 - |X_i - X_j|\}$ to ignore the boundary effect, although the results extend identically to the scenario when $d_L(X_i, X_j) = |X_i - X_j|$. One can also interpret $X_i, 1 \leq i \leq n$, to be uniformly distributed on the perimeter of a circle with radius $\frac{1}{2\pi}$ and the distance $d_L(\cdot, \cdot)$ to be the geodesic distance. As a shorthand, for any two vertices u, v, let $d(u, v)$ denote $d_L(X_u, X_v)$ where X_u, X_v are the random variables corresponding to the vertices. We also use $d(u, v)$ to denote the distance between a vertex u (or the embedding of that vertex in $[0, 1]$) and a point $v \in [0, 1]$ naturally. Our main result regarding VRGs is summarized in the following theorem.

▶ **Theorem 2** (Connectivity threshold of vertex-random graphs). *The* $\mathrm{VRG}(n, [\frac{b \ln n}{n}, \frac{a \ln n}{n}])$ *is connected with probability* $1 - o(1)$ *if* $a > 1$ *and* $a - b > 0.5$*. On the other hand, the* $\mathrm{VRG}(n, [\frac{b \ln n}{n}, \frac{a \ln n}{n}])$ *is not connected with probability* $1 - o(1)$ *if* $a < 1$ *or* $a - b < 0.5$.

Only for the special case of $b = 0$, the connectivity result was known before [25, 26]. See also [27]. Generalization to $b > 0$ is both nontrivial and counter-intuitive (the minimum connectivity gap is no longer $a - b \geq 1$). Indeed, our analysis also leads to an alternate simple proof of connectivity for one-dimensional RGGs.

▶ **Definition 3** (The Random Annulus Graph). *Let us define the* t*-dimensional unit sphere as* $S^t \equiv \{x \in \mathbb{R}^{t+1} \mid ||x||_2 = 1\}$*. A random annulus graph* $\mathrm{RAG}_t(n, [r_1, r_2])$ *on* n *vertices has parameters* $n, t \in \mathbb{Z}_+$*, and a pair of real numbers* $r_1, r_2 \in [0, 2], r_1 \leq r_2$*. It is defined by assigning a number* $X_i \in S^t$ *to vertex* $i, 1 \leq i \leq n$*, where* X_i*'s are independent and identical random vectors uniformly distributed in* S^t*. There will be an edge between vertices* i *and* $j, i \neq j$*, if and only if* $r_1 \leq \|X_i - X_j\|_2 \leq r_2$ *where* $\|\cdot\|_2$ *denote the* ℓ_2 *norm.*

When from the context it is clear that we are in high dimensions, we use $d(u, v)$ to denote $\|X_u - X_v\|_2$ or just the ℓ_2 distance between the arguments[1]. The following result summarizes the condition for the existence of isolated vertices in RAGs.

[1] If we substitute $t = 1$, then $\mathrm{RAG}_1(n, [r_1, r_2])$ is a random graph where each vertex is associated with a random variable uniformly distributed in the unit circle. The distance between two vertices is the length of the chord connecting the random variables corresponding to the two vertices. If the length of the chord is $r \leq 2$, then the length of the corresponding (smaller) chord length of the corresponding arc between the vertices along the circumference of the circle is $2 \sin^{-1} \frac{r}{2}$. If we normalize the circumference of the circle by 2π, we obtain a random graph model that is equivalent to our definition of the vertex-random graphs. Since handling geodesic distances is more cumbersome in the higher dimensions, we resorted to Euclidean distance.

▶ **Theorem 4** (Zero-One law for Isolated Vertex in RAG). *For a random annulus graph* $\mathrm{RAG}_t(n, [r_1, r_2])$ *where* $r_2 = a\left(\frac{\ln n}{n}\right)^{\frac{1}{t}}$ *and* $r_1 = b\left(\frac{\ln n}{n}\right)^{\frac{1}{t}}$*, there exists isolated nodes with probability* $1 - o(1)$ *if*

$$a^t - b^t < \frac{\sqrt{\pi}(t+1)\Gamma(\frac{t+2}{2})}{\Gamma(\frac{t+3}{2})} \equiv \psi(t),$$

where $\Gamma(x) = \int_0^\infty y^{x-1}e^{-y}dy$ *is the gamma function, and there does not exist an isolated vertex with probability* $1 - o(1)$ *if* $a^t - b^t > \psi(t)$*.*

As a corollary of the above, we observe an $\mathrm{RAG}_t(n, [b\left(\frac{\ln n}{n}\right)^{\frac{1}{t}}, a\left(\frac{\ln n}{n}\right)^{\frac{1}{t}}])$ is not connected with probability $1 - o(1)$ if $a^t - b^t < \psi(t)$. Our main result provides a sufficient condition for the connectivity.

▶ **Theorem 5.** *A* t *dimensional random annulus graph* $\mathrm{RAG}_t(n, [b\left(\frac{\ln n}{n}\right)^{\frac{1}{t}}, a\left(\frac{\ln n}{n}\right)^{\frac{1}{t}}])$ *is connected with probability* $1 - o(1)$ *if* $(a/2)^t - b^t \geq 8(t+1)\psi(t)$ *and* $a > 2b$*.*

These connectivity results find immediate application in analyzing the geometric block model (GBM), a generative model for networks with underlying community structure.

▶ **Definition 6** (Geometric Block Model). *Given* $V = V_1 \sqcup V_2, |V_1| = |V_2| = \frac{n}{2}$*, choose a random variable* X_u *uniformly distributed in* $[0, 1]$ *for all* $u \in V$*. The geometric block model* $\mathrm{GBM}(r_s, r_d)$ *with parameters* $1/2 \geq r_s > r_d$ *is a random graph where an edge exists between vertices* u *and* v *if and only if,*

$$d_L(X_u, X_v) \leq r_s \text{ when } u, v \in V_1 \text{ or } u, v \in V_2$$
$$d_L(X_u, X_v) \leq r_d \text{ when } u \in V_1, v \in V_2 \text{ or } u \in V_2, v \in V_1.$$

As a consequence of the connectivity lower bound on VRG, we are able to show community recovery lower bound, that is we show the recovery of the partition is not possible with high probability in $\mathrm{GBM}(\frac{a\ln n}{n}, \frac{b\ln n}{n})$ whenever $a - b < 0.5$ or $a < 1$ (see, Theorem 18). If in addition the vertex locations are known, then we can show a matching lower and upper bounds: the recovery is possible if and only if $a - b > 0.5$ or $a > 1$ (formal statement in full version [16]).

Coming back to the actual recovery problem, our main contribution for GBM is to provide a simple and efficient algorithm that performs well in the connectivity regime and recovers the clusters exactly. The following theorem provides a weaker (but simpler to understand) bound.

▶ **Theorem 7** (Recovery algorithm for GBM). *Suppose we have a graph* $G(V, E)$ *generated according to* $\mathrm{GBM}(r_s \equiv \frac{a\ln n}{n}, r_d \equiv \frac{b\ln n}{n})$ *and* $b > \frac{1}{4\ln 2 - 2}$*, then there exists an efficient algorithm (see Algorithm 1) which recovers the correct partition in* G *with probability* $1 - o(1)$ *if* $a - 8b > 1$*.*

For the full range of parameter b, the (stronger) recovery guarantees for Algorithm 1 is discussed in Theorem 22 in Section 5. Table 1 lists some examples of the parameters when the proposed algorithm (Algorithm 1) can successfully recover the clusters. As can be anticipated, the connectivity results for RAGs apply to the "high dimensional" GBM.

Table 1 Minimum value of a, given b for which Algorithm 1 resolves clusters correctly in the setting for GBM($\frac{a \ln n}{n}, \frac{b \ln n}{n}$).

b	1	2	3	4	5	6	7
Minimum value of a	8.96	12.63	15.9	18.98	21.93	24.78	27.57

▶ **Definition 8** (The GBM in High Dimensions). *Given $V = V_1 \sqcup V_2, |V_1| = |V_2| = \frac{n}{2}$, choose a random vector X_u independently uniformly distributed in S^t for all $u \in V$. The geometric block model* $\mathrm{GBM}_t(r_s, r_d)$ *with parameters $r_s > r_d$ is a random graph where an edge exists between vertices u and v if and only if,*

$$||X_u - X_v||_2 \leq r_s \text{ when } u, v \in V_1 \text{ or } u, v \in V_2$$
$$||X_u - X_v||_2 \leq r_d \text{ when } u \in V_1, v \in V_2 \text{ or } u \in V_2, v \in V_1.$$

We extend the algorithmic results to high dimensions.

▶ **Theorem 9.** *There exists a polynomial time efficient algorithm that recovers the partition from* $\mathrm{GBM}_t(r_s, r_d)$ *with probability $1 - o(1)$ if $r_s = \Theta((\frac{\ln n}{n})^{\frac{1}{t}})$ and $r_s - r_d = \Omega((\frac{\ln n}{n})^{\frac{1}{t}})$. Moreover, any algorithm fails to recover the parts with probability at least $1/2$ if $r_s - r_d = o((\frac{\ln n}{n})^{\frac{1}{t}})$ or $r_s = o((\frac{\ln n}{n})^{\frac{1}{t}})$.*

3 Connectivity of Vertex-Random Graphs

In this section we give a proof of Theorem 2.

3.1 Sufficient condition for connectivity of VRG

▶ **Theorem 10.** *The vertex-random graph* $\mathrm{VRG}(n, [\frac{b \ln n}{n}, \frac{a \ln n}{n}])$ *is connected with probability $1 - o(1)$ if $a > 1$ and $a - b > 0.5$.*

To prove this theorem we use two main technical lemmas that show two different events happen with high probability simultaneously. First, we show that a $\mathrm{VRG}(n, [\frac{b \ln n}{n}, \frac{a \ln n}{n}])$ can be decomposed into union of cycles such that each of them cover $[0, 1]$. Second, we show there exists a vertex u_0 such that it has at least one neighbor in each cycle[2].

▶ **Lemma 11.** *A set of vertices $\mathcal{C} \subseteq V$ is called a cover of $[0, 1]$, if for any point y in $[0, 1]$ there exists a vertex $v \in \mathcal{C}$ such that $d(v, y) \leq \frac{a \ln n}{2n}$. A $\mathrm{VRG}(n, [\frac{b \ln n}{n}, \frac{a \ln n}{n}])$ is a union of cycles such that every cycle forms a cover of $[0, 1]$ as long as $a - b > 0.5$ and $a > 1$ with probability $1 - o(1)$.*

Let us consider a weaker condition $a - b > 1$ than the statement of Lemma 11. This will be much easier to prove and already establishes the connectivity result for RGG in one dimension. Note that since the points are on a circle, it is natural to define a right (clockwise) and a left (counterclockwise) direction. When $a - b > 1$, we show each vertex has at least one neighbor on both directions. To see this for each vertex u , assign two indicator $\{0, 1\}$-random variables A_u^l and A_u^r, with $A_u^l = 1$ if and only if there is no node x to the left

[2] If the points are assumed to be present on a unit line [0,1], the same proof works with a difference that $\mathrm{VRG}(n, [\frac{b \ln n}{n}, \frac{a \ln n}{n}])$ can now be decomposed into a collection of paths that cover [0,1] and all these paths are connected through a vertex u_0. This analysis requires us to handle the nodes present in the boundary region – $[0, \frac{a \ln n}{n}]$ and $[1 - \frac{a \ln n}{n}, 1]$ separately.

of node u such that $d(u,x) \in [\frac{b\ln n}{n}, \frac{a\ln n}{n}]$. Similarly, let $A_u^r = 1$ if and only if there is no node x to the right of node u such that $d(u,x) \in [\frac{b\ln n}{n}, \frac{a\ln n}{n}]$. Now define $A = \sum_u (A_u^l + A_u^r)$. We have,

$$\Pr(A_u^l = 1) = \Pr(A_u^r = 1) = (1 - \frac{(a-b)\ln n}{n})^{n-1},$$

and,

$$\mathbb{E}[A] = 2n(1 - \frac{(a-b)\ln n}{n})^{n-1} \leq 2n^{1-(a-b)}.$$

If $a - b > 1$ then $\mathbb{E}[A] = o(1)$ which implies, by invoking Markov inequality, that with high probability every node will have neighbors (connected by an edge in the VRG) on either side. Therefore every vertex will lie on a cycle that covers $[0,1]$. This is true for every vertex, hence the graph is simply a union of cycles each of which is a cover of $[0,1]$. The main technical challenge is to show that this conclusion remains valid even when $a - b > 0.5$, which is proved in Lemma 11 in Appendix A. Indeed, when $a - b > 0.5$, not every vertex will have neighbors on both sides; rather we need to analyze the connectivity via multi-hops to establish the desired result.

▶ **Lemma 12.** *Set two real numbers $k \equiv \lceil \frac{b}{(a-b)} \rceil + 1$ and $\epsilon < \frac{1}{2k}$. In an* VRG$(n, [\frac{b\ln n}{n}, \frac{a\ln n}{n}])$, *$0 < b < a$, with probability $1 - o(1)$ there exists a vertex u_0 and k nodes $\{u_1, u_2, \ldots, u_k\}$ to the right of u_0 such that $d(u_0, u_i) \in [\frac{(i(a-b)-2i\epsilon)\ln n}{n}, \frac{(i(a-b)-(2i-1)\epsilon)\ln n}{n}]$ and another set of k nodes $\{v_1, v_2, \ldots, v_k\}$ also to the right of u_0 such that $d(u_0, v_i) \in [\frac{((i(a-b)+b-(2i-1)\epsilon)\ln n}{n}, \frac{(i(a-b)+b-(2i-2)\epsilon)\ln n}{n}]$, for $i = 1, 2, \ldots, k$. The arrangement of the vertices is shown in Figure 1.*

We delegate the proof of this lemma to Appendix A.

Proof of Theorem 10. We have shown that the two events mentioned in Lemmas 11 and 12 happen with high probability. Therefore they simultaneously happen under the condition $a > 1$ and $a - b > 0.5$. Now we will show that these events together imply that the graph is connected. To see this, consider the vertices $u_0, \{u_1, u_2, \ldots, u_k\}$ and $\{v_1, v_2, \ldots, v_k\}$ that satisfy the conditions of Lemma 12. We can observe that each vertex v_i has an edge with u_i and u_{i-1}, $i = 1, \ldots, k$. This is because (see Figure 1 for a depiction)

$$d(u_i, v_i) \geq \frac{((i(a-b)+b-(2i-1)\epsilon)\ln n}{n} - \frac{i(a-b)-(2i-1)\epsilon)\ln n}{n} = \frac{b\ln n}{n} \quad \text{and}$$

$$d(u_i, v_i) \leq \frac{i(a-b)+b-(2i-2)\epsilon\ln n}{n} - \frac{(i(a-b)-2i\epsilon)\ln n}{n} = \frac{(b+2\epsilon)\ln n}{n}.$$

Similarly,

$$\begin{aligned} d(u_{i-1}, v_i) &\geq \frac{((i(a-b)+b-(2i-1)\epsilon)\ln n}{n} - \frac{(i-1)(a-b)-(2i-3)\epsilon)\ln n}{n} \\ &= \frac{(a-2\epsilon)\ln n}{n} \quad \text{and} \end{aligned}$$

$$d(u_{i-1}, v_i) \leq \frac{i(a-b)+b-(2i-2)\epsilon\ln n}{n} - \frac{((i-1)(a-b)-2(i-1)\epsilon)\ln n}{n} = \frac{a\ln n}{n}.$$

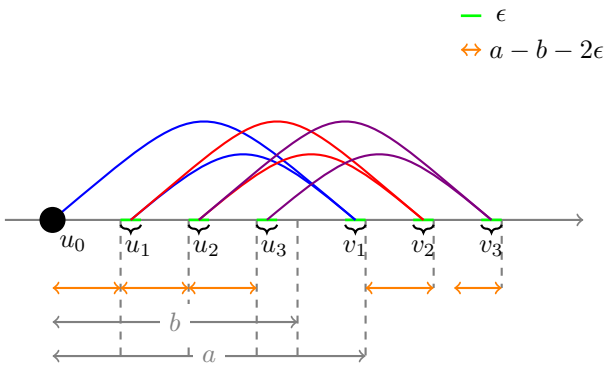


Figure 1 The location of u_i and v_i relative to u scaled by $\frac{\ln n}{n}$ in Lemma 12. Edges stemming put of v_1, v_2, v_3 are shown as blue, red and violet respectively.

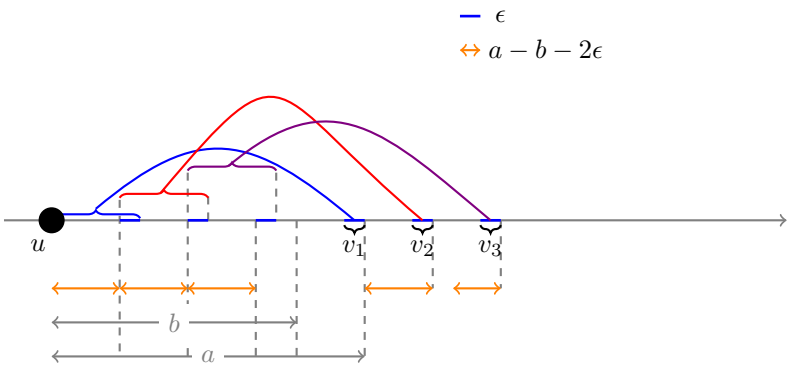


Figure 2 The line segments where v_1, v_2, v_3 can have neighbors (scaled by $\frac{\log n}{n}$) in the proof of Theorem 10. The point t has to lie in one of these regions.

This implies that u_0 is connected to u_i and v_i for all $i = 1, \ldots, k$. Using Lemma 11, the first event implies that the connected components are cycles spanning the entire line $[0, 1]$. Now consider two such disconnected components, one of which consists of the nodes $u_0, \{u_1, u_2, \ldots, u_k\}$ and $\{v_1, v_2, \ldots, v_k\}$. There must exist a node t in the other component (cycle) such that t is on the right of u_0 and $d(u_0, t) \equiv \frac{x \ln n}{n} \leq \frac{a \ln n}{n}$. If $x \leq b$, then there exists an i such that $i \leq k$ and $i(a-b) + b - a - (2i-2)\epsilon \leq x \leq i(a-b) - (2i-1)\epsilon$ (see Figure 2). Thus, when $x \leq b$, we can calculate the distance between t and v_i as

$$d(t, v_i) \geq \frac{(i(a-b) + b - (2i-1)\epsilon) \ln n}{n} - \frac{(i(a-b) - (2i-1)\epsilon) \ln n}{n} = \frac{b \ln n}{n}$$

and

$$d(t, v_i) \leq \frac{(i(a-b) + b - (2i-2)\epsilon) \ln n}{n} - \frac{(i(a-b) + b - a - (2i-2)\epsilon) \ln n}{n} = \frac{a \ln n}{n}.$$

Therefore t is connected to v_i when $x \leq b$. If $x > b$ then t is already connected to u_0. Therefore the two components (cycles) in question are connected.This is true for all cycles and hence there is only a single component in the entire graph. Indeed, if we consider the cycles to be disjoint super-nodes, then we have shown that there must be a star configuration. ◀

The following result is an immediate corollary of the connectivity upper bound.

▶ **Corollary 13.** *Consider a random graph $G(V,E)$ is being generated as a variant of the VRG where each $u,v \in V$ forms an edge if and only if $d(u,v) \in \left[0, c\frac{\ln n}{n}\right] \cup \left[b\frac{\ln n}{n}, a\frac{\ln n}{n}\right], 0 < c < b < a$. This graph is connected with probability $1-o(1)$ if $a-b+c > 1$ or if $a-b > 0.5, a > 1$.*

3.2 Necessary condition for connectivity of VRG

▶ **Theorem 14** (VRG connectivity lower bound)**.** *The* $\mathrm{VRG}(n, [\frac{b\ln n}{n}, \frac{a\ln n}{n}])$ *is not connected with probability $1-o(1)$ if $a < 1$ or $a-b < 0.5$.*

Proof. First of all, it is known that $\mathrm{VRG}(n, [0, \frac{a\ln n}{n}])$ is not connected with high probability when $a < 1$ [25, 26]. Therefore $\mathrm{VRG}(n, [\frac{b\ln n}{n}, \frac{a\ln n}{n}])$ must not be connected with high probability when $a < 1$ as the connectivity interval is a strict subset of the previous case, and $\mathrm{VRG}(n, [\frac{b\ln n}{n}, \frac{a\ln n}{n}])$ can be obtained from $\mathrm{VRG}(n, [0, \frac{a\ln n}{n}])$ by deleting all the edges that has the two corresponding random variables separated by distance less than $\frac{b\ln n}{n}$.

Next we will show that if $a-b < 0.5$ then there exists an isolated vertex with high probability. It would be easier to think of each vertex as a uniform random point in $[0,1]$. Define an indicator variable A_u for every node u which is 1 when node u is isolated and 0 otherwise. We have,

$$\Pr(A_u = 1) = \left(1 - \frac{2(a-b)\ln n}{n}\right)^{n-1}.$$

Define $A = \sum_u A_u$, and hence

$$\mathbb{E}[A] = n\Big(1 - \frac{2(a-b)\ln n}{n}\Big)^{n-1} = n^{1-2(a-b)-o(1)}.$$

Therefore, when $a-b < 0.5$, $\mathbb{E}[A] = \Omega(1)$. To prove this statement with high probability we can show that the variance of A is bounded. Since A is a sum of indicator random variables, we have that

$$\begin{aligned}\mathrm{Var}(A) &\le \mathbb{E}[A] + \sum_{u\neq v} \mathrm{Cov}(A_u, A_v)\\ &= \mathbb{E}[A] + \sum_{u\neq v} (\Pr(A_u = 1 \cap A_v = 1) - \Pr(A_u = 1)\Pr(A_v = 1)).\end{aligned}$$

Now, consider the scenario when the vertices u and v are at a distance more than $\frac{2a\ln n}{n}$ apart (happens with probability $1 - \frac{4a\ln n}{n}$). Then the region in $[0,1]$ that is between distances $\frac{b\ln n}{n}$ and $\frac{a\ln n}{n}$ from both of the vertices is empty and therefore $\Pr(A_u = 1 \cap A_v = 1) = \left(1 - \frac{4(a-b)\ln n}{n}\right)^{n-2}$. When the vertices are within distance $\frac{2a\ln n}{n}$ of one another, then $\Pr(A_u = 1 \cap A_v = 1) \le \Pr(A_u = 1)$. Therefore,

$$\begin{aligned}\Pr(A_u = 1 \cap A_v = 1) \le& (1 - \frac{4a\ln n}{n})\Big(1 - \frac{4(a-b)\ln n}{n}\Big)^{n-2} + \frac{4a\ln n}{n}\Pr(A_u = 1)\\ &\le (1 - \frac{4a\ln n}{n})n^{-4(a-b)+o(1)} + \frac{4a\ln n}{n}n^{-2(a-b)+o(1)}.\end{aligned}$$

Consequently for large enough n,

$$\begin{aligned}\Pr(A_u = 1 \cap A_v = 1) - \Pr(A_u = 1)\Pr(A_v = 1) &\le (1 - \frac{4a\ln n}{n})n^{-4(a-b)+o(1)}\\ &+\frac{4a\ln n}{n}n^{-2(a-b)+o(1)} - n^{-4(a-b)+o(1)} \le \frac{8a\ln n}{n}\Pr(A_u = 1).\end{aligned}$$

Now,

$$\mathrm{Var}(A) \leq \mathbb{E}[A] + \binom{n}{2}\frac{8a\ln n}{n}\Pr(A_u = 1) \leq \mathbb{E}[A](1 + 4a\ln n).$$

By using Chebyshev bound, with probability at least $1 - \frac{1}{\ln n}$,

$$A > n^{1-2(a-b)} - \sqrt{n^{1-2(a-b)}(1 + 4a\ln n)\ln n},$$

which imply for $a - b < 0.5$, there will exist isolated nodes with high probability. ◀

4 Connectivity of High Dimensional Random Annulus Graphs: Proof of Theorem 5

In this section we provide a proof sketch of Theorem 5 to establish the sufficient condition of connectivity of random annulus graphs. The details of the proof and the necessary conditions are provided in the full version [16].

Note, here $r_1 \equiv b\left(\frac{\ln n}{n}\right)^{1/t}$ and $r_2 \equiv a\left(\frac{\ln n}{n}\right)^{1/t}$. To show the upper bound for connectivity, the very first step is to define a *pole* which is a vertex that is connected to all vertices within a distance of r_2 from itself. We show such a pole exists with high probability in Lemma 15. This is a significant generalization of Lemma 12 from Section 3. We prove there exist annuli of suitably small radii around a node u_0 such that they are each non-empty and the vertices in these annuli are connected to each other along with u_0. Moreover the center of the annuli are collinear. Every point within distance r_2 from u_0 is then shown to be connected to at least one vertex in these constructed annuli.

▶ **Lemma 15.** *In a* $\mathrm{RAG}_t\left(n, \left[b\left(\frac{\ln n}{n}\right)^{1/t}, a\left(\frac{\ln n}{n}\right)^{1/t}\right]\right), 0 < b < a$, *with probability* $1 - o(1)$ *there exists a pole.*

Next, Lemma 16 shows that for every vertex u and every hyperplane L passing through u and not too close to the tangent hyperplane at u, there will be a neighbor of u on either side of the plane. Therefore, there should be a neighbor towards the direction of the pole. In order to formalize this, let us define a few regions associated with a node u and a hyperplane $L : w^T x = \beta$ passing through u.

$$\mathcal{R}_L^1 \equiv \{x \in S^t \mid r_1 \leq d(u,x) \leq r_2, w^T x \leq \beta\}$$
$$\mathcal{R}_L^2 \equiv \{x \in S^t \mid r_1 \leq d(u,x) \leq r_2, w^T x \geq \beta\}$$
$$\mathcal{A}_L \equiv \{x \mid x \in \mathcal{S}^t, \quad w^T x = \beta\}.$$

Informally, $\mathcal{R}_L^1$ and $\mathcal{R}_L^2$ represent the partition of the annulus on either side of the hyperplane L and $\mathcal{A}_L$ represents the region on the sphere lying on L.

▶ **Lemma 16.** *If we sample* n *nodes from* S^t *according to* $\mathrm{RAG}_t\left(n, \left[b\left(\frac{\ln n}{n}\right)^{1/t}, a\left(\frac{\ln n}{n}\right)^{1/t}\right]\right)$, *then for every node* u *and every hyperplane* L *passing through* u *such that* $\mathcal{A}_L$ *is not all within distance* r_2 *of* u, *node* u *has a neighbor on both sides of the hyperplane* L *with probability at least* $1 - \frac{1}{n}$ *provided* $(a/2)^t - b^t \geq \frac{8\sqrt{\pi}(t+1)^2\Gamma(\frac{t+2}{2})}{\Gamma(\frac{t+3}{2})}$ *and* $a > 2b$.

The proof of this lemma is quite challenging. Since, we do not know the location of the pole, we need to show that every point has a neighbor on both sides of the plane L no matter what the orientation of the plane. Since the number of possible orientations is uncountably infinite, we

cannot use a union-bound type argument. To show this we have to rely on the VC Dimension of the family of sets $\{x \in S^t \mid r_1 \leq \|u-x\|_2 \leq r_2, w^T x \geq \beta, \mathcal{A}_{L:w^T x=\beta}$ not all within r_2 of $u\}$ for all hyperplanes L (which can be shown to be less than $t+1$). We rely on the celebrated result of [21] (we derive a continuous version of it), see full version [16], to deduce our conclusion.

For a node u and its corresponding location $X_u = (u_1, u_2, \ldots, u_{t+1})$, define the particular hyperplane $L_u^\star : x_1 = u_1$ which is normal to the line joining $u_0 \equiv (1, 0, \ldots, 0)$ and the origin and passes through u. We now need one more lemma that will help us prove Theorem 5.

▶ **Lemma 17.** *For a particular node u and corresponding hyperplane $L_u^\star$, if every point in $\mathcal{A}_{L_u^\star}$ is within distance r_2 from u, then u must be within r_2 of u_0.*

Proof of Theorem 5. We consider an alternate (rotated but not shifted) coordinate system by multiplying every vector by an orthonormal matrix such that the new position of the pole is the $t+1$-dimensional vector $(1, 0, \ldots, 0)$ where only the first coordinate is non-zero. Let the $t+1$ dimensional vector describing any node u in this new coordinate system be $\hat{u} = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_{t+1})$. Now consider the hyperplane $L : x_1 = \hat{u}_1$ and if u is not connected to the pole already, then by Lemma 16 and Lemma 17, the node u has a neighbor u_2 which has a higher first coordinate ($\hat{u}_2 > \hat{u}_1$). The same analysis applies for u_2 and hence we have a path where the first coordinate of every node is higher than the previous node. Since the number of nodes is finite, this path cannot go on indefinitely and at some point, one of the nodes is going to be within r_2 of the pole and will be connected to the pole. Therefore every node is going to be connected to the pole and hence our theorem is proved. ◀

5 The Geometric Block Model

In this section, we prove a necessary condition for exact cluster recovery of the GBM and give an efficient algorithm that matches that within a constant factor. Very interestingly, our algorithm is based on a simple triangle counting method, whose variants are used as popular heuristics for community recovery in many real networks [3, 29, 11]. This further validates the suitability of GBMs as a community detection model.

5.1 Immediate consequence of VRG connectivity

The following lower bound for GBM can be obtained as a consequence of Theorem 2.

▶ **Theorem 18** (Impossibility in GBM). *Any algorithm to recover the partition in* GBM($\frac{a \ln n}{n}$, $\frac{b \ln n}{n}$) *will give incorrect output with probability* $1 - o(1)$ *if* $a - b < 0.5$ *or* $a < 1$.

Proof. Consider the scenario that not only the geometric block model graph GBM($\frac{a \ln n}{n}$, $\frac{b \ln n}{n}$) was provided to us, but also the random values $X_u \in [0, 1]$ for all vertex u in the graph were provided. We will show that we will still not be able to recover the correct partition of the vertex set V with probability at least 0.5 (with respect to choices of X_u, $u, v \in V$ and any randomness in the algorithm).

In this situation, the edge (u, v) where $d_L(X_u, X_v) \leq \frac{b \ln n}{n}$ does not give any new information than X_u, X_v. However the edges (u, v) where $\frac{b \ln n}{n} \leq d_L(X_u, X_v) \leq \frac{a \ln n}{n}$ are informative, as existence of such an edge will imply that u and v are in the same part. These edges constitute a vertex-random graph VRG(n, $[\frac{b \ln n}{n}, \frac{a \ln n}{n}]$). But if there are more than two components in this vertex-random graph, then it is impossible to separate out the vertices into the correct two parts, as the connected components can be assigned to any of the two parts and the VRG along with the location values $(X_u, u \in V)$ will still be consistent.

What remains to be seen that $\text{VRG}(n, [\frac{b \ln n}{n}, \frac{a \ln n}{n}])$ will have $\omega(1)$ components with high probability if $a - b < 0.5$ or $a < 1$. This is certainly true when $a - b < 0.5$ as we have seen in Theorem 14, there can indeed be $\omega(1)$ isolated nodes with high probability. On the other hand, when $a < 1$, just by using an analogous argument it is possible to show that there are $\omega(1)$ vertices that do not have any neighbors on the left direction (counterclockwise). We delegate the proof of this claim as Lemma 19. If there are k such vertices, there must be at least $k - 1$ disjoint candidates. This completes the proof. ◀

▶ **Lemma 19.** *A random geometric graph $G(n, \frac{a \ln n}{n})$ will have $\omega(1)$ disconnected components for $a < 1$.*

Proof. Define an indicator random variable A_u for a node u which is 1 if it does not have a neighbor on its left. We must have that $\Pr(A_u) = \left(1 - \frac{a \ln n}{n}\right)^{n-1}$. Therefore we must have that $\sum_u \mathbb{E}A_u = n^{1-a} = \Omega(1)$ if $a < 1$. This statement also holds true with high probability. To show this we need to prove that the variance of $\sum_u \mathbb{E}A_u$ is bounded. We have

$$\begin{aligned}\text{Var}(A) &< \mathbb{E}[A] + \sum_{u \neq v} \text{Cov}(A_u, A_v) \\ &= \mathbb{E}[A] + \sum_{u \neq v} \Pr(A_u = 1 \cap A_v = 1) - \Pr(A_u = 1)\Pr(A_v = 1)\end{aligned}$$

Now, consider the scenario when the vertices u and v are at a distance more than $\frac{2a \ln n}{n}$ apart (happens with probability at least $1 - \frac{4a \ln n}{n}$). Then the region in $[0, 1]$ that is within distance $\frac{a \ln n}{n}$ from both of the vertices is empty and therefore $\Pr(A_u = 1 \cap A_v = 1) = \Pr(A_u = 1)\Pr(A_v = 1 | A_u = 1) \leq \Pr(A_u = 1)\Pr(A_v = 1) = (\Pr(A_u = 1))^2$. When the vertices are within distance $\frac{2a \ln n}{n}$ of one another, then $\Pr(A_u = 1 \cap A_v = 1) \leq \Pr(A_u = 1)$. Therefore,

$$\Pr(A_u = 1 \cap A_v = 1) \leq (1 - \frac{4a \ln n}{n})(\Pr(A_u = 1))^2 + \frac{4a \ln n}{n} \Pr(A_u = 1).$$

Consequently,

$$\begin{aligned}\Pr(A_u = 1 \cap A_v = 1) - \Pr(A_u = 1)\Pr(A_v = 1) &\leq (1 - \frac{4a \ln n}{n})(\Pr(A_u = 1))^2 \\ &+ \frac{4a \ln n}{n} \Pr(A_u = 1) - (\Pr(A_u = 1))^2 \leq \frac{4a \ln n}{n} \Pr(A_u = 1).\end{aligned}$$

Now,

$$\text{Var}(A) \leq \mathbb{E}[A] + \binom{n}{2} \frac{4a \ln n}{n} \Pr(A_u = 1) \leq \mathbb{E}[A](1 + 2a \ln n).$$

By using Chebyshev bound, with probability at least $1 - \frac{1}{\ln n}$,

$$A > n^{1-a} - \sqrt{n^{1-a}(1 + 2a \ln n) \ln n},$$

Now, observe that if there exist k nodes with no neighbor on one side, then there must exist $k - 1$ disconnected components. Hence the number of components in $G(n, \frac{a \ln n}{n})$ is $\omega(1)$. ◀

Indeed, when the locations X_u associated with every vertex u is provided, it is also possible to recover the partition when $a - b > 0.5$ and $a > 1$, matching the above lower bound exactly. Similar impossibility result extends to higher dimensional GBM from the necessary condition on connectivity of RAG.

5.2 A Recovery Algorithm for GBM

We now turn our attention to an efficient recovery algorithm for GBM. Intriguingly, we show a simple triangle counting based algorithm works well for GBM and recovers the communities in the connectivity regime.

Algorithm 1 Community recovery in GBM.

Require: GBM $G = (V, E)$, r_s, r_d
1: **for** $(u, v) \in E$ **do**
2: **if** process(u, v, r_s, r_d)=false **then**
3: $E.remove((u, v))$
4: **end if**
5: **end for**
6: **return** connectedComponent(V, E)

Algorithm 2 `process`.

Require: u,v, r_s, r_d
Ensure: true/false
{Comment: When $a > 2b$, $t_1 = \min\{t : (2b + t) \ln \frac{2b+t}{2b} - t > 1\}, t_2 = \min\{t : (2b - t) \ln \frac{2b-t}{2b} + t > 1$ and $E_S = (2b + t_1)\frac{\ln n}{n}$ and $E_D = (2b - t_2)\frac{\ln n}{n}\}$
1: count $\leftarrow |\{z : (z, u) \in E, (z, v) \in E\}|$
2: **if** $\frac{\text{count}}{n} \geq E_S(r_d, r_s)$ or $\frac{\text{count}}{n} \leq E_D(r_d, r_s)$ **then**
3: **return** true
4: **end if**
5: **return** false

Suppose we are given a graph $G = (V : |V| = n, E)$ with two disjoint parts, $V_1, V_2 \subseteq V$ generated according to GBM(r_s, r_d). The algorithm (Algorithm 1) goes over all edges $(u, v) \in E$. It counts the number of triangles containing the edge (u, v) by calling the `process` function that counts the number of common neighbors of u and v.

`process` outputs "true" if it is confident that the nodes u and v belong to the same cluster and "false" otherwise. More precisely, if the count is within some prescribed values E_S and E_D, it returns "false". Note that the thresholds E_S and E_D refer to the maximum and minimum value of triangle-count for an "inter-cluster" edge. The algorithm removes the edge on getting a "false" from `process` function. After processing all the edges of the network, the algorithm is left with a reduced graphs (with certain edges deleted from the original). It then finds the connected components in the graph and returns them as the parts V_1 and V_2.

It would have been natural to consider two thresholds E_D and E_S and if the triangle count of an edge is closer to E_S than E_D, then the two end-points are assigned to the same cluster and otherwise in separate clusters. Indeed such a natural algorithm has been analyzed in [15]. On the other hand, here we remove an edge if the triangle count lies in an interval. This is apparently non-intuitive, but gives a significant improvement over the previously known bound (see Figure 3).

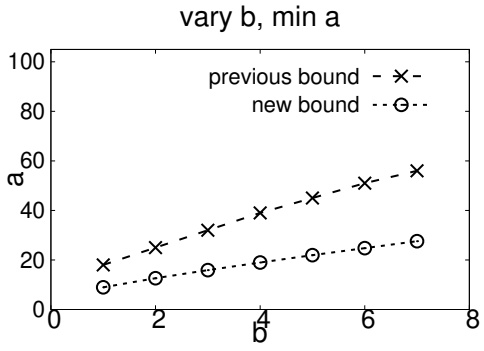


Figure 3 The minimum gap between a and b permitted by our algorithm vs the previously known bound of [15].

5.3 Analysis of Algorithm 1

Given a graph $G(V,E) \equiv GBM(r_s \equiv \frac{a \ln n}{n}, r_d = \frac{b \ln n}{n})$ with two clusters $V = V_1 \sqcup V_2$, and a pair of vertices $u, v \in V$, the events $\mathcal{E}_z^{u,v}, z \in V$ of any other vertex z being a common neighbor of both u and v given $(u,v) \in E$ are dependent; however given the distance between the corresponding random variables $d_L(X_u, X_v) = x$, the events are independent. This is a crucial observation that lets us overcome the difficulty of handling correlated edge formation.

Moreover, given the distance between two nodes u and v are the same, the probabilities of $\mathcal{E}_z^{u,v} \mid (u,v) \in E$ are different when u and v are in the same cluster and when they are in different clusters. Therefore the count of the common neighbors are going to be different, and substantially separated with high probability for two vertices in cases when they are from the same cluster or from different clusters. However, this may not be the case, if we do not restrict the distance to be the same and look at the entire range of possible distances.

The distribution of the number of common neighbors given $(u,v) \in E$ and $d(u,v) = x$ is given in Table 2 (follows from Lemma 23 and Lemma 24 from Appendix). As throughout this paper, we have assumed that there are only two clusters of equal size. In the table, $u \sim v$ means u and v are in the same cluster and $\text{Bin}(n,p)$ denotes a binomial random variable with mean np.

Table 2 Distribution of triangle count for an edge (u,v) conditioned on the distance between them $d(u,v) = d_L(X_u, X_v) = x$, when there are two equal sized clusters.

$(u,v) \in E$	Distribution of count $(r_s > 2r_d)$		Distribution of count $(r_s \leq 2r_d)$	
$d(u,v) = x$	$u \sim v, x \leq r_s$	$u \nsim v, x \leq r_d$	$u \sim v, x \leq r_s$	$u \nsim v, x \leq r_d$
Motif : $z \mid (z,u) \in E, (z,v) \in E$	$\text{Bin}(\frac{n}{2} - 2, 2r_s - x) + \mathbb{1}\{x \leq 2r_d\}\text{Bin}(\frac{n}{2}, 2r_d - x)$	$\text{Bin}(n-2, 2r_d)$	$\text{Bin}(\frac{n}{2} - 2, 2r_s - x) + \text{Bin}(\frac{n}{2}, 2r_d - x)$	$\text{Bin}(n-2, \min(r_s + r_d - x, 2r_d))$

At this point in a $\text{GBM}(r_s, r_d)$ for any edge u, v that does not belong to the same part, the expected total number of common neighbors of u and v does not depend on their distance. In Lemma 20, we show that in this case the normalized total number of common neighbors is concentrated around $2r_d$.

▶ **Lemma 20.** *Suppose we are given a graph $G(V,E)$ generated according to* GBM($r_s \equiv \frac{a\ln n}{n}, r_d \equiv \frac{b\ln n}{n}$), $a \geq 2b$. *Our algorithm with $E_S = (2b+t_1)\frac{\ln n}{n}$ and $E_D = (2b-t_2)\frac{\ln n}{n}$, deletes all the edges $(u,v) \in E$ such that u and v are in different parts with probability at least $1-o(1)$, where*

$$t_1 = \min\{t : (2b+t)\ln\frac{2b+t}{2b} - t > 1\}, \quad t_2 = \min\{t : (2b-t)\ln\frac{2b-t}{2b} + t > 1\}.$$

Therefore, when Algorithm 1 finishes processing all the edges, all the "inter-cluster" edges are removed with high probability. However some of the "in-cluster" edges are also deleted, namely, those that have a count of common neighbors between E_S and E_D. In the next lemma, we show the necessary condition on the "in-cluster" edges such that they do not get removed by Algorithm 1.

▶ **Lemma 21.** *Suppose we are given a graph $G(V,E)$ generated according to* GBM($r_s \equiv \frac{a\ln n}{n}, r_d \equiv \frac{b\ln n}{n}$), $a \geq 2b$. *Define t_1, t_2, E_D, E_S as in Lemma 20. Consider an edge $(u,v) \in E$ where u, v belong to the same part of the GBM and let $d(u,v) \equiv x \equiv \frac{\theta\ln n}{n}$. Suppose θ satisfies either of the following conditions:*

1. $\frac{1}{2}\Big((4b+2t_1)\ln\frac{4b+2t_1}{2a-\theta} + 2a - \theta - 4b - 2t_1\Big) > 1$ *and* $0 \leq \theta \leq 2a - 4b - 2t_1$
2. $\frac{1}{2}\Big((4b-2t_2\ln\frac{4b-2t_2}{2a-\theta} + 2a - \theta - 4b + 2t_2\Big) > 1$ *and* $a \geq \theta \geq \max\{2b, 2a-4b+2t_2\}$..

Then Algorithm 1 with $E_S = (2b+t_1)\frac{\ln n}{n}$ and $E_D = (2b-t_2)\frac{\ln n}{n}$ will not remove this edge with probability at least $1 - O(\frac{1}{n(\ln n)^2})$.

Now we are in a position to prove our main theorem from this part.

▶ **Theorem 22.** *Suppose we are given a graph $G(V,E)$ generated according to* GBM($r_s \equiv \frac{a\ln n}{n}, r_d \equiv \frac{b\ln n}{n}$), $a \geq 2b$. *Define t_1, t_2, E_S and E_D as in Lemma 20, and θ_1 and θ_2 as in Lemma 21. Then Algorithm 1 recovers the correct partition in G with probability $1-o(1)$ if $a - \theta_2 + \theta_1 > 2$ OR $a - \theta_2 > 1, a > 2$.*

Proof. From Lemma 20, we know that after Algorithm 1 has processed all the edges, the edges with end-points in different parts of the GBM are all deleted with probability $1-o(1)$. Moreover, from Lemma 21, an intra-cluster edge (u,v) will continue to exist if $d(u,v) \in [0,\theta_1] \cup [\theta_2, a]$ (by simply applying a union bound over at most $O(n\log n)$ edges). From Corollary 13, it is evident that each of the two parts of size $\frac{n}{2}$ each will be connected if either $a - \theta_2 + \theta_1 > 2$ or $a - \theta_2 > 1$ and $a > 2$. ◀

Theorem 7 is a weaker version of Theorem 22 which we obtain by setting specific values.

Proof of Theorem 7. Following the proof of Theorem 22, when $E_D = 0$ and $E_S = (2b + t_1)\frac{\ln n}{n}$, after Algorithm 1 processes all the edges, an edge between a pair u and v will continue to exist if $d(u,v) \in [0,\theta_1]$ which is equivalent to setting $\theta_2 \leq a$. Consider the case when $b > \frac{1}{4\ln 2 - 2}$. Note that from Theorem 22, $t_1 = \min\{t : (2b+t)\ln\frac{2b+t}{2b} - t > 1\}$. We see that $t = 2b$ satisfies the above condition since, $(2b+t)\ln\frac{2b+t}{2b} - t = 4b\ln 2 - 2b > 1$. This shows that $t_1 \leq 2b$. Similarly, from Theorem 22,

$$\theta_1 = \max\{\theta : \frac{1}{2}\Big((4b+2t_1)\ln\frac{4b+2t_1}{2a-\theta} + 2a - \theta - 4b - 2t_1\Big) > 1 \text{ and } 0 \leq \theta \leq 2a - 4b - 2t_1.\}$$

When $t_1 \leq 2b$, the expression $\theta \leq 2a - 4b - 2t_1$ is satisfied for all values of $\theta \leq 2a - 8b$. Hence, we choose $\theta = 2a - 16b$ to simplify the other expression and get the following chain of equations:

$$\begin{aligned}
&\frac{1}{2}\Big((4b + 2t_1)\ln\frac{4b + 2t_1}{2a - \theta} + 2a - \theta - 4b - 2t_1\Big) \\
&\geq \frac{1}{2}\Big((4b + 2t_1)\ln\frac{4b + 2t_1}{2a - \theta} + 2a - \theta - 8b\Big) = \frac{1}{2}\Big((4b + 2t_1)\ln\frac{4b + 2t_1}{2a - \theta}\Big) + 4b \\
&\geq \frac{1}{2}\Big((4b)\ln\frac{4b}{2a - \theta}\Big) + 4b \geq \frac{1}{2}\Big((4b)\ln\frac{4b}{16b}\Big) + 4b = -2b\ln 4 + 4b
\end{aligned}$$

which is greater than 1 whenever b satisfies $b > \frac{1}{4-4\ln 2}$. However, since we assumed that $b > \frac{1}{2(2\ln 2-1)}$, the condition $b > \frac{1}{4-4\ln 2}$ is automatically satisfied as $\frac{1}{2(2\ln 2-1)} > \frac{1}{4-4\ln 2}$. This implies that $\theta_1 > 2a - 16b$.

Using, $\theta_1 > 2a - 16b$ and $\theta_2 = a$, the final condition of Theorem 22, $a - \theta_2 + \theta_1 > 2$ is satisfied whenever $\theta_1 > 2$ that is, $2a - 16b > 2$. Hence, whenever $2a - 16b > 2$, or, $a - 8b > 1$, Algorithm 1 will recover the correct partition with probability $1 - o(1)$. ◀

References

1. Emmanuel Abbe, Afonso S. Bandeira, and Georgina Hall. Exact Recovery in the Stochastic Block Model. *IEEE Trans. Information Theory*, 62(1):471–487, 2016.
2. Emmanuel Abbe and Colin Sandon. Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery. In *56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 670–688, 2015.
3. Austin R Benson, David F Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016.
4. Béla Bollobás. Random Graphs. *Cambridge Press*, 2001.
5. Béla Bollobás. Percolation. *Cambridge Press*, 2006.
6. Sébastien Bubeck, Jian Ding, Ronen Eldan, and Miklós Z Rácz. Testing for high-dimensional geometry in random graphs. *Random Structures & Algorithms*, pages 503–532, 2016.
7. Peter Chin, Anup Rao, and Van Vu. Stochastic block model and community detection in sparse graphs: A spectral algorithm with optimal rate of recovery. In *Conference on Learning Theory (COLT)*, pages 391–423, 2015.
8. Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, 2011.
9. Carl P Dettmann and Orestis Georgiou. Random geometric graphs with general connection functions. *Physical Review E*, 93(3):032313, 2016.
10. Martin E. Dyer and Alan M. Frieze. The solution of some random NP-hard problems in polynomial expected time. *Journal of Algorithms*, 10(4):451–489, 1989.
11. David Easley and Jon Kleinberg. Networks, crowds, and markets. *Cambridge Books*, 2012.
12. Paul Erdös and Alfréd Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.
13. Stephen Eubank, Hasan Guclu, VS Anil Kumar, Madhav V Marathe, Aravind Srinivasan, Zoltan Toroczkai, and Nan Wang. Modelling disease outbreaks in realistic urban social networks. *Nature*, 429(6988):180, 2004.
14. Ehud Friedgut and Gil Kalai. Every monotone graph property has a sharp threshold. *Proceedings of the American mathematical Society*, 124(10):2993–3002, 1996.
15. Sainyam Galhotra, Arya Mazumdar, Soumyabrata Pal, and Barna Saha. The Geometric Block Model. In *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018.

16 Sainyam Galhotra, Arya Mazumdar, Soumyabrata Pal, and Barna Saha. Connectivity in random annulus graphs and the geometric block model. *arXiv preprint*, 2019. `arXiv:1804.05013`.

17 Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

18 Edward N Gilbert. Random plane networks. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):533–543, 1961.

19 Martin Haenggi, Jeffrey G Andrews, François Baccelli, Olivier Dousse, and Massimo Franceschetti. Stochastic geometry and random graphs for the analysis and design of wireless networks. *IEEE Journal on Selected Areas in Communications*, 27(7):1029–1046, 2009.

20 Bruce E. Hajek, Yihong Wu, and Jiaming Xu. Computational Lower Bounds for Community Detection on Random Graphs. In *Proceedings of The 28th Conference on Learning Theory (COLT)*, pages 899–928, 2015. URL: `http://proceedings.mlr.press/v40/Hajek15.html`.

21 David Haussler and Emo Welzl. epsilon-nets and simplex range queries. *Discrete & Computational Geometry*, 2(2):127–151, 1987.

22 Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

23 Jure Leskovec, Kevin J Lang, Anirban Dasgupta, and Michael W Mahoney. Statistical properties of community structure in large social and information networks. In *17th international conference on World Wide Web*, pages 695–704, 2008.

24 Elchanan Mossel, Joe Neeman, and Allan Sly. Consistency thresholds for the planted bisection model. In *47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 69–75, 2015.

25 S Muthukrishnan and Gopal Pandurangan. The bin-covering technique for thresholding random geometric graph properties. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 989–998, 2005.

26 Mathew Penrose. Random geometric graphs. *Oxford University Press*, 2003.

27 Mathew D Penrose. Connectivity of soft random geometric graphs. *The Annals of Applied Probability*, 26(2):986–1028, 2016.

28 Abishek Sankararaman and François Baccelli. Community Detection on Euclidean Random Graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2181–2200, 2018.

29 Charalampos E Tsourakakis, Jakub Pachocki, and Michael Mitzenmacher. Scalable motif-aware graph clustering. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pages 1451–1460, 2017.

30 Stephen J Young and Edward R Scheinerman. Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 138–149, 2007.

31 Weituo Zhang, Chjan C Lim, Gyorgy Korniss, and Boleslaw K Szymanski. Opinion dynamics and influencing on random geometric graphs. *Scientific reports, Nature Publishing Group*, 4:5568, 2014.

A Proof of Lemma 11 and Lemma 12

Proof of Lemma 11. The proof of this lemma is somewhat easily explained if we consider a weaker result (a stronger condition) with $a - b > 2/3$. Let us first briefly describe this case.

Consider a node u and assume without loss of generality that the position of u is 0 (i.e. $X_u = 0$). Associate four indicator $\{0,1\}$-random variables $A_u^i, i = 1,2,3,4$ which take the value of 1 if and only if there does not exist any node x such that

1. $d(u,x) \in [b\frac{\ln n}{n}, a\frac{\ln n}{n}] \cup [0, \frac{a-b}{2}\frac{\ln n}{n}]\}$ for $i = 1$
2. $d(u,x) \in [b\frac{\ln n}{n}, a\frac{\ln n}{n}] \cup [\frac{-a-b}{2}\frac{\ln n}{n}, -b\frac{\ln n}{n}]\}$ for $i = 2$
3. $d(u,x) \in [-a\frac{\ln n}{n}, -b\frac{\ln n}{n}] \cup [\frac{-a+b}{2}\frac{\ln n}{n}, 0]\}$ for $i = 3$
4. $d(u,x) \in [-a\frac{\ln n}{n}, -b\frac{\ln n}{n}] \cup [b\frac{\ln n}{n}, \frac{a+b}{2}\frac{\ln n}{n}]\}$ for $i = 4$.

The intervals representing these random variables are shown in Figure 4.

Notice that $\Pr(A_u^i = 1) \leq \max\{\left(1 - 1.5(a-b)\frac{\ln n}{n}\right)^{n-1}, \left(1 - a\frac{\ln n}{n}\right)^{n-1}\}$ and therefore $\sum_{i,u} \mathbb{E}A_u^i \leq 4\max\{n^{1-1.5(a-b)}, n^{1-a}\} = 4n^{\min\{1-1.5(a-b),1-a\}}$. This means that for $a - b \geq 0.67$ and $a \geq 1$, $\sum_{i,u} \mathbb{E}A_u^i = o(1)$. Hence there exist vertices in all the regions described above for every node u with high probability.

Now, A_u^1 and A_u^2 being zero implies that either there is a vertex in $[b\frac{\ln n}{n}, a\frac{\ln n}{n}]$ or there exists two vertices v_1, v_2 in $[0, \frac{a-b}{2}\frac{\ln n}{n}]$ and $[\frac{-a-b}{2}\frac{\ln n}{n}, -b\frac{\ln n}{n}]$ respectively (see, Figure 4). In the second case, u is connected to v_2 and v_2 is connected to v_1. Therefore u has nodes on left (v_2) and right (v_1) and u is connected to both of them through one hop in the graph.

Similarly, A_u^3 and A_u^4 being zero implies that either there exists a vertex in $[-a\frac{\ln n}{n}, -b\frac{\ln n}{n}]$ or again u will have vertices on left and right and will be connected to them. So, when all the four $A_u^i, i = 1, 2, 3, 4$ are zero together:

- $A_u^1 = A_u^2 = 0$ implies there is a neighbor of u on either sides or there is a single node in $[b\frac{\ln n}{n}, a\frac{\ln n}{n}]$
- $A_u^3 = A_u^4 = 0$ implies there is a neighbor of u on either sides or there is a single node in $[-a\frac{\ln n}{n}, -b\frac{\ln n}{n}]$

This shows that when $A_u^1 = A_u^2 = 0$ and $A_u^3 = A_u^4 = 0$ guarantee a node on only one side of u, there are nodes in $[b\frac{\ln n}{n}, a\frac{\ln n}{n}]$ and $[-a\frac{\ln n}{n}, -b\frac{\ln n}{n}]$. But in that case u has direct neighbors on both its left and right. We can conclude that every vertex u is connected to a vertex v on its right and a vertex w on its left such that $d(u, v) \in [0, a\frac{\ln n}{n}]$ and $d(u, w) \in [-a\frac{\ln n}{n}, 0]$; therefore every vertex is part of a cycle that covers $[0, 1]$.

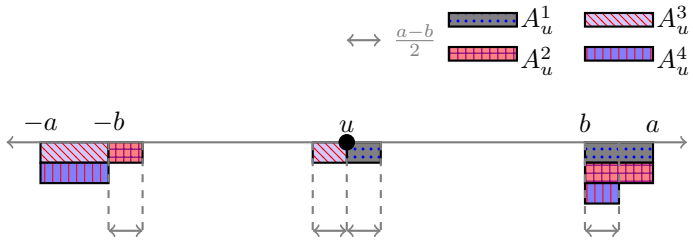


Figure 4 Representation of four different random variables for Lemma 11.

We can now extend this proof to the case when $a - b > 0.5$.

Let c be large number to be chosen specifically later. Consider a node u and assume that the position of u is 0. Now consider the four different regions $[-a\frac{\ln n}{n}, -b\frac{\ln n}{n}]$, $[-(a-b)\frac{\ln n}{n}, 0]$, $[b\frac{\ln n}{n}, a\frac{\ln n}{n}]$ and $[0, a - b\frac{\ln n}{n}]$ around u each divided into $L \equiv 2^c$ patches (intervals) of size $\theta = \frac{a-b}{2^c}$ in the following way:

1. $I_u^i = [\frac{(-a+(i-1)\theta)\ln n}{n}, \frac{(-a+i\theta)\ln n}{n}]$
2. $J_u^i = [\frac{(-(a-b)+(i-1)\theta)\ln n}{n}, \frac{(-(a-b)+i\theta)\ln n}{n}]$
3. $K_u^i = [\frac{(b+(i-1)\theta)\ln n}{n}, \frac{(b+i\theta)\ln n}{n}]$
4. $M_u^i = [\frac{((i-1)\theta)\ln n}{n}, \frac{i\theta\ln n}{n}]$

where $i = 1, 2, 3, \ldots, L$. Note that any vertex in $\cup I_u^i \cup K_u^i$ is connected to u. See, Figure 5 for a depiction.

Consider a $\{0, 1\}$-indicator random variable X_u that is 1 if and only if there does not exist any node in a region formed by union of any $2L - 1$ patches amongst the ones described above. Notice that when $a < 2b$, the patches do not overlap and the total size of $2L - 1$

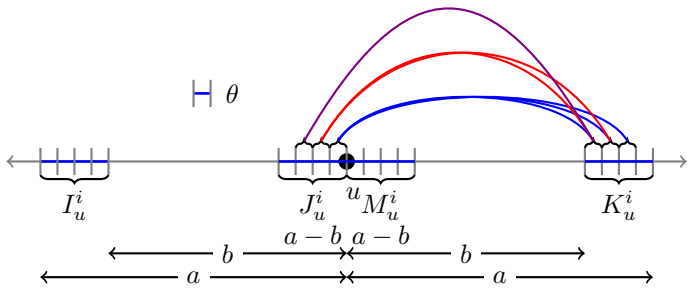


Figure 5 Pictorial representation of $I_u^i, J_u^i, K_u^i, M_u^i$ and their connectivity as described in Lemma 11. The colored lines show the regions that are connected to each other.

patches is $\frac{2^{c+1}-1}{2^c}\frac{(a-b)\ln n}{n}$ and when $a \geq 2b$, the patches can overlap and the total size of the $2L-1$ patches is going to be more than $\min\{\frac{2^{c+1}-1}{2^c}\frac{(a-b)\ln n}{n}, \frac{a\ln n}{n}\}$. Since there are $\binom{4L}{2L-1} \leq n^{\frac{4L}{\ln n}}$ possible regions that consists of $2L-1$ patches,

$$\sum_u \mathbb{E}X_u \leq n\binom{4L}{2L-1}\left(1-\min\{\frac{2^{c+1}-1}{2^c}\frac{(a-b)\ln n}{n}, \frac{a\ln n}{n}\}\right)^{n-1}$$
$$\leq \max\{n^{1-\frac{2^{c+1}-1}{2^c}(a-b)+\frac{4L}{\ln n}}, n^{1-a+\frac{4L}{\ln n}}\}.$$

At this point we can choose $c = c_n = o(\ln n)$ such that $\lim_n c_n = \infty$. Hence when $a-b > \frac{1}{2}$ and $a > 1$, for every vertex u there exists at least one patch amongst every $2L-1$ patches in $\cup I_u^i \cup J_u^j \cup K_u^k, i,j,k = 1,2,\ldots,L$ that contains a vertex.

Consider a collection of patches $\cup_i I_u^i \cup_j K_u^j, i,j = 1,2,\ldots,L$. We know that there exist two patches amongst these I_u^is and K_u^js that contain at least one vertices. If one of I_u^is and one of K_u^js contain two vertices, we found one neighbor of u on both left and right directions (see, Figure 5).

We consider the other case now. Without loss of generality assume that there are no vertex in all I_u^is and there exist at least two patches in K_u^is that contain at least one vertex each. Hence, there exists at least one of $\{K_u^i \mid i \in \{1,2,\ldots,L-1\}\}$ that contains a vertex. Similarly, we can also conclude in this case that there exists at least one of $\{J_u^i \mid i \in \{2,3\ldots,L\}\}$ which contain a node. Assume J_u^ϕ to be the left most patch in $\cup J_u^i \mid i \in \{1,2,\ldots,L\}$ that contains a vertex (see, Figure 5) . From our previous observation, we can conclude that $\phi \geq 2$.

We can observe that any vertex in J_u^j is connected to the vertices in patches $K_u^k, \forall k < j$. This is because for two vertices $v \in J_u^j$ and $w \in K_u^k$, we have

$$d(v,w) \geq \frac{(b+(k-1)\theta)\ln n}{n} - \frac{(-(a-b)+j\theta)\ln n}{n} = \frac{(a+(k-j-1)\theta)\ln n}{n};$$
$$d(v,w) \leq \frac{(b+k\theta)\ln n}{n} - \frac{(-(a-b)+(j-1)\theta)\ln n}{n} = \frac{(a+(k-j+1)\theta)\ln n}{n}.$$

Consider a collection of $2L-1$ patches $\{\cup I_u^i \cup J_u^j \cup K_u^k \mid i,j,k \in \{1,\ldots,L\}, j > \phi, k \leq \phi-1\}$ where $\phi \geq 2$. This is a collection of $2L-1$ patches out of which one must have a vertex and since none of $\{J_u^j \mid j > \phi\}$ and I_u^i can contain a vertex, one of $\{K_u^k \mid k \leq \phi-1\}$ must contain the vertex. Recall that the vertex in J_u^ϕ is connected to any node in K_u^k for any $k \leq \phi-1$ and therefore u has a node to the right direction and left direction that are connected to u. Therefore every vertex is part of a cycle and each of the circles covers $[0,1]$. ◀

Proof of Lemma 12. Recall that we want to show that there exists a node u_0 and k nodes $\{u_1, u_2, \ldots, u_k\}$ to the right of u_0 such that $d(u_0, u_i) \in [\frac{(i(a-b)-2i\epsilon)\ln n}{n}, \frac{(i(a-b)-(2i-1)\epsilon)\ln n}{n}]$ and exactly k nodes $\{v_1, \ldots, v_k\}$ to the right of u_0 such that $d(u_0, v_i) \in [\frac{((i(a-b)+b-(2i-1)\epsilon)\ln n}{n}, \frac{(i(a-b)+b-(2i-2)\epsilon)\ln n}{n}]$, for $i = 1, 2, \ldots, k$ and ϵ is a constant less than $\frac{1}{2k}$ (see Figure 1 for a depiction). Let A_u be an indicator $\{0,1\}$-random variable for every node u which is 1 if u satisfies the above conditions and 0 otherwise. We will show $\sum_u A_u \geq 1$ with high probability. We have,

$$\begin{aligned}\Pr(A_u = 1) &= n(n-1)\ldots(n-(2k-1))\Big(\frac{\epsilon \ln n}{n}\Big)^{2k}\Big(1 - 2k\epsilon\frac{\ln n}{n}\Big)^{n-2k} \\ &= c_0 n^{-2k\epsilon}(\epsilon \ln n)^{2k}\prod_{i=0}^{2k-1}(1 - i/n) = c_1 n^{-2k\epsilon}(\epsilon \ln n)^{2k}\end{aligned}$$

where c_0, c_1 are just absolute constants independent of n (recall k is a constant). Hence,

$$\sum_u \mathbb{E}A_u = c_1 n^{1-2k\epsilon}(\epsilon \ln n)^{2k} \geq 1$$

as long as $\epsilon \leq \frac{1}{2k}$. Now, in order to prove $\sum_u A_u \geq 1$ with high probability, we will show that the variance of $\sum_u A_u$ is bounded from above. This calculation is very similar to the one in the proof of Theorem 14. Recall that if $A = \sum_u A_u$ is a sum of indicator random variables, we must have

$$\mathrm{Var}(A) \leq \mathbb{E}[A] + \sum_{u \neq v}\mathrm{Cov}(A_u, A_v) = \mathbb{E}[A] + \sum_{u\neq v}\Pr(A_u = 1 \cap A_v = 1) - \Pr(A_u = 1)\Pr(A_v = 1).$$

Now first consider the case when vertices u and v are at a distance of at least $\frac{2(a+b)\ln n}{n}$ apart (happens with probability $1 - \frac{4(a+b)\ln n}{n}$). Then the region in $[0,1]$ that is within distance $\frac{(a+b)\ln n}{n}$ from both u and v is the empty-set. In this case, $\Pr(A_u = 1 \cap A_v = 1) = n(n-1)\ldots(n-(4k-1))\Big(\frac{\epsilon \ln n}{n}\Big)^{4k}\Big(1 - 4k\epsilon\frac{\ln n}{n}\Big)^{n-4k} = c_2 n^{-4k\epsilon}(\epsilon \ln n)^{4k}$, where c_2 is a constant.

In all other cases, $\Pr(A_u = 1 \cap A_v = 1) \leq \Pr(A_u = 1)$. Therefore,

$$\begin{aligned}\Pr(A_u = 1 \cap A_v = 1) \leq &\Big(1 - \frac{4(a+b)\ln n}{n}\Big)c_2 n^{-4k\epsilon}(\epsilon \ln n)^{4k} \\ &+ \frac{4(a+b)\ln n}{n}c_1 n^{-2k\epsilon}(\epsilon \ln n)^{2k}\end{aligned}$$

and

$$\begin{aligned}\mathrm{Var(A)} &\leq c_1 n^{1-2k\epsilon}(\epsilon \ln n)^{2k} + \binom{n}{2}\Big(\Pr(A_u = 1 \cap A_v = 1) - \Pr(A_u = 1)\Pr(A_v = 1)\Big) \\ &\leq c_1 n^{1-2k\epsilon}(\epsilon \ln n)^{2k} + c_3 n^{1-2k\epsilon}(\ln n)^{2k+1} \leq c_4 n^{1-2k\epsilon}(\ln n)^{2k+1}\end{aligned}$$

where c_3, c_4 are constants. Again invoking Chebyshev's inequality, with probability at least $1 - \frac{1}{\ln n}$

$$A > c_1 n^{1-2k\epsilon}(\epsilon \ln n)^{2k} - \sqrt{c_4 n^{1-2k\epsilon}(\ln n)^{2k+2}}.$$ ◀

B Missing Proofs of Section 5

▶ **Lemma 23.** *For any two vertices $u, v \in V_i : (u, v) \in E, i = 1, 2$ belonging to the same cluster with $d_L(X_u, X_v) = x$, the count of common neighbors $C_{u,v} \equiv |\{z \in V : (z, u), (z, v) \in E\}|$ is a random variable distributed according to $\mathrm{Bin}(\frac{n}{2} - 2, 2r_s - x)$ when $r_s \geq x > 2r_d$ and according to $\mathrm{Bin}(\frac{n}{2} - 2, 2r_s - x) + \mathrm{Bin}(\frac{n}{2}, 2r_d - x)$ when $x \leq \min(2r_d, r_s)$, where $\mathrm{Bin}(n, p)$ is a binomial random variable with mean np.*

Proof. Without loss of generality, assume $u, v \in V_1$. For any vertex $z \in V$, let $\mathcal{E}_z^{u,v} \equiv \{(u, z), (v, z) \in E\}$ be the event that z is a common neighbor. For $z \in V_1$,

$$\begin{aligned}\Pr(\mathcal{E}_z^{u,v}) &= \Pr((z, u) \in E, (z, v) \in E) \\ &= 2r_s - x,\end{aligned}$$

since $d_L(X_u, X_v) = x$. For $z \in V_2$, we have,

$$\begin{aligned}\Pr(\mathcal{E}_z^{u,v}) &= \Pr((z, u), (z, v) \in E) \\ &= \begin{cases} 2r_d - x & \text{if } x < 2r_d \\ 0 & \text{otherwise} \end{cases}.\end{aligned}$$

Now since there are $\frac{n}{2} - 2$ points in $V_1 \setminus \{u, v\}$ and $\frac{n}{2}$ points in V_2, we have the statement of the lemma. ◀

In a similar way, we can prove.

▶ **Lemma 24.** *For any two vertices $u \in V_1, v \in V_2 : (u, v) \in E$ belonging to different clusters with $d_L(X_u, X_v) = x$, the count of common neighbors $C_{u,v} \equiv |\{z \in V : (z, u), (z, v) \in E\}|$ is a random variable distributed according to $\mathrm{Bin}(n - 2, 2r_d)$ when $r_s > 2r_d$ and according to $\mathrm{Bin}(n - 2, \min(r_s + r_d - x, 2r_d))$ when $r_s \leq 2r_d$ and $x \leq r_d$.*

Proofs of Lemma 20 and Lemma 21

Proof of Lemma 20. Here we will use the fact that for $a \geq 1$, the number of edges in $\mathrm{GBM}(r_s \equiv \frac{a \ln n}{n}, r_d \equiv \frac{b \ln n}{n})$ is $O(n \ln n)$ with probability $1 - \frac{1}{n^{\Theta(1)}}$. Consider any vertex $u \in V_1$ (symmetrically for $u \in V_2$), since the vertices are thrown uniformly at random in $[0, 1]$, the probability that a $v \in V_1$, $v \neq u$, is a neighbor of u is $\frac{a \ln n}{n}$, and for $v \in V_2$, the corresponding probability is $\frac{b \ln n}{n}$. Therefore, the expected degree of u is $\frac{(a+b)}{2} \ln n$. By a simple Chernoff bound argument, the degree of u is therefore $O(\ln n)$ with probability $1 - \frac{1}{n^c}$ for $c \geq 2$. By union bound over all the vertices, the total number of edges is $O(n \ln n)$ with probability $1 - \frac{1}{n}$.

Let Z denote the random variable that equals the number of common neighbors of two nodes $u, v \in V : (u, v) \in E$ such that u, v are from different parts of the GBM. Using Lemma 24, we know that Z is sampled from the distribution $\mathrm{Bin}(n - 2, 2r_d)$, where $r_d = \frac{b \ln n}{n}$. Therefore,

$$\Pr(Z \geq nE_S) \leq \sum_{i=nE_S}^{n} \binom{n}{i} (2r_d)^i (n - 2r_d)^{n-i} \leq \exp\Big(- nD\Big((2b + t_1)\frac{\ln n}{n} \| \frac{2b \ln n}{n}\Big)\Big),$$

where $D(p\|q) \equiv p \ln \frac{p}{q} + (1 - p) \ln \frac{1-p}{1-q}$ is the KL divergence between Bernoulli(p) and Bernoulli(q) distributions. It is easy to see that,

$$nD(\frac{\alpha \ln n}{n} || \frac{\beta \ln n}{n}) = \Big(\alpha \ln \frac{\alpha}{\beta} + (\alpha - \beta)\Big) \ln n - o(\ln n).$$

Therefore $\Pr(Z \geq nE_S) \leq \frac{1}{n(\ln n)^2}$ because $(2b+t_1)\ln\frac{2b+t_1}{2b} - t_1 > 1$. Similarly, we have that

$$\begin{aligned}\Pr(Z \leq nE_D) &\leq \sum_{i=0}^{nE_D}\binom{n}{i}(2r_d)^i(n-2r_d)^{n-i} \leq \exp(-nD((2b-t)\frac{\ln n}{n}\|\frac{2b\ln n}{n})) \\ &\leq \frac{1}{n(\ln n)^2}.\end{aligned}$$

So all of the inter-cluster edges will be removed by Algorithm 1 with probability $1 - O(\frac{n\ln n}{n(\ln n)^2}) = 1 - o(1)$, as with probability $1-o(1)$ the total number of edges in the graph is $O(n\ln n)$. ◀

Proof of Lemma 21. Let Z be the number of common neighbors of u, v. Recall that, u and v are in the same cluster. We know from Lemma 24 that Z is sampled from the distribution $\text{Bin}(\frac{n}{2}-2, 2r_s - x) + \text{Bin}(\frac{n}{2}, 2r_d - x)$ when $x \leq 2r_d$, and from the distribution $\text{Bin}(\frac{n}{2}-2, 2r_s - x)$ when $x \geq 2r_d$. We have,

$$\begin{aligned}&\Pr(Z \leq nE_S) \\ &= \begin{cases}\sum_{i=0}^{nE_S}\binom{\frac{n}{2}-2}{i}(2r_s-x)^i(1-2r_s+x)^{\frac{n}{2}-i-2} \\ \times \sum_{j=0}^{nE_S-i}\binom{\frac{n}{2}}{j}(2r_d-x)^j(1-2r_d+x)^{\frac{n}{2}-j} \text{ if } x \leq 2r_d \\ \sum_{i=0}^{nE_s}\binom{\frac{n}{2}-2}{i}(2r_s-x)^i(1-2r_s+x)^{\frac{n}{2}-i} \text{ otherwise}\end{cases} \\ &\leq e^{-\frac{n}{2}D(2E_S\|\frac{(2a-\theta)\ln n}{n})} \text{ since } 2a-\theta \geq 4b+2t_1 \\ &\leq e^{-\frac{n}{2}D(\frac{(4b+2t_1)\ln n}{n}\|\frac{(2a-\theta)\ln n}{n})} \leq \frac{1}{n\ln^2 n},\end{aligned}$$

because of Condition 1 of this lemma. Therefore, this edge will not be deleted with high probability.

Similarly, let us find the probability of $Z \geq nE_D = (2b-t_2)\ln n$. Let us just assume the worst case when $\theta \leq 2b$: that the edge is being deleted (see Condition 2, this is prohibited if that condition is satisfied). Otherwise, $\theta > 2b$ and,

$$\begin{aligned}\Pr(Z \geq nE_D) &= \sum_{i=nE_D}^{n}\binom{\frac{n}{2}-2}{i}(2r_s-x)^i(1-2r_s+x)^{\frac{n}{2}-i-2} \\ &\leq e^{-\frac{n}{2}D(2E_D\|\frac{(2a-\theta)\ln n}{n})} \text{ if } 2a-\theta \leq 4b-2t_2 \\ &= e^{-\frac{n}{2}D(\frac{(4b-2t_2)\ln n}{n}\|\frac{(2a-\theta)\ln n}{n})} \leq \frac{1}{n\ln^2 n}\end{aligned}$$

because of Condition 2 of this lemma. ◀

A Local Stochastic Algorithm for Separation in Heterogeneous Self-Organizing Particle Systems



Sarah Cannon
Claremont McKenna College, Claremont, CA, USA
scannon@cmc.edu



Joshua J. Daymude
Computer Science, CIDSE, Arizona State University, Tempe, AZ, USA
jdaymude@asu.edu



Cem Gökmen
Georgia Institute of Technology, Atlanta, GA, USA
cgokmen@gatech.edu

Dana Randall
Georgia Institute of Technology, Atlanta, GA, USA
randall@cc.gatech.edu

Andréa W. Richa
Computer Science, CIDSE, Arizona State University, Tempe, AZ, USA
aricha@asu.edu

Abstract

We present and rigorously analyze the behavior of a distributed, stochastic algorithm for *separation* and *integration* in *self-organizing particle systems*, an abstraction of programmable matter. Such systems are composed of individual computational *particles* with limited memory, strictly local communication abilities, and modest computational power. We consider *heterogeneous* particle systems of two different colors and prove that these systems can collectively *separate* into different color classes or *integrate*, indifferent to color. We accomplish both behaviors with the same fully distributed, local, stochastic algorithm. Achieving separation or integration depends only on a single global parameter determining whether particles prefer to be next to other particles of the same color or not; this parameter is meant to represent external, environmental influences on the particle system. The algorithm is a generalization of a previous distributed, stochastic algorithm for *compression* (PODC '16) that can be viewed as a special case of separation where all particles have the same color. It is significantly more challenging to prove that the desired behavior is achieved in the heterogeneous setting, however, even in the bichromatic case we focus on. This requires combining several new techniques, including the *cluster expansion* from statistical physics, a new variant of the *bridging* argument of Miracle, Pascoe and Randall (RANDOM '11), the *high-temperature expansion* of the Ising model, and careful probabilistic arguments.

2012 ACM Subject Classification Mathematics of computing → Stochastic processes; Theory of computation → Random walks and Markov chains; Theory of computation → Self-organization

Keywords and phrases Markov chains, Programmable matter, Cluster expansion

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.54

Category RANDOM

Related Version A full version is available online at `https://arxiv.org/abs/1805.04599`.

Funding *Sarah Cannon*: Supported by National Science Foundation (NSF) award DMS-1803325.
Joshua J. Daymude: Supported by NSF awards CCF-1422603, CCF-1637393, and CCF-1733680.
Cem Gökmen: Supported by NSF award CCF-1733812.
Dana Randall: Supported by NSF awards CCF-1526900, CCF-1637031, and CCF-1733812.
Andréa W. Richa: Supported by NSF awards CCF-1422603, CCF-1637393, and CCF-1733680.



© Sarah Cannon, Joshua J. Daymude, Cem Gökmen, Dana Randall, and Andréa W. Richa;
licensed under Creative Commons License CC-BY

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh; Article No. 54; pp. 54:1–54:22



Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Across many disciplines spanning computational, physical, and social sciences, heterogeneous systems self-organize into both separated (or segregated) and integrated states. Examples include molecules exhibiting attractive and repulsive forces, distinct types of bacteria competing for resources while collaborating towards common goals (e.g., [35, 39]), social insects tolerating or aggressing towards those from other colonies (e.g., [20, 30]), and inherent human biases that influence how we form and maintain social groups (e.g., [16, 37]). In each of these, individuals are of different "types": integration occurs when the ensemble gathers together without much preference about the type of their neighbors, while separation occurs when individuals cluster with others of the same type. Here, we investigate these fundamental behaviors of separation or integration as they apply to *programmable matter*, a material that can alter its physical properties based on user input or stimuli from its environment. Instead of studying a particular instantiation of programmable matter, of which there are many [1, 7, 31, 36], we abstractly envision these systems as collections of simple, active computational *particles* that individually execute local distributed algorithms to collectively achieve some emergent behavior. We consider *heterogeneous* particle systems in which particles have immutable *colors*. We seek local, distributed algorithms that, when run by each particle independently and concurrently, result in emergent, self-organizing *separation* or *integration* of color classes.

This work uses the *stochastic approach to self-organizing particle systems* first used for *compression*, where (monochromatic) particles self-organize to gather together as tightly as possible [6]. Using this stochastic approach, one first defines an energy function where desired configurations have the lowest energy values. One then designs a Markov chain whose long run behavior favors these low energy configurations. This Markov chain is carefully designed so that all its transition probabilities can be computed locally, allowing it to be translated to a fully local distributed algorithm each particle can run independently. The resulting collective, emergent behavior of this distributed algorithm is thus described by the long run behavior of the Markov chain. Using this stochastic approach, we previously extended our compression algorithm [6] to an algorithm for *shortcut bridging* [2] – or maintaining bridge structures that balance the tradeoff between bridge efficiency and cost – and developed the theoretical basis for an experimental study in swarm robotics [32]. While the process of designing distributed algorithms for self-organizing particle systems via this stochastic approach is fairly well-understood, proving that such algorithms achieve their desired objectives remains quite challenging. In particular, it is not enough to know the desired configurations have the highest long-run probability; there may be so many other, lower probability configurations that they collectively outweigh the desirable ones. This energy/entropy trade-off has been studied in various Markov chains for the purposes of proving slow mixing, but we analyze it directly to show our algorithms achieve the desired objectives with high probability.

Here, we focus on separation and integration in heterogeneous systems. Our inspiration comes from the classical Ising model in statistical physics [18, 38], where the vertices of a graph are assigned positive and negative "spins" and there are rules governing the probability that adjacent vertices have the same spin. Connected to the Ising model is classical work from stochastic processes on the Schelling model of segregation [33, 34], which explores how individuals' micro-motives can induce macro-level phenomena like racial segregation in residential neighborhoods. Recent variants of this model from computer science have investigated the degree of individual bias required to induce such segregation [5, 17], and a related distributed algorithm has been developed [29]. Our work differs from those on

the Ising and Schelling models because of natural physical constraints on systems of self-organizing, active particles like ours. For example, interpreting particles of one color to be vertices with positive spin and particles of another color to be particles with negative spin, this acts like an Ising model, but on a graph that evolves as particles more. Despite these obstacles, we apply ideas developed for rigorously analyzing the Ising and similar models to prove our distributed algorithm for separation and integration accomplishes the desired goals.

While we are interested in distributed algorithms, it is worth noting that efficient stochastic algorithms for separation can be challenging even with centralized Markov chains. Separation of a region into equitably sized, compact districts has been widely explored recently in the context of gerrymandering, where the aim is to sample colorings of a weighted graph from an appropriately defined stationary distribution [10, 15]. Heuristics for random districting have been discussed in the media, but there are still no known rigorous, efficient algorithms.

1.1 Results

We present a distributed algorithm for self-organizing separation and integration that takes as input two bias parameters, λ and γ. Setting $\lambda > 1$ corresponds to particles favoring having more neighbors; this is known to cause compression in homogeneous systems when λ is large enough [6]. For separation in the heterogeneous setting, we introduce a second parameter γ, where $\gamma > 1$ corresponds to particles favoring having more neighbors *of their own color*. We then investigate for what values of λ and γ our algorithm yields compression and separation. Informally, a particle system is separated if there is a subset of particles such that (*i*) the boundary between this subset and the rest of the system is small, (*ii*) a large majority of particles in this subset are of the same color, say c, and (*iii*) very few particles with color c exist outside of this subset. This notion of separation (defined formally in Definition 3) captures what it means for a system to have large monochromatic regions of particles.

We prove that for any $\lambda > 1$ and $\gamma > 4^{5/4} \sim 5.66$ such that $\lambda\gamma > 2(2+\sqrt{2})e^{0.0003} \sim 6.83$, our algorithm accomplishes separation with high probability.[1] However, we prove the opposite for some values of γ close to one; counterintuitively, this even includes some values of $\gamma > 1$, the regime where particles favor having like-colored neighbors. Formally, we prove that for any $\lambda > 1$ and $\gamma \in (79/81, 81/79)$ such that $\lambda(\gamma+1) > 2(2+\sqrt{2})e^{0.00003} \sim 6.83$, our algorithm fails to achieve separation (i.e., it achieves integration) with high probability.

1.2 Proof Techniques

Because our distributed algorithm is based on a Markov chain, we can use standard tools such as detailed balance to understand its long-term behavior and prove its convergence to a unique probability distribution π over particle system configurations. This stationary distribution π depends on the input parameters λ and γ. Our main contribution is analyzing π for various ranges of λ and γ, showing that a configuration drawn from distribution π is either very likely (for large γ) or very unlikely (for γ close to one) to be separated.

To show separation occurs when λ and γ are both large, we modify the proof technique of *bridging* introduced by Miracle, Pascoe, and Randall [28]. To show separation does not occur when λ is large and γ is small (close to one), we use a probabilistic argument, a Chernoff-type bound, and a decomposition of configurations into different regions. These arguments – both

[1] We say an event A occurs with high probability (w.h.p.) if $\Pr[A] \geq 1 - c^{n^\delta}$, where $0 < c < 1$ and $\delta > 0$ are constants and n is the number of particles. Our w.h.p. results all have $\delta \in \{1/2, 1/2 - \varepsilon\}$, for arbitrarily small $\varepsilon > 0$.

for large and small γ – require that the particle system is compressed; i.e., that the system has perimeter $\Theta(\sqrt{n})$. However, the arguments from [6] showing compression occurs for homogeneous systems when λ is large do not extend to the heterogeneous setting.

We instead turn to the *cluster expansion* from statistical physics to show our separation algorithm achieves compression for large enough γ. The cluster expansion was first introduced in 1937 by Mayer [27], though a more modern treatment can be found in the textbook [12] where it is used to derive several properties of statistical physics models including the Ising and hard-core models. In the past year, the cluster expansion has received renewed attention in the computer science community due to the recent work of Helmuth, Perkins, and Regts that uses the cluster expansion to develop approximate counting and sampling algorithms for low-temperature statistical physics models on lattices including the Potts and hard-core models [14]. Subsequent work has considered similar techniques on expander graphs [19] and random regular bipartite graphs [23]. Inspired by the interpolation method of Barvinok [3, 4], these works give algorithms for estimating partition functions that explicitly calculate the first $\log n$ coefficients of the cluster expansion. We use the cluster expansion differently, to separate the volume and surface contributions to a partition function.

The cluster expansion is a power series representation of $\ln Z$ where Z is a *polymer partition function*. We relate each of our quantities of interest to a particular polymer partition function, and then use a version of the Kotecký-Preiss condition [21] to show that the power series in the cluster expansion is convergent for the ranges of parameters we are interested in. We then use this convergent cluster expansion to split our polymer partition functions into a *volume term*, depending only on the size of the region of interest, and a *surface term*, depending only on its perimeter. This separation into volume and surface terms turns out to be the key to our compression argument, both for large γ and for γ close to one. While splitting partition functions into volume and surface terms is not a new idea in the statistical physics literature (for example, Section 5.7.1 of [12] uses it to derive an explicit expression for the infinite volume pressure of the Ising model on $\mathbb{Z}^d$ with large magnetic field), we are the first to bring this approach into the computer science literature. We are hopeful it will be useful beyond its specific applications in this paper.

2 Background

We begin by defining our amoebot model for programmable matter and stating a few key results. We then extend the amoebot model to heterogeneous particle systems and formally define what it means for a system to be separated or integrated. We conclude with the necessary terminology and results on Markov chains.

2.1 The Amoebot Model

In the *amoebot model*, introduced in [9] and fully described in [8], programmable matter consists of individual, homogeneous computational elements called *particles*. In its geometric variant, particles are assumed to occupy nodes of the triangular lattice $G_\Delta = (V, E)$ and can move along its edges (see Figure 1a). Each node in V can be occupied by at most one particle at a time. Each particle occupies either a single node in V (i.e., it is *contracted*) or a pair of adjacent nodes in V (i.e., it is *expanded*), as in Figure 1b. Particles move via a series of *expansions* and *contractions*: a contracted particle can expand into an unoccupied adjacent node to become expanded, and completes its movement by contracting to once again occupy a single node.



Figure 1 (a) A section of the triangular lattice G_Δ. (b) Expanded and contracted particles (black dots) on G_Δ (gray lattice). Particles with a black line between their nodes are expanded.

Two particles occupying adjacent nodes are said to be *neighbors*. Each particle is *anonymous*, lacking a unique identifier, but can locally identify each of its neighboring locations and can determine which of these are occupied by particles. Each particle has a constant-size local memory that it can write to and its neighbors can read from for communication. In particular, a particle stores whether it is contracted or expanded in its memory. Particles do not have any access to global information such as a shared compass, coordinate system, or estimate of the size of the system.

The system progresses through *atomic actions* according to the standard asynchronous model of computation from distributed computing (see, e.g., [25]). A classical result under this model states that for any concurrent asynchronous execution of atomic actions, there exists a sequential ordering of actions producing the same end result, provided conflicts that arise in the concurrent execution are resolved. In the amoebot model, an atomic action corresponds to the activation of a single particle. Once activated, a particle can (*i*) perform a constant amount of computation involving information it reads from its local memory and its neighbors' memories, (*ii*) write to its local memory, and (*iii*) perform at most one expansion or contraction. Conflicts involving simultaneous particle expansions into the same unoccupied node are assumed to be resolved arbitrarily such that at most one particle moves to some unoccupied node at any given time. Thus, while in reality many particles may be active concurrently, it suffices when analyzing algorithms under the amoebot model to consider a sequence of activations where only one particle is active at a time.

2.2 Terminology and Results for Homogeneous Particle Systems

We now recall the relevant terminology and notation from our previous work on compression [6]. A particle system *arrangement* is the set of vertices of the triangular lattice G_Δ occupied by particles. Two arrangements are equivalent if they are translations of each other; we define a particle system *configuration* to be an equivalence class of arrangements. An *edge* of a configuration is an edge of G_Δ where both endpoints are occupied by particles. A configuration is *connected* if for any two particles in the system, there is a path of such edges between them. A configuration has a *hole* if there is a maximal, finite, connected component of unoccupied vertices in G_Δ.

As we justify with Lemma 6, our analysis will focus on connected, hole-free configurations. The *boundary* of such a configuration σ is the closed walk $\mathcal{P}$ on edges of σ that encloses all particles of σ and no unoccupied vertices of G_Δ. The *perimeter* $p(\sigma)$ of configuration σ is the length of this walk, also denoted $|\mathcal{P}|$. The following bounds the number of configurations with a given perimeter.

▶ **Lemma 1** ([6], Lemma 4.3). *For any $\nu > 2+\sqrt{2}$, there is an integer $n_1(\nu)$ such that for all $n \geq n_1(\nu)$, the number of connected, hole-free particle system configurations with n particles and perimeter k is at most ν^k.*

Let $p_{min}(n)$ be the minimum possible perimeter for a configuration of n particles; it is easy to see that $p_{min}(n) = \Theta(\sqrt{n})$. Given any $\alpha > 1$, a configuration of n particles is said to be *α-compressed* if $p(\sigma) \leq \alpha \cdot p_{min}(n)$. The following lemma establishes a concrete upper bound on $p_{min}(n)$.

▶ **Lemma 2.** *For any $n \geq 1$, there is a connected, hole-free particle system configuration of n particles with perimeter at most $2\sqrt{3}\sqrt{n}$. That is, $p_{min}(n) \leq 2\sqrt{3}\sqrt{n}$.*

Proof. This lemma follows easily from noting that hexagonal configurations of n particles have perimeter on the order of $2\sqrt{3}\sqrt{n}$; a proof can be found in Appendix A.1. ◀

2.3 Heterogeneous Particle Systems

Generalizing previous work on the amoebot model in which all particles are homogeneous and indistinguishable, we assume that each particle P has a fixed *color* $c(P) \in \{c_1, \ldots, c_k\}$ that is visible to itself and its neighbors, where $k \ll n$ is a constant. We extend the definition of *configuration* given in Section 2.2 to include both the vertices of G_Δ occupied by particles as well as the colors of those particles. An edge of configuration σ with endpoints occupied by particles P and Q is *homogeneous* if $c(P) = c(Q)$ and *heterogeneous* otherwise.

We further extend the original model by allowing neighboring particles to exchange their positions in a *swap move*. Swap moves have no meaning in homogeneous systems as all particles are indistinguishable, but they grant heterogeneous systems flexibility in allowing particles trapped in the interior of the system to move freely.[2] These swap moves are not necessary for the correctness of our algorithm or our rigorous analysis, but enable faster convergence in practice.

In this paper, we study heterogeneous systems with $k = 2$ color classes. As discussed in Section 5, our algorithm performs well in practice for larger values of k and we expect our proof techniques would generalize without needing significant new ideas. However, this generalization would be cumbersome; thus, for simplicity, we restrict our attention to systems with colors $\{c_1, c_2\}$. For 2-heterogeneous systems, we can formally define separation with respect to having large monochromatic regions.

▶ **Definition 3.** *For $\beta > 0$ and $\delta \in (0, 1/2)$, a 2-heterogeneous particle system configuration σ is said to be* (β, δ)-separated *if there is a subset of particles R such that:*

1. *There are at most $\beta\sqrt{n}$ edges of σ with exactly one endpoint in R;*
2. *The density of particles of color c_1 in R is at least $1 - \delta$; and*
3. *The density of particles of color c_1 not in R is at most δ.*

Unpacking this definition, β controls how small a boundary there is between the monochromatic region R and the rest of the system, with smaller β requiring smaller boundaries. The δ parameter expresses the tolerance for having particles of the wrong color within the monochromatic region R: small values of δ require stricter separation of the color classes, while larger values of δ allow for more integrated configurations. Notably, R does not need to be connected.

2.4 Markov Chains

A thorough treatment of Markov chains can be found in the standard textbook [22]. A *Markov chain* is a memoryless random process on a state space Ω; for our purposes, Ω is finite and discrete. We focus on discrete time Markov chains, where one transition occurs

[2] In domains where physical swap moves are unrealistic, colors could be treated as in-memory attributes that could be exchanged by neighboring particles to simulate a swap move.

per *iteration* (or *step*). Because of its stochasticity, we can completely describe a Markov chain by its transition matrix M, which is an $|\Omega| \times |\Omega|$ matrix where for $x, y \in \Omega$, $M(x, y)$ is the probability, if in state x, of transitioning to state y in one step. The t-step transition probability $M^t(x, y)$ is the probability of transitioning from x to y in exactly t steps.

A Markov chain is *ergodic* if it is both *irreducible* (i.e., for all $x, y \in \Omega$ there is a t such that $M^t(x, y) > 0$) and *aperiodic* (i.e., for all $x \in \Omega$, $\gcd\{t : M^t(x, x) > 0\} = 1$) . A *stationary distribution* of a Markov chain is a probability distribution π over Ω such that $\pi M = \pi$. Any finite, ergodic Markov chain converges to a unique stationary distribution given by $\pi(y) = \lim_{t\to\infty} M^t(x, y)$ for any $x, y \in \Omega$; importantly, for such chains this distribution is independent of starting state x. To verify π' is the unique stationary distribution of a finite ergodic Markov chain, it suffices to check that $\pi'(x)M(x, y) = \pi'(y)M(y, x)$ for all $x, y \in \Omega$ (the *detailed balance condition*; see, e.g., [11]).

Given a state space Ω, a set of allowable transitions between states, and a desired stationary distribution π on Ω, the Metropolis-Hastings algorithm [13] gives a Markov chain on Ω with those transitions that converges to π. For separation, the state space contains particle configurations and transitions correspond to configurations that differ by one particle move; the stationary distribution π favors well-separated configurations; and we calculate transition probabilities according to the Metropolis-Hasting algorithm (using a *Metropolis filter*). Importantly, we choose π so that these transition probabilities can be calculated by an individual particle using only information in its local neighborhood.

3 The Separation Algorithm

We now present our stochastic, local, distributed algorithm for separation. Our algorithm achieves separation by biasing particles towards moves that both gain them more neighbors overall and more like-colored neighbors. We use two bias parameters to control this preference: $\lambda > 1$ corresponds to particles favoring having more neighbors, and $\gamma > 1$ corresponds to particles favoring having more neighbors of their own color.

In order to leverage powerful techniques from Markov chain analysis and statistical physics to prove the correctness of our algorithm, we design our algorithm to follow certain invariants. First, assuming the initial particle system configuration is connected, our algorithm ensures it remains connected; this is necessary because particles have strictly local communication abilities so a disconnected particle is unable to communicate with or even find the rest of the particles. Second, our algorithm eventually eliminates all holes in the configuration, and no new holes are ever formed. This is necessary because our proof techniques only apply to hole-free configurations. Third, once all holes have been eliminated, all moves allowed by our algorithm are *reversible*: if a particle moves from node u to an adjacent node v in one step, there is a nonzero probability that it moves back to u in the next step. Finally, the moves allowed by our algorithm suffice to transform any connected, hole-free configuration into any other connected, hole-free configuration.

Our algorithm uses two locally-checkable properties that ensure particles do not disconnect the system or form a hole when moving (our first two invariants). We use the following notation. For a location ℓ – i.e., a node of the triangular lattice G_Δ – let $N_i(\ell)$ denote the particles of color c_i occupying locations adjacent to ℓ. For neighboring locations ℓ and ℓ', let $N_i(\ell \cup \ell')$ denote the set $N_i(\ell) \cup N_i(\ell')$, excluding particles occupying ℓ and ℓ'. When ignoring color, let $N(\ell) = \bigcup_i N_i(\ell)$; define $N(\ell \cup \ell')$ analogously. Let $\mathbb{S} = N(\ell) \cap N(\ell')$ denote the set of particles adjacent to both locations. A particle can move from location ℓ to ℓ' if one of the following are satisfied:

▶ **Property 4.** *$|\mathbb{S}| \in \{1, 2\}$ and every particle in $N(\ell \cup \ell')$ is connected to exactly one particle in $\mathbb{S}$ by a path through $N(\ell \cup \ell')$.*

▶ **Property 5.** *$|\mathbb{S}| = 0$, and both $N(\ell) \setminus \{\ell'\}$ and $N(\ell') \setminus \{\ell\}$ are nonempty and connected.*

Note these properties do not need to be verified for swap moves, since swap moves do not change the set of occupied locations and thus cannot disconnect the system or create a hole.

We now define the Markov chain $\mathcal{M}$ for separation. The state space Ω of $\mathcal{M}$ is the set of all connected heterogeneous particle system configurations of n contracted particles, and Algorithm 1 defines its transition probabilities. We note that $\mathcal{M}$, a centralized Markov chain, can be directly translated to a fully distributed, local, asynchronous algorithm $\mathcal{A}$ that can be run by each particle independently and concurrently to achieve the same system behavior. This translation is much the same as for previous algorithms developed using the stochastic approach to self-organizing particle systems [2, 6]; we refer the interested reader to those papers for details. Importantly, this translation is only possible because all probability calculations and property checks in $\mathcal{M}$ use strictly local information available to the particles involved. Simulations of $\mathcal{M}$ can be found in Section 3.2.

Algorithm 1 Markov Chain $\mathcal{M}$ for Separation and Integration.

Beginning at any connected configuration σ_0 of n particles, repeat:

Choose a particle P uniformly at random; let c_i be its color and ℓ its location.
Choose a neighboring location ℓ' and $q \in (0, 1)$ each uniformly at random.
if ℓ' is unoccupied **then**
    P expands to occupy both ℓ and ℓ'.
    Let $e = |N(\ell)|$ (resp., $e_i = |N_i(\ell)|$) be the number of neighbors (resp., of color c_i) P had when contracted at location ℓ, and define $e' = |N(\ell')|$ and $e'_i = |N_i(\ell')|$ analogously.
    if (i) $e \neq 5$, (ii) ℓ and ℓ' satisfy Property 4 or 5, and (iii) $q < \lambda^{e'-e} \cdot \gamma^{e'_i - e_i}$ **then**
        P contracts to ℓ'.
    else P contracts back to ℓ.
else if ℓ' is occupied by particle Q of color c_j **then**
    if $q < \gamma^{|N_i(\ell') \setminus \{P\}| - |N_i(\ell)| + |N_j(\ell) \setminus \{Q\}| - |N_j(\ell')|}$ **then** P and Q perform a swap move.

3.1 The Stationary Distribution of Markov Chain $\mathcal{M}$

In this section, we prove that Markov chain $\mathcal{M}$ maintains the four invariants described previously and then characterize its stationary distribution.

▶ **Lemma 6.** *If the particle system is initially connected, it remains connected throughout the execution of $\mathcal{M}$. Moreover, $\mathcal{M}$ eventually eliminates any holes in the initial configuration, after which no holes are ever introduced again.*

Proof. This follows directly from analogous results for compression [6]. Although the separation and compression algorithms assign different probabilities to particle moves, the set of allowed movements is exactly the same, excluding swap moves that do not change the set of occupied nodes of G_Δ, so they cannot disconnect the system or introduce a hole. ◀

▶ **Lemma 7.** *Once all holes have been eliminated, every possible particle move is reversible; that is, if there is a positive probability of moving from configuration σ to configuration τ, then there is a positive probability of moving from τ to σ.*

Proof. Suppose, for example, that a particle P moves from location ℓ to ℓ'. In the next time step, it is possible for P to be chosen again (Step 1) and for ℓ to be chosen as the position to explore (Step 2). Because Properties 4 and 5 are symmetric with respect to ℓ and ℓ', whichever was satisfied in the forward move will also be satisfied in this reverse move. Finally, the probability checked in Condition (*iii*) of Step 7 is always nonzero, so all together there is a nonzero probability that P moves back to ℓ in this reverse move. Swap moves can be shown to be reversible in a similar way. ◀

▶ **Lemma 8.** *Markov chain $\mathcal{M}$ is ergodic on the space of connected, hole-free configurations.*

Proof Sketch. One can show that $\mathcal{M}$ is irreducible (i.e., the moves of $\mathcal{M}$ suffice to transform any configuration to any other configuration) similarly to the proof of the same fact for compression [6]: it is first shown that any configuration can be reconfigured into a straight line; then, the line can be sorted by the color of the particles; finally, by reversibility (Lemma 7), the line can be reconfigured into any configuration. Additionally, it is easy to see that $\mathcal{M}$ is aperiodic: at each iteration of $\mathcal{M}$, there is a nonzero probability that the configuration does not change. Thus, because $\mathcal{M}$ is irreducible and aperiodic, we conclude it is ergodic. ◀

Because $\mathcal{M}$ is finite and ergodic, it converges to a unique stationary distribution π that we now characterize. For a configuration σ, let $h(\sigma)$ be the number of heterogeneous edges in σ.

▶ **Lemma 9.** *For $Z = \sum_\sigma (\lambda\gamma)^{-p(\sigma)} \cdot \gamma^{-h(\sigma)}$, the stationary distribution of $\mathcal{M}$ is:*

$$\pi(\sigma) = \begin{cases} (\lambda\gamma)^{-p(\sigma)} \cdot \gamma^{-h(\sigma)}/Z & \text{if } \sigma \text{ is connected and hole-free;} \\ 0 & \text{otherwise.} \end{cases}$$

Proof Sketch. By Lemma 6, when $\mathcal{M}$ starts at a connected configuration it eventually reaches and remains in the set of configurations that are connected and hole-free. Thus, disconnected configurations and configurations with holes have zero weight at stationarity. In Appendix A.2, we show using detailed balance that the unique stationary distribution of $\mathcal{M}$ can be written, for σ connected and hole-free, as $\pi(\sigma) = \lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}/Z_e$ where $e(\sigma)$ is the number of edges and $a(\sigma)$ is the number of homogeneous edges of σ and $Z_e = \sum_\sigma \lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}$. This can be rewritten as in the lemma using two facts: (*i*) since every edge is either homogeneous or heterogeneous, $e(\sigma) = a(\sigma) + h(\sigma)$; and (*ii*) for any connected, hole-free configuration σ, $e(\sigma) = 3n - p(\sigma) - 3$, a result shown in [6]. ◀

The remainder of this paper will be spent analyzing this stationary distribution.

3.2 Simulations

We supplement our rigorous results with simulations that show separation occurs for even better values of λ and γ than our proofs guarantee, indicating that our proven bounds are likely not tight. We simulated $\mathcal{M}$ on heterogeneous particle systems with two colors, using 50 particles of each color. Figure 2 shows the progression of $\mathcal{M}$ over time with bias parameters $\lambda = 4$ and $\gamma = 4$, the regime in which particles prefer to have more neighbors, especially those of their own color. The simulation ran for nearly 70 million iterations, but much of the system's compression and separation occurs in the first million iterations. Separation still occurs even when swap moves are disallowed, but takes much longer to achieve.

Figure 3 compares the resulting system configurations after running $\mathcal{M}$ from the same initial configuration for the same number of iterations, varying only the values of λ and γ. We observe four distinct phases: compressed-separated, compressed-integrated, expanded-separated, and expanded-integrated. We rigorously verify the compressed-separated behavior

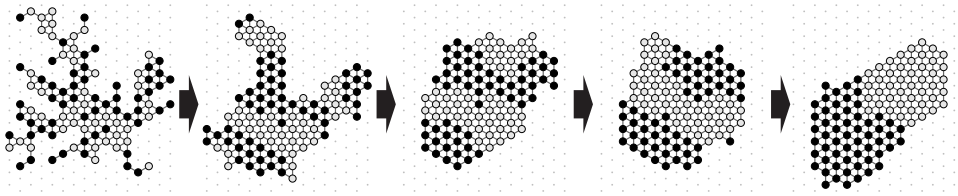


Figure 2 A 2-heterogeneous particle system of 100 particles starting from an arbitrary initial configuration after (from left to right) 0; 50,000; 1,050,000; 17,050,000; and 68,250,000; iterations of $\mathcal{M}$ with $\lambda = 4$ and $\gamma = 4$.

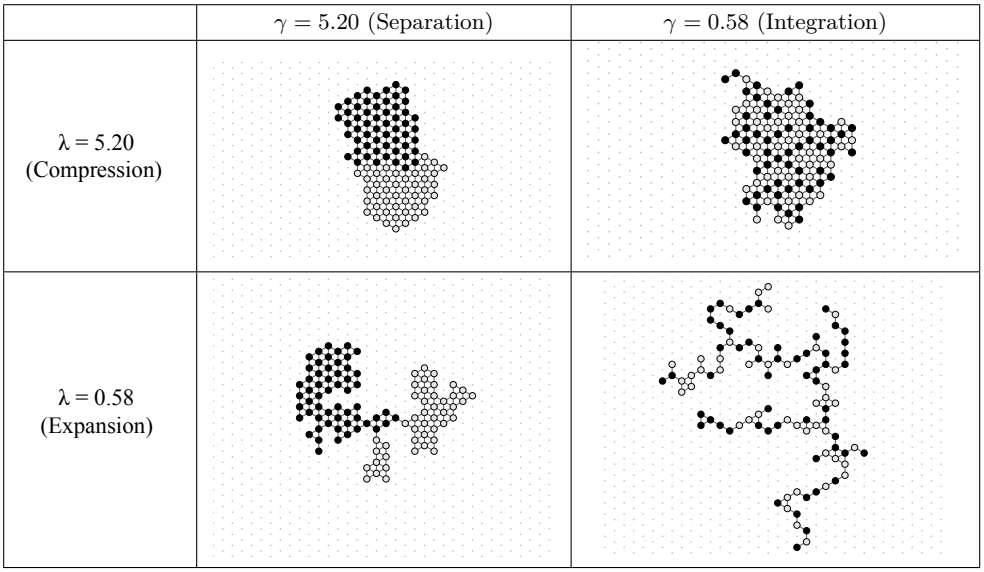


Figure 3 A 2-heterogeneous particle system of 100 particles starting in the leftmost configuration of Figure 2 after 50,000,000 iterations of $\mathcal{M}$ for various values of the parameters λ and γ.

(i.e., when λ and γ are large), and do the same for the compressed-integrated behavior (i.e., when λ is large and γ is small). We do not give proofs for expanded configurations; in fact, our current definition of separation may not accurately capture what occurs in expanded configurations.

4 Summary of Results and Proofs

Here we summarize our results and proofs; details have been omitted due to length constraints.

We want to know for which values of λ and γ separation does or does not occur. Our proof techniques only apply to compressed configurations, so we must first show that Markov chain $\mathcal{M}$ achieves compression for the values of λ and γ we are interested in. Previous proofs of compression in homogeneous particle systems break down for heterogeneous systems, so we utilize the *cluster expansion* to overcome this obstacle. The cluster expansion comes from statistical physics and allows us to rewrite a sum over collections of disjoint objects in terms of a sum over collections of overlapping objects. This latter sum is often much easier to work with. For the cluster expansion to be useful, the formal power series it involves must be convergent. We highly recommend Chapter 5 of [12] to learn more about the cluster expansion. Here we present only the relevant definitions and results from this chapter.

In a *polymer model*, we consider a finite set Γ, the elements of which are called *polymers*. We will consider polymers that are collections of edges of G_Δ having certain properties; for large γ, our polymers are minimal cut sets that we call *loops*, and when γ is close to one, our polymers are connected edge sets with an even number of edges incident on each vertex. Formally, polymers only need to satisfy:

- Each polymer $\xi \in \Gamma$ has a real *weight* $w(\xi)$.[3]
- There is a notion of pairwise *compatibility* for polymers.

Polymers are typically compatible when they are well-separated in some sense. Our loop polymers will be compatible when they share no edges, and our even polymers will be compatible when they are not incident on any of the same vertices. We say a collection of polymers $\Gamma' \subseteq \Gamma$ is *compatible* if all polymers in Γ' are pairwise compatible.

The *polymer partition function* is defined as:

$$\Xi = \sum_{\substack{\Gamma' \subseteq \Gamma \\ compatible}} \prod_{\xi \in \Gamma'} w(\xi).$$

Many partition functions of spin systems, such as the Ising model or the hard-core lattice gas model, can be written in this form as polymer partition functions. Such an abstract sum can sometimes be hard to analyze, but the *cluster expansion* gives a way of rewriting this expression in terms of a sum over subsets $\Gamma' \subseteq \Gamma$ where many polymers are incompatible; because incompatible polymers "touch", we can enumerate such collections more easily and thus such sums are often easier to work with

Formally, consider an ordered multiset $X = \{\xi_1, \xi_2, \ldots, \xi_m\} \subseteq \Gamma$. Let H_X be the *incompatibility graph* on vertex set $\{1, 2, \ldots, m\}$ where $i \sim j$ whenever ξ_i and ξ_j are incompatible. We say that the X is a *cluster* if H_X is connected.[4] Let $|X| = m$ denote the number of polymers in cluster X (with polymers counted with the appropriate multiplicities).

The *cluster expansion* is the formal power series for $\ln \Xi$ given in Equation 2. Often this power series does not converge, but the *Kotecky-Preiss condition* guarantees convergence and is often easy to verify [21]. The following theorem states the Kotecky-Preiss condition (Equation 1) and the cluster expansion of Ξ.

▶ **Theorem 10** ([12], Chapter 5). *Let Γ be a finite set of polymers ξ with real weights $w(\xi)$ and a notion of pairwise compatibility. If there exists a function $a : \Gamma \to \mathbb{R}_{>0}$ such that for all $\xi^* \in \Gamma$,*

$$\sum_{\substack{\xi \in \Gamma: \\ \xi, \xi^* \ incompatible}} |w(\xi)| e^{a(\xi)} \leq a(\xi^*), \tag{1}$$

then the polymer partition function Ξ satisfies

$$\ln \Xi = \sum_{X:\, cluster} \frac{1}{|X|!} \left(\sum_{\substack{G \subseteq H_X: \\ connected, \\ spanning}} (-1)^{|E(G)|} \right) \left(\prod_{\xi \in X} w(\xi) \right), \tag{2}$$

where $G \subseteq H_X$ means G is a subgraph of H_X.

[3] In general $w(\xi)$ can be complex, but for our purposes it will always be a (positive or negative) real number.

[4] Many sources define clusters to be unordered multisets, necessitating additional combinatorial terms in the cluster expansion; for simplicity, we assume clusters are ordered.

The cluster expansion is derived and this theorem is proved in Chapter 5 of [12], for a slightly different (but equivalent) definition of a cluster.

We apply the cluster expansion twice, with two different notions of polymers and compatibility. In both cases, our polymers will be connected edge sets $\xi \subseteq E(G_\Delta)$, and we use that to state a general result here. Let Γ be an infinite set of such polymers that is invariant under translation and rotation of polymers. Two polymers in Γ will be compatible if they are well-separated in the model-dependent sense described above. Polymers are incompatible when they are "too close"; for a polymer $\xi \in \Gamma$, let $[\xi] \subseteq E(G_\Delta)$ be the the minimal edge set such that if ξ' is not compatible with ξ, then ξ' must contain an edge of $[\xi]$. We use brackets, consistent with the notation of [12], because this is a type of *closure* of a polymer. For our loop polymers, which are compatible if they share no edges, $[\xi] = \xi$. For our even polymers, which are compatible if they are not incident on any of the same vertices, $[\xi]$ is all edges that share an endpoint with an edges of ξ. We denote the size of this edge set as $|[\xi]|$.

We will be interested in some finite region $\Lambda \subseteq E(G_\Delta)$, and we say $\Gamma_\Lambda \subseteq \Gamma$ is all polymers of Γ whose edges are contained in Λ. Let $\partial\Lambda$ be an edge set such that a cluster containing an edge in Λ and an edge not in Λ must contain an edge of $\partial\Lambda$. We will consider loop polymers with edges from $E^{int}_{\mathcal{P}}$, the set of edges with at least one endpoint strictly inside boundary $\mathcal{P}$, so in this case we use $\Lambda = E^{int}_{\mathcal{P}}$ and $\partial\Lambda$ the edges in $\mathcal{P}$. For even polymers, we use $\Lambda = E_{\mathcal{P}}$, all edges on or inside $\mathcal{P}$, and $\partial\Lambda$ is all edges with one endpoint on $\mathcal{P}$ and the other outside.

The following states the key fact about the cluster expansion that we will need. Namely, when a certain mild condition is satisfied, we can use the cluster expansion to give upper and lower bounds on the polymer partition function for Λ in terms of a volume term, depending only on $|\Lambda|$, and a surface term, depending only on $|\partial\Lambda|$.

▶ **Theorem 11.** *Let Γ be an infinite set of polymers $\xi \subseteq E(G_\Delta)$ that is closed under translation and rotation, and let $\Lambda \subseteq E(G_\Delta)$ be finite. If there is a constant c such that for any edge $e \in E(G_\Delta)$,*

$$\sum_{\substack{\xi \in \Gamma: \\ e \in \xi}} |w(\xi)| e^{c|[\xi]|} \leq c,$$

then for any Λ the partition function

$$\Xi_\Lambda := \sum_{\substack{\Gamma' \subseteq \Gamma_\Lambda \\ compatible}} \prod_{\xi \in \Gamma'} w(\xi)$$

satisfies

$$e^{\psi|\Lambda| - c|\partial\Lambda|} \leq \Xi_\Lambda \leq e^{\psi|\Lambda| + c|\partial\Lambda|},$$

for some constant $\psi \in [-c, c]$ that is independent of Λ.

We prove this theorem in Appendix A.3.

This result is the key step needed to show that when λ and γ are both large, compression occurs; as our techniques for establishing separation first require configurations to be compressed, this is a necessary first step. For compression, we look at the *partition function* $Z_{\mathcal{P}}$ for different fixed boundaries $\mathcal{P}$, where $Z_{\mathcal{P}}$ is the sum over all configurations σ with boundary $\mathcal{P}$ of their weights $(\lambda\gamma)^{-|\mathcal{P}|} \cdot \gamma^{-h(\sigma)}$. We cannot analyze $Z_{\mathcal{P}}$ directly, so we instead relate $Z_{\mathcal{P}}$ to a specific polymer partition function $\Xi^{\mathcal{L}}_{\mathcal{P}}$ which does have a cluster expansion. Using the sufficient condition of Theorem 10, we show the cluster expansion for $\Xi^{\mathcal{L}}_{\mathcal{P}}$ is convergent

when $\gamma > 4^{5/4}$. We then use this expression of $\ln \Xi_{\mathcal{P}}^{\mathcal{L}}$ as a convergent power series and Theorem 11 to bound $\Xi_{\mathcal{P}}^{\mathcal{L}}$ in terms of a *volume term*, depending only on the number of particles n, and a *surface term*, depending only on $|\mathcal{P}|$, the length of boundary $\mathcal{P}$.

▶ **Lemma 12.** *When $\gamma > 4^{5/4}$, for $c = 0.0001$, there exists a constant $\psi \in [-c, c]$ that depends on γ but is independent of $\mathcal{P}$ such that*

$$e^{(3n-3)\psi - 3c|\mathcal{P}|} \leq \Xi_{\mathcal{P}}^{\mathcal{L}} \leq e^{(3n-3)\psi + 3c|\mathcal{P}|}.$$

This means, in particular, that the ratios of $\Xi_{\mathcal{P}}^{\mathcal{L}}$ and $\Xi_{\mathcal{P}'}^{\mathcal{L}}$ for different boundaries $\mathcal{P}$ and $\mathcal{P}'$ that enclose the same number n of particles can be bounded by an expression that is exponential in the lengths of these boundaries but independent of n. This is essential to our compression argument, which will focus on boundaries of various lengths. We note that it is straightforward, using the previous lemma, to get similar bounds on $Z_{\mathcal{P}}$, the quantity we are actually interested in. We use this to apply a Peierls argument similar to the one used to show compression in [6]. This argument relates the total weight of undesirable configurations – those with boundaries longer than $\alpha \cdot p_{min}$ for some constant $\alpha > 1$ – to the weight of configurations with minimum perimeter, p_{min}. The result is as follows.

▶ **Theorem 13.** *Consider algorithm $\mathcal{M}$ when there are n total particles of two different colors. For $c = 0.0001$, when constants $\alpha > 1$, $\lambda > 1$, and $\gamma > 4^{5/4}$ satisfy*

$$\frac{2(2+\sqrt{2})e^{3c}}{\lambda\gamma}\left(e^{3c}\lambda\gamma^{3/2}\right)^{1/\alpha} < 1,$$

when n is sufficiently large then for $\mathcal{M}$ with parameters λ and γ, configurations drawn from distribution π are α-compressed with probability at least $1 - \zeta^{\sqrt{n}}$ for some constant $\zeta < 1$.

One corollary is that if $\lambda > 1$ and $\gamma > 4^{5/4}$ such that $\lambda\gamma > 2(2+\sqrt{2})e^{0.0003} \sim 6.83$, there exists a constant α such that a configuration drawn from the stationary distribution π of $\mathcal{M}$ is α-compressed with high probability. (Recall, we say an event A occurs with high probability, or w.h.p., if $\Pr[A] \geq 1 - c^{n^{\delta}}$, where $0 < c < 1$ and $\delta > 0$ are constants. Unless we explicitly state otherwise, it will always be the case that $\delta = 1/2$.) Conversely, for any $\alpha > 1$, there exist λ and γ such that $\mathcal{M}$ with these parameter values achieves α-compression at stationarity w.h.p.

We next show, again when λ and γ are large enough, that separation provably occurs. By the previous theorem, it suffices to show this among compressed configurations. We use a technique known as *bridging* that was developed to analyze molecular mixtures called *colloids* [28]. Adapting the bridging approach to our setting required several new innovations to overcome obstacles such as the irregular shapes of particle system configurations, the non-self-duality of the triangular lattice, the interchangeability between color classes, and other technicalities related to interfaces between particles of different colors. The main result of this section is the following theorem. Recall that for a fixed boundary $\mathcal{P}$, the probability distribution $\pi_{\mathcal{P}}$ is over colored particle configurations with this boundary where $\pi_{\mathcal{P}}(\sigma)$ is proportional to $\gamma^{-h(\sigma)}$.

▶ **Theorem 14.** *Let $\mathcal{P}$ be the boundary of n particles with $|\mathcal{P}| \leq \alpha p_{min}$. For any $\beta > 2\sqrt{3}\alpha$ and any $\delta < 1/2$, if γ is large enough that*

$$3^{\frac{2\alpha\sqrt{3}}{\beta}} 4^{\frac{1+3\delta}{4\delta}} \gamma^{-1+\frac{2\alpha\sqrt{3}}{\beta}} < 1$$

then for sufficiently large n a configuration drawn from $\pi_{\mathcal{P}}$ is (β, δ)-separated with probability at least $1 - \zeta^{\sqrt{n}}$ for some constant $\zeta < 1$.

Combining this with the previous theorem, we see that for any $\lambda > 1$ and $\gamma > 4^{5/4} \sim 5.66$ such that $\lambda\gamma > 2(2+\sqrt{2})e^{0.0003} \sim 6.83$, there exist constants β and δ such that for large enough n, $\mathcal{M}$ provably achieves (β, δ)-separation at stationarity w.h.p. Furthermore, for any $\beta > 2\sqrt{3}$ and any $\delta < 1/2$, there are values for λ and γ such that for large enough n, $\mathcal{M}$ provably achieves (β, δ)-separation at stationarity w.h.p.

We are also able to show that there are some values of γ close to one for which separation does not occur. This counterintuitively includes values where $\gamma > 1$ and particles have a preference for being next to particles of the same color. As we did for large values of γ, we first show that when λ is large and γ is close to one, compression provably occurs. The polymer partition function $\Xi_{\mathcal{P}}^{\mathcal{L}}$ from above does not have a convergent cluster expansion when γ is close to one, so we cannot use it to show compression. Instead, we carefully relate $Z_{\mathcal{P}}$ to a different polymer partition function $\Xi_{\mathcal{P}}^{HT}$ by considering the *high temperature expansion*, which rewrites a sum over configurations with a fixed boundary as a sum over even edge sets within that boundary. The high-temperature expansion is well-studied for the Ising model (see, e.g., [12], Section 3.7.3). We show $\Xi_{\mathcal{P}}^{HT}$ has a convergent cluster expansion when γ is close to one. We then use the cluster expansion for this high temperature representation, much the same as above, to show compression provably occurs.

▶ **Theorem 15.** *Consider algorithm $\mathcal{M}$ when there are n total particles of two different colors. For $a = 10^{-5}$, when constants $\alpha > 1$, $\lambda > 1$, and $\gamma \in (79/81, 81/79)$ satisfy*

$$\frac{2(2+\sqrt{2})e^{3a}}{\lambda(\gamma+1)}\left(\frac{\lambda(\gamma+1)}{2e^{-3a}\left(\frac{79}{81}\right)}\right)^{1/\alpha} < 1$$

when n is sufficiently large then for $\mathcal{M}$ with parameters λ and γ, configurations drawn from $\mathcal{M}$'s stationary distribution π are α-compressed with probability at least $1 - \zeta^{\sqrt{n}}$ for some constant $\zeta < 1$.

This theorem implies that for any $\lambda > 1$ and $\gamma \in (79/81, 81/79)$ such that $\lambda(\gamma+1) > 2(2+\sqrt{2})e^{0.00003} \sim 6.83$, there exists a constant α such that a configuration drawn from the stationary distribution π of $\mathcal{M}$ is α-compressed w.h.p. Conversely, for any $\alpha > 1$ and any $\gamma \in (79/81, 81/79)$, for large enough λ algorithm $\mathcal{M}$ with parameters λ and γ achieves α-compression at stationarity w.h.p.

Once we have shown that compression occurs for large λ and γ near one, we show that among these compressed configurations a large amount of separation between color classes is very unlikely. We prove this with a probabilistic argument in which we find a set of polynomially many events such that if separation occurs, then at least one of these events occurs. We then show that each event occurs with probability at most $\zeta^{n^{1/2-\varepsilon}}$ for some $\zeta < 1$ and arbitrarily small $\varepsilon > 0$, which via a union bound over the polynomial number of events implies separation is very unlikely.

▶ **Theorem 16.** *Let $\mathcal{P}$ be any α-compressed boundary. Let $\delta < 1/4$ and γ close enough to one such that there exists a $\mu \in (\delta/(1-2\delta), 1/2)$ where*

$$\left(\frac{\mu}{1-\mu}\right)^{(\mu-\delta/(1-2\delta))/11} < \gamma < \left(\frac{1-\mu}{\mu}\right)^{(\mu-\delta/(1-2\delta))/11}.$$

For any β and any $c < 1/4$, there is a constant $\zeta < 1$ such that the probability a particle configuration drawn at random from $\pi_{\mathcal{P}}$ is (β, δ)-separated is at most $\zeta^{n^{2c}}$.

Combining this with the results above, we see that for $\lambda > 1$ and $\gamma \in (79/81, 81/79)$ such that $\lambda(\gamma+1) > 2(2+\sqrt{2})e^{0.00003} \sim 6.83$, there are constants β and δ such that the probability $\mathcal{M}$ with parameters λ and γ achieves (β, δ)-separation at stationarity is at most $\zeta^{n^{1/2-\varepsilon}}$,

where $\varepsilon > 0$ and $\zeta < 1$. Conversely, for any $\beta > 0$ and any $\delta < 1/4$, there exists λ and γ such that $\mathcal{M}$ with these parameters achieves (β, δ)-separation at stationarity with probability at most $\zeta^{n^{1/2-\varepsilon}}$ for $\varepsilon > 0$ and $\zeta < 1$.

5 Conclusion

We considered separation with two colors, but expect our proofs to generalize in a straightforward way to heterogeneous systems with more colors using insights that generalize cluster expansion polymers from the Ising model to the Potts model (see the notion of a *contour* in Pirogov-Sinai theory, e.g., in Chapter 7 of [12]). The proofs would follow the same strategy for two colors, requiring little additional insight but a fairly large amount of technical detail.

We note that, as with previous papers using stochastic, distributed algorithms for programmable matter, we are unable to give any nontrivial bounds on the mixing time of our Markov chain $\mathcal{M}$. The difficulties in proving polynomial upper bounds on the mixing time are unsurprising, given similarities between $\mathcal{M}$ and a well-studied open problem in statistical physics about the mixing time of Glauber dynamics of the Ising model on $\mathbb{Z}^2$ with plus boundary conditions starting from the all minus state [24, 26] (see remarks concluding [6]). However, the mixing time may not be the best bound for characterizing when compression and separation occur. Simulations show that both compression and separation occur fairly quickly (Figure 2), although the algorithm continues to gradually achieve more compression and separation, confirming we likely achieve these goals well before converging to stationarity.

We believe the stochastic approach to self-organizing particle systems, used here to develop a distributed algorithm for separation and integration in programmable matter, is much more broadly applicable. This approach can potentially be applied to any objective described by a global energy function (where the desirable configurations have low energy values), provided changes in energy due to particle movements can be calculated with only local information. Choosing the correct global energy function is the key; translating the energy function into a Markov chain and then into a distributed algorithm is, by now, fairly routine (see [2, 6]). However, proving that the stationary distribution has our desired properties with high probability remains challenging, requiring application-specific proof techniques.

Last, we believe the proof techniques developed here extend beyond our current work. For separation and integration, the key ingredient is the cluster expansion, used recently to develop efficient low-temperature approximations and sampling algorithms, and the related Pirogov-Sinai theory, used to show slow mixing of certain Markov chains. Here, however, we used a completely different aspect of the cluster expansion by separating partition functions into surface and volume terms. The cluster expansion and Pirogov-Sinai theory have been widely used in statistical physics for many purposes, and we believe there are many more ways a thorough understanding of these methods can benefit computer science.

References

1 Leonard M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266(5187):1021–1024, 1994.

2 Marta Andrés Arroyo, Sarah Cannon, Joshua J. Daymude, Dana Randall, and Andréa W. Richa. A Stochastic Approach to Shortcut Bridging in Programmable Matter. *Natural Computing*, 17(4):723–741, 2018.

3 Alexander I. Barvinok. *Combinatorics and complexity of partition functions*, volume 30 of *Algorithms and Combinatorics*. Springer International Publishing, 2016.

4 Alexander I. Barvinok and Pablo Soberón. Computing the partition function for graph homomorphisms with multiplicities. *Journal of Combinatorial Theory, Series A*, 137:1–26, 2016.

5 Prateek Bhakta, Sarah Miracle, and Dana Randall. Clustering and mixing times for segregation models on $\mathbb{Z}^2$. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 327–340, 2014.

6 Sarah Cannon, Joshua J. Daymude, Dana Randall, and Andréa W. Richa. A Markov chain algorithm for compression in self-organizing particle systems. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, PODC '16, pages 279–288, Chicago, IL, USA, 2016. ACM. A significantly updated version is available at `arXiv:1603.07991`.

7 David Correa, Athina Papadopoulou, Christophe Guberan, Nynika Jhaveri, Steffen Reichert, Achim Menges, and Skylar Tibbits. 3D-Printed Wood: Programming Hygroscopic Material Transformations. *3D Printing and Additive Manufacturing*, 2(3):106–116, 2015.

8 Joshua J. Daymude, Kristian Hinnenthal, Andréa W. Richa, and Christian Scheideler. Computing by Programmable Particles. In *Distributed Computing by Mobile Entities: Current Research in Moving and Computing*, pages 615–681. Springer, Cham, 2019.

9 Zahra Derakhshandeh, Shlomi Dolev, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Brief announcement: amoebot - a new model for programmable matter. In *Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '14, pages 220–222, New York, NY, USA, 2014. ACM.

10 Moon Duchin and Bridget E. Tenner. Discrete geometry for electoral geography. Preprint available online at `arXiv:1808.05860`, 2018.

11 William Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, New York, 1968.

12 Sacha Friedli and Yvan Velenik. *Statistical Mechanics of Lattice Systems: A Concrete Mathematical Introduction*. Cambridge University Press, Cambridge, 2018.

13 Wilfred K. Hastings. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97–109, 1970.

14 Tyler Helmuth, Will Perkins, and Guus Regts. Algorithmic Pirogov-Sinai Theory. In *Proceedings of the 51st ACM Symposium on Theory of Computing*, STOC '19. ACM, 2019.

15 Gregory Herschlag, Han Sung Kang, Justin Luo, Christy V. Graves, Sachet Bangia, Robert Ravier, and Jonathan C. Mattingly. Quantifying Gerrymandering in North Carolina. Preprint available online at `arXiv:1801.03783`, 2018.

16 Michael A. Hogg and John C. Turner. Interpersonal attraction, social identification and psychological group formation. *European Journal of Social Psychology*, 15(1):51–66, 1985.

17 Nicole Immorlica, Robert Kleinberg, Brendan Lucier, and Morteza Zadomighaddam. Exponential Segregation in a Two-dimensional Schelling Model with Tolerant Individuals. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 984–993, 2017.

18 Ernst Ising. Beitrag zur theorie des ferromagnetismus [Contribution to the Theory of Ferromagnetism]. *Zeitschrift für Physik*, 31(1):253–258, 1925.

19 Matthew Jenssen, Peter Keevash, and Will Perkins. Algorithms for #BIS-hard problems on expander graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '19, pages 2235–2247, 2019.

20 Brian R. Johnson, Ellen van Wilgenburg, and Neil D. Tsutsui. Nestmate recognition in social insects: overcoming physiological constraints with collective decision making. *Behavioral Ecology and Sociobiology*, 65(5):935–944, 2011.

21 Roman Kotecký and David Preiss. Cluster Expansion for Abstract Polymer Models. *Communications in Mathematical Physics*, 103:491–498, 1986.

22 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, Providence, RI, USA, 2009.

23 Chao Liao, Jiabao Lin, Pinyan Lu, and Zhenyu Mao. Counting independent sets and colorings on random regular bipartite graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, APPROX/RANDOM 2019, 2019.

24 Eyal Lubetzky, Fabio Martinelli, Alan Sly, and Fabio Lucio Toninelli. Quasi-polynomial mixing of the 2D stochastic Ising model with "plus" boundary up to criticality. *Journal of the European Mathematical Society (JEMS)*, 15(2):339—-386, 2013.

25 Nancy Lynch. *Distributed Algorithms*. Morgan Kauffman, San Francisco, CA, USA, 1996.

26 Fabio Martinelli and Fabio Lucio Toninelli. On the Mixing Time of the 2D Stochastic Ising Model with "Plus" Boundary Conditions at Low Temperature. *Communications in Mathematical Physics*, 296(1):175–213, 2010.

27 Joseph E. Mayer. The Statistical Mechanics of Condensing Systems. I. *The Journal of Chemical Physics*, 5:67–73, 1937.

28 Sarah Miracle, Dana Randall, and Amanda Pascoe Streib. Clustering in Interfering Binary Mixtures. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, APPROX/RANDOM 2011, pages 652–663, 2011.

29 Hamed Omidvar and Massimo Franceschetti. Self-organized Segregation on the Grid. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, PODC '17, pages 401–410, New York, NY, USA, 2017. ACM.

30 T'ai H. Roulston, Grzegorz Buczkowski, and Jules Silverman. Nestmate discrimination in ants: effect of bioassay on aggressive behavior. *Insectes Sociaux*, 50(2):151–159, 2003.

31 Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014.

32 William Savoie, Sarah Cannon, Joshua J. Daymude, Ross Warkentin, Shengkai Li, Andréa W. Richa, Dana Randall, and Daniel I. Goldman. Phototactic Supersmarticles. *Artificial Life and Robotics*, 23(4):459–468, 2018.

33 Thomas C. Schelling. Models of Segregation. *The American Economic Review*, 59(2):488–493, 1969.

34 Thomas C. Schelling. Dynamic models of segregation. *The Journal of Mathematical Sociology*, 1(2):143–186, 1971.

35 Philip S. Stewart and Michael J. Franklin. Physiological heterogeneity in biofilms. *Nature Reviews Microbiology*, 6:199–210, 2008.

36 Rohan Thakker, Ajinkya Kamat, Sachin Bharambe, Shital Chiddarwar, and K. M. Bhurchandi. ReBiS - reconfigurable bipedal snake robot. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 309–314, 2014.

37 John C. Turner. Towards a cognitive redefinition of the social group. *Cahiers de Psychologie Cognitive/Current Psychology of Cognition*, 1(2):93–118, 1981.

38 Dejan Vinković and Alan Kirman. A physical analogue of the Schelling model. *Proceedings of the National Academy of Sciences*, 103(51):19261–19265, 2006.

39 Guopeng Wei, Connor Walsh, Irina Cazan, and Radu Marculescu. Molecular Tweeting: Unveiling the Social Network Behind Heterogeneous Bacteria Populations. In *Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics*, BCB '15, pages 366–375, New York, NY, USA, 2015. ACM.

A Appendix

Here we include the proofs of some of our claims that were omitted from the main body of this paper for conciseness and clarity. We do not include any detailed proofs of our technical results due to length constraints.

A.1 Proof of Lemma 2

Recall that Lemma 2 states that for any $n \geq 1$, there is a connected, hole-free particle configuration of n particles with perimeter at most $2\sqrt{3}\sqrt{n}$. That is, $p_{min}(n) \leq 2\sqrt{3}\sqrt{n}$.

Proof. The lemma can easily be verified for $n \leq 6$. For $n \geq 7$, we begin with the case where $n = 3\ell^2 + 3\ell + 1$ for some integer $\ell \geq 1$. A regular hexagon with side length ℓ can be decomposed into six triangles, each with $\ell(\ell+1)/2$ particles, and a single center vertex, for $3\ell^2 + 3\ell + 1$ total particles; see Figure 4a. Such a hexagon has perimeter 6ℓ. We see that

$$p_{min}(3\ell^2 + 3\ell + 1) \leq 6\ell \leq 2\sqrt{3}\sqrt{3\ell(\ell+1)} \leq 2\sqrt{3}\sqrt{n-1} \leq 2\sqrt{3}\sqrt{n}.$$

Now we consider $n = 3\ell^2 + 3\ell + 1 + k$, for integers ℓ and k, where $k \in [1, 6\ell + 6)$. As $(3\ell^2 + 3\ell + 1) + 6\ell + 6 = 3(\ell+1)^2 + 3(\ell+1) + 1$, this covers all possible values of n. We construct a particle configuration on $n = 3\ell^2 + 3\ell + 1 + k$ particles by first constructing a regular hexagon of side length ℓ and then adding the remaining k particles around the outside of this hexagon in a single layer, completing one side before beginning the next; see Figure 4b, where $\ell = 3$ and $k = 6$. For $k \leq \ell$, the perimeter of this configuration is $6\ell + 1$. More generally, the perimeter increases by one when particles begin to be added to a new side of the hexagon, and so for $i = 2, 3, 4, 5, 6$, for $(i-1)\ell + (i-2) < k \leq i\ell + (i-1)$ the perimeter of this configuration is $6\ell + i$. We see that (using $i \leq 6$ and $\ell \geq 1$), for any $i = 1, 2, 3, 4, 5, 6$,

$$\begin{aligned} p_{min}(3\ell^2 + 3\ell + 1 + k) &\leq 6\ell + i \leq 2\sqrt{3}\sqrt{\left(\sqrt{3}\ell + \frac{i}{2\sqrt{3}}\right)^2} = 2\sqrt{3}\sqrt{3\ell^2 + \frac{i^2}{12} + i} \\ &\leq 2\sqrt{3}\sqrt{3\ell^2 + 3 + i} \\ &\leq 2\sqrt{3}\sqrt{3\ell^2 + 3\ell + 1 + i - 1} \\ &\leq 2\sqrt{3}\sqrt{3\ell^2 + 3\ell + 1 + k} = 2\sqrt{3}\sqrt{n}. \end{aligned}$$

This concludes our proof. ◀

A.2 Detailed Balance Proof that π is the Stationary Distribution of $\mathcal{M}$

Recall that Lemma 9 states that the stationary distribution of $\mathcal{M}$ is given by $\pi(\sigma) = 0$ if σ is disconnected or has holes, and by $\pi(\sigma) = (\lambda\gamma)^{-p(\sigma)} \cdot \gamma^{-h(\sigma)}/Z$ otherwise, where $Z = \sum_\sigma (\lambda\gamma)^{-p(\sigma)} \cdot \gamma^{-h(\sigma)}$. Here, we analyze the necessary cases to verify this with detailed balance.



(a) **(b)**

Figure 4 (a) The regular hexagon with side length $\ell = 3$ with $3\ell^2 + 3\ell + 1$ total particles. (b) A configuration with $n = 3\ell^2 + 3\ell + 1 + k$ particles for $\ell = 3$ and $k = 6$ with perimeter $20 < 2\sqrt{3}\sqrt{n}$.

Proof. We first verify that $\pi(\sigma) = \lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}/Z_e$ – where $e(\sigma)$ is the number of edges of σ, $a(\sigma)$ is the number of homogeneous edges of σ, and $Z_e = \sum_\sigma \lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}$ – is the stationary distribution by detailed balance. We then show that this form of π can be rewritten as in the lemma.

Consider any two connected, hole-free configurations σ, τ that differ by one move of some particle from location ℓ in σ to a neighboring location ℓ' in τ. By examining $\mathcal{M}$, we see that the probability of transitioning from σ to τ is:

$$M(\sigma,\tau) = \min\left\{1, \lambda^{|N(\ell')|-|N(\ell)|} \cdot \gamma^{|N_i(\ell')|-|N_i(\ell)|}\right\}/6n.$$

A similar analysis shows:

$$M(\tau,\sigma) = \min\left\{1, \lambda^{|N(\ell)|-|N(\ell')|} \cdot \gamma^{|N_i(\ell)|-|N_i(\ell')|}\right\}/6n.$$

Without loss of generality, suppose $\lambda^{|N(\ell')|-|N(\ell)|} \cdot \gamma^{|N_i(\ell')|-|N_i(\ell)|} < 1$, meaning $M(\sigma,\tau)$ is this value over $6n$ and $M(\tau,\sigma) = 1/6n$. Because the only edges that differ in σ and τ are incident to ℓ or ℓ',

$$\begin{aligned}
\pi(\sigma)M(\sigma,\tau) &= \frac{\lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}}{Z_e} \cdot \frac{1}{n} \cdot \frac{1}{6} \cdot \lambda^{|N(\ell')|-|N(\ell)|} \cdot \gamma^{|N_i(\ell')|-|N_i(\ell)|} \\
&= \frac{\lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}}{Z_e} \cdot \frac{1}{n} \cdot \frac{1}{6} \cdot \lambda^{e(\tau)-e(\sigma)} \cdot \gamma^{a(\tau)-a(\sigma)} \\
&= \frac{\lambda^{e(\tau)} \cdot \gamma^{a(\tau)}}{Z_e} \cdot \frac{1}{n} \cdot \frac{1}{6} \cdot 1 = \pi(\tau)M(\tau,\sigma)
\end{aligned}$$

Thus, detailed balance is satisfied for particle moves that are not swaps.

Suppose instead that σ and τ differ by a swap move of particle P with color c_i at location ℓ in σ and particle Q with color c_j at neighboring location ℓ' in σ. This move could occur if P or Q is chosen in Step 1 of $\mathcal{M}$, so:

$$M(\sigma,\tau) = \min\left\{1, \gamma^{|N_i(\ell')\setminus\{P\}|-|N_i(\ell)|+|N_j(\ell)\setminus\{Q\}|-|N_j(\ell')|}\right\}/3n.$$

Similarly, because τ has P at location ℓ' and Q at location ℓ, we have:

$$M(\tau,\sigma) = \min\left\{1, \gamma^{|N_i(\ell)\setminus\{P\}|-|N_i(\ell')|+|N_j(\ell')\setminus\{Q\}|-|N_j(\ell)|}\right\}/3n.$$

Without loss of generality, suppose that $\gamma^{|N_i(\ell')\setminus\{P\}|-|N_i(\ell)|+|N_j(\ell)\setminus\{Q\}|-|N_j(\ell')|} < 1$, so $M(\sigma,\tau)$ is this value over $3n$ and $M(\tau,\sigma) = 1/3n$. Then,

$$\begin{aligned}
\pi(\sigma)M(\sigma,\tau) &= \frac{\lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}}{Z_e} \cdot \frac{2}{n} \cdot \frac{1}{6} \cdot \gamma^{|N_i(\ell')\setminus\{P\}|-|N_i(\ell)|+|N_j(\ell)\setminus\{Q\}|-|N_j(\ell')|} \\
&= \frac{\lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}}{Z_e} \cdot \frac{2}{n} \cdot \frac{1}{6} \cdot \gamma^{(|N_i(\ell')\setminus\{P\}|+|N_j(\ell)\setminus\{Q\}|)-(|N_i(\ell)|+|N_j(\ell')|)} \\
&= \frac{\lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}}{Z_e} \cdot \frac{2}{n} \cdot \frac{1}{6} \cdot \gamma^{a(\tau)-a(\sigma)} \\
&= \frac{\lambda^{e(\tau)} \cdot \gamma^{a(\tau)}}{Z_e} \cdot \frac{2}{n} \cdot \frac{1}{6} \cdot 1 = \pi(\tau)M(\tau,\sigma)
\end{aligned}$$

In both cases, detailed balance is satisfied, so we conclude the stationary distribution π (which is only non-zero over connected, hole-free configurations) is given by $\pi(\sigma) = \lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}/Z_e$.

Since every edge of σ is either homogeneous or heterogeneous, we have $e(\sigma) = a(\sigma) + h(\sigma)$. From [6], we have $e(\sigma) = 3n - p(\sigma) - 3$, where n is the number of particles in the system. Thus, we can rewrite this unique stationary distribution as follows:

$$\begin{aligned}
\pi(\sigma) &= \frac{\lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}}{Z_e} \\
&= \frac{\lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}}{\sum_\sigma \lambda^{e(\sigma)} \cdot \gamma^{a(\sigma)}} \\
&= \frac{(\lambda\gamma)^{-3n+3} \cdot (\lambda\gamma)^{e(\sigma)} \cdot \gamma^{a(\sigma)-e(\sigma)}}{(\lambda\gamma)^{-3n+3} \cdot \sum_\sigma (\lambda\gamma)^{e(\sigma)} \cdot \gamma^{a(\sigma)-e(\sigma)}} \\
&= \frac{(\lambda\gamma)^{e(\sigma)-3n+3} \cdot \gamma^{a(\sigma)-e(\sigma)}}{\sum_\sigma (\lambda\gamma)^{e(\sigma)-3n+3} \cdot \gamma^{a(\sigma)-e(\sigma)}} \\
&= \frac{(\lambda\gamma)^{-p(\sigma)} \cdot \gamma^{-h(\sigma)}}{\sum_\sigma (\lambda\gamma)^{-p(\sigma)} \cdot \gamma^{-h(\sigma)}}.
\end{aligned}$$

This concludes our proof. ◀

A.3 Proof of Boundary-Volume Decomposition of Cluster Expansion

In this section we provide the proof of Theorem 11, which is our decomposition of a polymer partition function into boundary and volume terms via the cluster expansion. For the sake of clarity we restate this theorem here, including all of its hypotheses and assumptions.

▶ **Theorem 11.** *Let Γ be an infinite set of polymers $\xi \subseteq E(G_\Delta)$ that is closed under translation and rotation, and let $\Lambda \subseteq E(G_\Delta)$ be finite. If there is a constant c such that for any edge $e \in E(G_\Delta)$,*

$$\sum_{\substack{\xi \in \Gamma: \\ e \in \xi}} |w(\xi)| e^{c|[\xi]|} \leq c, \tag{3}$$

then for any Λ the partition function

$$\Xi_\Lambda := \sum_{\substack{\Gamma' \subseteq \Gamma_\Lambda \\ compatible}} \prod_{\xi \in \Gamma'} w(\xi)$$

satisfies

$$e^{\psi|\Lambda| - c|\partial\Lambda|} \leq \Xi_\Lambda \leq e^{\psi|\Lambda| + c|\partial\Lambda|},$$

for some constant $\psi \in [-c, c]$ that is independent of Λ.

Proof. We follow the same outline as the proof of the same fact for the Ising model in Section 5.7.1 of [12].

Let $\mathcal{X}$ be all clusters comprised of polymers from Γ, and let $\mathcal{X}_\Lambda$ be all clusters of polymers in Γ_Λ. Note that Equation 3 implies the hypothesis of Theorem 10 (Equation 1) is satisfied, with function $a : \Gamma \to \mathbb{R}$ given by $a(\xi) = c|[\xi]|$:

$$\sum_{\substack{\xi \in \Gamma: \\ \xi, \xi^* \ incompatible}} |w(\xi)| e^{a(\xi)} \leq \sum_{e \in [\xi^*]} \sum_{\substack{\xi \in \Gamma: \\ e \in \xi}} |w(\xi)| e^{c|[\xi]|} \leq c|[\xi^*]|.$$

Because this hypothesis is satisfied for all $\xi^* \in \Gamma$, it certainly holds when we restrict our attention to polymers in Γ_Λ. By Theorem 10, because Γ_Λ is a finite set, this means the cluster expansion for Ξ_Λ converges:

$$\ln \Xi_\Lambda = \sum_{X \in \mathcal{X}_\Lambda} \Psi(X)$$

Let $\overline{X} = \cup_{\xi \in X} \xi$ be the *support* of cluster X and $|\overline{X}|$ the size of this support. Using Equation 3 and standard techniques (see [12], the proof of Theorem 5.4 and Equation (5.29)), the translation and rotation invariance of Γ imply that for any edge $e \in E(G_\Delta)$,

$$\sum_{\substack{X \in \mathcal{X}: \\ e \in \overline{X}}} |\Psi(X)| \leq c. \tag{4}$$

The proof of this fact is the reason we need a slightly stronger hypothesis (Equation 3) than is needed to guarantee the cluster expansion converges (Equation 1).

For any cluster $X \in \mathcal{X}_\Lambda$, it trivially holds that $1 = (\sum_{e \in \Lambda} \mathbf{1}_{e \in \overline{X}})/\overline{X}$. We can use this fact to rewrite the cluster expansion for Ξ_Λ:

$$\begin{aligned}
\ln \Xi_\Lambda &= \sum_{X \in \mathcal{X}_\Lambda} \Psi(X) = \sum_{\substack{X \in \mathcal{X}: \\ \overline{X} \subseteq \Lambda}} \Psi(X) = \sum_{e \in \Lambda} \sum_{\substack{X \in \mathcal{X}: \\ e \in \overline{X}, \\ \overline{X} \subseteq \Lambda}} \frac{1}{|\overline{X}|} \Psi(X) \\
&= \sum_{e \in \Lambda} \left(\sum_{\substack{X \in \mathcal{X}: \\ e \in \overline{X}}} \frac{1}{|\overline{X}|} \Psi(X) - \sum_{\substack{X \in \mathcal{X}: \\ e \in \overline{X}, \\ \overline{X} \not\subseteq \Lambda}} \frac{1}{|\overline{X}|} \Psi(X) \right) \\
&= \left(\sum_{e \in \Lambda} \sum_{\substack{X \in \mathcal{X}: \\ e \in \overline{X}}} \frac{1}{|\overline{X}|} \Psi(X) \right) - \left(\sum_{e \in \Lambda} \sum_{\substack{X \in \mathcal{X}: \\ e \in \overline{X}, \\ \overline{X} \not\subseteq \Lambda}} \frac{1}{|\overline{X}|} \Psi(X) \right).
\end{aligned} \tag{5}$$

The two infinite sums in parentheses above are absolutely convergent by Equation 4, so this difference is well-defined.

To analyze the first term of Equation 5, we note that by the translation and rotation invariance of Γ, the sum

$$\psi := \sum_{\substack{X \in \mathcal{X}: \\ e \in \overline{X}}} \frac{1}{|\overline{X}|} \Psi(X)$$

is independent of e and of Λ and only depends on our particular polymer model; this is the value ψ that appears in the statement of the theorem, and by Equation 4, $|\psi| \leq c$. We conclude the first term of Equation 5 is $\psi|\Lambda|$.

To analyze the second term of Equation 5, recall if cluster X satisfies both $e \in \overline{X}$ for some $e \in \Lambda$ and $\overline{X} \not\subseteq \Lambda$, then $\overline{X}$ must contain some edge $f \in \partial\Lambda$. We rewrite the absolute value of this second sum as

$$\begin{aligned}\left|\sum_{e\in\Lambda}\sum_{\substack{X\in\mathcal{X}:\\ e\in\overline{X},\\ \overline{X}\not\subseteq\Lambda}}\frac{1}{|\overline{X}|}\Psi(X)\right| &\leq \sum_{e\in\Lambda}\sum_{\substack{X\in\mathcal{X}:\\ e\in\overline{X},\\ \overline{X}\not\subseteq\Lambda}}\frac{1}{|\overline{X}|}\,|\Psi(X)| \\ &\leq \sum_{f\in\partial\Lambda}\sum_{\substack{X\in\mathcal{X}:\\ f\in\overline{X}}}|\overline{X}\cap\Lambda|\frac{1}{|\overline{X}|}\,|\Psi(X)| \\ &\leq \sum_{f\in\partial\Lambda}\sum_{\substack{X\in\mathcal{X}:\\ f\in\overline{X}}}|\Psi(X)| \leq c\,|\partial\Lambda|\,.\end{aligned}$$

The last inequality above follows from Equation 4 and the translation and rotation invariance of Λ.

We conclude that Equation 5 implies

$$\psi|\Lambda| - c|\partial\Lambda| \leq \ln \Xi_\Lambda \leq \psi|\Lambda| + c|\partial\Lambda|.$$

Exponentiation proves the theorem. ◀

The Large-Error Approximate Degree of AC^0

Mark Bun
Boston University, Boston, MA, USA
http://cs-people.bu.edu/mbun/
mbun@bu.edu

Justin Thaler
Georgetown University, Washington, DC, USA
http://people.cs.georgetown.edu/jthaler/
justin.thaler@georgetown.edu

Abstract

We prove two new results about the inability of low-degree polynomials to uniformly approximate constant-depth circuits, even to slightly-better-than-trivial error. First, we prove a tight $\tilde{\Omega}(n^{1/2})$ lower bound on the threshold degree of the SURJECTIVITY function on n variables. This matches the best known threshold degree bound for any AC^0 function, previously exhibited by a much more complicated circuit of larger depth (Sherstov, FOCS 2015). Our result also extends to a $2^{\tilde{\Omega}(n^{1/2})}$ lower bound on the sign-rank of an AC^0 function, improving on the previous best bound of $2^{\Omega(n^{2/5})}$ (Bun and Thaler, ICALP 2016).

Second, for any $\delta > 0$, we exhibit a function $f\colon \{-1,1\}^n \to \{-1,1\}$ that is computed by a circuit of depth $O(1/\delta)$ and is hard to approximate by polynomials in the following sense: f cannot be uniformly approximated to error $\varepsilon = 1 - 2^{-\Omega(n^{1-\delta})}$, even by polynomials of degree $n^{1-\delta}$. Our recent prior work (Bun and Thaler, FOCS 2017) proved a similar lower bound, but which held only for error $\varepsilon = 1/3$.

Our result implies $2^{\Omega(n^{1-\delta})}$ lower bounds on the complexity of AC^0 under a variety of basic measures such as discrepancy, margin complexity, and threshold weight. This nearly matches the trivial upper bound of $2^{O(n)}$ that holds for every function. The previous best lower bound on AC^0 for these measures was $2^{\Omega(n^{1/2})}$ (Sherstov, FOCS 2015). Additional applications in learning theory, communication complexity, and cryptography are described.

2012 ACM Subject Classification Mathematics of computing → Approximation; Theory of computation → Communication complexity; Theory of computation → Circuit complexity

Keywords and phrases approximate degree, discrepancy, margin complexity, polynomial approximations, secret sharing, threshold circuits

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.55

Category RANDOM

Related Version The full version of this work appears at http://eccc.weizmann.ac.il/report/2018/143/.

Funding *Mark Bun*: This work was done while author was at Princeton University and the Simons Institute for the Theory of Computing, supported by a Google Research Fellowship.
Justin Thaler: Supported by NSF Grant CCF-1845125.

Acknowledgements The authors are grateful to Robin Kothari, Nikhil Mande, Jonathan Ullman, and the anonymous reviewers for valuable comments on earlier versions of this manuscript.

1 Introduction

The *threshold degree* of a Boolean function $f\colon \{-1,1\}^n \to \{-1,1\}$, denoted $\deg_{\pm}(f)$, is the least degree of a real polynomial p that sign-represents f, i.e., $p(x) \cdot f(x) > 0$ for all $x \in \{-1,1\}^n$. A closely related notion is the ε-approximate degree of f, denoted $\widetilde{\deg}_{\varepsilon}(f)$, which is the least degree of a real polynomial p such that $|p(x) - f(x)| \leq \varepsilon$ for all $x \in \{-1,1\}^n$.


© Mark Bun and Justin Thaler;
licensed under Creative Commons License CC-BY

The parameter setting $\varepsilon = 1$ is a degenerate case: $\widetilde{\deg}_1(f) = 0$ because the constant 0 function approximates any Boolean f to error $\varepsilon = 1$. However, as soon as ε is strictly less than 1, ε-approximate degree is a highly non-trivial notion with a rich mathematical theory. In particular, it is easily seen that

$$\deg_{\pm}(f) = \lim_{\varepsilon \nearrow 1} \widetilde{\deg}_{\varepsilon}(f).$$

In other words, threshold degree is equivalent to the notion of ε-approximate degree when ε is permitted to be *arbitrarily* close to (but strictly less than) 1.[1]

In this paper, we are concerned with proving ε-approximate degree lower bounds when either:

- ε is *arbitrarily* close to 1, or
- ε is *exponentially* close to 1 (i.e., $\varepsilon = 1 - 2^{-n^{1-\delta}}$ for some constant $\delta > 0$).

The former parameter regime captures threshold degree, while we refer to the latter as *large-error* approximate degree. While the approximate and threshold degree of a function f capture simple statements about its approximability by polynomials, these quantities relate intimately to the complexity of computing f in concrete computational models. Specifically, the query complexity models $\mathbf{UPP}^{\mathrm{dt}}$ and $\mathbf{PP}^{\mathrm{dt}}$, and the communication models $\mathbf{UPP}^{\mathrm{cc}}, \mathbf{PP}^{\mathrm{cc}}$, are all defined (cf. Section 2) as natural analogs of the Turing machine class $\mathbf{PP}$, which in turn captures probabilistic computation with arbitrarily small advantage over random guessing. It is known that the threshold degree of f is equivalent to its complexity $\mathbf{UPP}^{\mathrm{dt}}(f)$, while a fundamental matrix-analytic analog of threshold degree known as *sign-rank* characterizes $\mathbf{UPP}^{\mathrm{cc}}$. Similarly, large-error approximate degree characterizes the query complexity measure $\mathbf{PP}^{\mathrm{dt}}$, in the following sense: for any $d > 0$, $\widetilde{\deg}_{1-2^{-d}}(f) \geq \Omega(d) \iff \mathbf{PP}^{\mathrm{dt}}(f) \geq \Omega(d)$. Section 2 elaborates on these models and their many applications in learning theory, circuit complexity, and cryptography.

Our Results in a Nutshell. We prove two results about the threshold degree and large-error approximate degree of functions in AC^0.[2] First, we prove a tight $\tilde{\Omega}(n^{1/2})$ lower bound on the threshold degree (i.e., $\mathbf{UPP}^{\mathrm{dt}}$ complexity) of a natural function called SURJECTIVITY, which is computed by a depth three circuit with logarithmic bottom fan-in. This matches the previous best threshold degree lower bound for any AC^0 function, due to Sherstov [34]. Our analysis is much simpler than Sherstov's, which takes up the bulk of a (70+)-page manuscript [34]. An additional advantage of our analysis is that our lower bound on the threshold degree of SURJECTIVITY "lifts" to give a lower bound for the communication analog $\mathbf{UPP}^{\mathrm{cc}}$ as well. In particular, we obtain an $\Omega(n^{1/2})$ $\mathbf{UPP}^{\mathrm{cc}}$ lower bound for a related AC^0 function; this improves over the previous best $\mathbf{UPP}^{\mathrm{cc}}$ lower bound for AC^0, of $\Omega(n^{2/5})$ [12].

Second, we give nearly optimal bounds on the large-error approximate degree (and hence, $\mathbf{PP}^{\mathrm{dt}}$ complexity) of AC^0. For any constant $\delta > 0$, we show that there is an AC^0 function with ε-approximate degree $\Omega(n^{1-\delta})$, where $\varepsilon = 1 - 2^{-\Omega(n^{1-\delta})}$. This result lifts to an analogous $\mathbf{PP}^{\mathrm{cc}}$ lower bound.

---

[1] It is known that for any $d > 0$, there are functions of threshold degree d that cannot be approximated by degree d polynomials to error better than $1 - 2^{-\tilde{\Omega}(n^d)}$ [27], and this bound is tight [7]. Hence, threshold degree is also equivalent to the notion of ε-approximate degree for some value of ε that is *doubly*-exponentially close to 1.

[2] AC^0 is the non-uniform class of sequences of functions computed by polynomial size Boolean circuits of constant depth.

Table 1 Comparison of our new bounds for AC^0 to prior work in roughly chronological order. The circuit depth column lists the depth of the Boolean circuit used to exhibit the bound, δ denotes an arbitrarily small positive constant, and k an arbitrary positive integer. All Boolean circuits are polynomial size.

Reference	$\mathbf{PP}^{\mathrm{dt}}$ log(threshold weight)	$\mathbf{PP}^{\mathrm{cc}}$ log(1/discrepancy)	$\mathbf{UPP}^{\mathrm{dt}}$ threshold degree	$\mathbf{UPP}^{\mathrm{cc}}$ log(sign-rank)	Circuit Depth
[23]	—	—	$\Omega(n^{1/3})$	—	2
[20]	$\Omega(n^{1/3})$	—	—	—	3
[16]	—	$\Omega(\log^k(n))$	—	$\Omega(\log^k(n))$	$O(k)$
[25]	$\Omega(n^{1/3}\log^k n)$	—	$\Omega(n^{1/3}\log^k(n))$	—	$O(k)$
[29]	—	$\Omega(n^{1/5})$	—	—	3
[7,31]	—	$\Omega(n^{1/3})$	—	—	3
[28]	—	—	—	$\Omega(n^{1/3})$	3
[10]	$\Omega(n^{2/5})$	$\Omega(n^{2/5})$	—	—	3
[33]	$\Omega(n^{1/2-\delta})$	$\Omega(n^{1/2-\delta})$	$\Omega(n^{1/2-\delta})$	—	$O(1/\delta)$
[34]	$\Omega(n^{3/7})$	—	$\Omega(n^{3/7})$	—	3
[34]	$\Omega(n^{1/2})$	$\Omega(n^{1/2})$	$\Omega(n^{1/2})$	—	4
[12]	—	—	—	$\Omega(n^{2/5})$	3
[11]	$\Omega(n^{1/2-\delta})$	$\Omega(n^{1/2-\delta})$	—	—	3
This work	$\tilde{\Omega}(n^{1/2})$	$\tilde{\Omega}(n^{1/2})$	$\tilde{\Omega}(n^{1/2})$	—	3
This work	—	—	—	$\tilde{\Omega}(n^{1/2})$	7
This work	$\Omega(n^{1-\delta})$	$\Omega(n^{1-\delta})$	—	—	$O(1/\delta)$

To summarize our results succinctly:

- We prove a $\tilde{\Omega}(n^{1/2})$ lower bound on the **UPP** complexity of SURJECTIVITY in the query setting, and of a related AC^0 function in the communication setting.
- We prove a $\Omega(n^{1-\delta})$ lower bound on the **PP** complexity of some AC^0 circuit of depth $O(1/\delta)$, in both the query and communication settings.

Table 1 compares our new lower bounds for AC^0 to the long line of prior works with similar goals.

Context and Prior Work. The study of both large-error approximate degree and threshold degree has led to many breakthrough results in theoretical computer science, especially in the algorithmic and complexity-theoretic study of constant depth circuits. For example, threshold degree upper bounds are at the core of many of the fastest known PAC learning algorithms. This includes the notorious case of polynomial size CNF formulas on n variables, for which the fastest known algorithm [19] runs in time $\exp(\tilde{O}(n^{1/3}))$ owing to a $\tilde{O}(n^{1/3})$ upper bound on the threshold degree of any such formula. This upper bound is tight, matching a classic $\Omega(n^{1/3})$ lower bound of Minsky and Papert [23] for the following read-once CNF: $\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}$ (here, we use subscripts to clarify the number of inputs on which a function is defined).

In complexity theory, breakthrough results of Sherstov [29,31] and Buhrman et al. [7] used lower bounds on large-error approximate degree to show that there are AC^0 functions with polynomial $\mathbf{PP}^{\mathrm{cc}}$ complexity. One notable implication of these results is that Allender's [1] classic simulation of AC^0 functions by depth-three majority circuits is optimal. (This resolved an open problem of Krause and Pudlák [20].) A subsequent, related breakthrough of Razborov and Sherstov [28] used Minsky and Papert's lower bound on the threshold degree of $\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}$ to prove the first polynomial $\mathbf{UPP}^{\mathrm{cc}}$ lower bound for a function in AC^0, answering an old open question of Babai et al. [2].

These breakthrough lower bounds raised the intriguing possibility that AC^0 functions could be *maximally* hard for the $\mathbf{UPP}^{cc}$ and $\mathbf{PP}^{cc}$ communication models, as well as for related complexity measures. Nevertheless, the quantitative parameters achieved in these works are far from actually showing that this is the case. Indeed, the following basic questions about the complexity of AC^0 remain open.

▶ **Problem 1.** *Is there an AC^0 function $F\colon \{-1,1\}^{n\times n} \to \{-1,1\}$ with $\mathbf{UPP}^{cc}$ complexity $\Omega(n)$?*

▶ **Problem 2.** *Is there an AC^0 function $F\colon \{-1,1\}^{n\times n} \to \{-1,1\}$ with $\mathbf{PP}^{cc}$ complexity $\Omega(n)$?*

An affirmative answer to either question would be tight: *Every* function $F\colon \{-1,1\}^n \times \{-1,1\}^n \to \{-1,1\}$ has $\mathbf{UPP}^{cc}$ and $\mathbf{PP}^{cc}$ complexity at most n. Obtaining an affirmative answer to Open Problem 1 is harder than for Open Problem 2, since $\mathbf{UPP}^{cc}(f) \leq \mathbf{PP}^{cc}(f)$ for all f.

Guided by these open problems, a sequence of works has established quantitatively stronger and more general lower bounds for AC^0 functions [9–13, 33, 34]. In addition to making partial progress toward resolving these questions, the techniques developed in these works have found fruitful applications in new domains. For example, Bouland et al. [6] built on techniques from a number of aforementioned works [9,10,12,33] to resolve several old open questions about the relativized power of statistical zero knowledge proofs and their variants. As another example, our recent prior works [8,13] built on the same line of work to resolve or nearly resolve a number of longstanding open questions in quantum query complexity. Finally, large-error and threshold degree lower bounds on AC^0 functions have recently proved instrumental in the development of cryptographic secret-sharing schemes with reconstruction procedures in AC^0 [4,5,14]. We thus believe that the new techniques developed in this work will find further applications, perhaps in unexpected areas.

Prior to our work, the best known result toward a resolution of Open Problem 1 was a $\Omega(n^{2/5})$ lower bound on $\mathbf{UPP}^{cc}$ complexity of an AC^0 function [12], while the best known result toward Open Problem 2 was a $\Omega(n^{1/2})$ bound on the $\mathbf{PP}^{cc}$ complexity of a very complicated AC^0 circuit [34].

1.1 Our Results In Detail

1.1.1 Resolving the Threshold Degree of SURJECTIVITY

Surjectivity and its History. Let R be a power of 2 and $n = N \log R$. The function $\mathsf{SURJECTIVITY}_n$ ($\mathsf{SURJ}_{R,N}$ for short) is defined as follows. Given an input in $\{-1,1\}^n$, $\mathsf{SURJ}_{R,N}$ interprets the input as a list of N numbers $(s_1,\ldots,s_N)$ from a range $[R] := \{1,\ldots,R\}$, and evaluates to -1 if and only if every element of the range $[R]$ appears at least once in the list.[3] $\mathsf{SURJ}_{R,N}$ is computed by an AC^0 circuit of depth three and logarithmic bottom fan-in, since it is equivalent to the AND_R (over all range items $r \in [R]$) of the OR_N (over all inputs $i \in [N]$) of "Is input s_i equal to r?", where the quoted question is computed by a conjunction of width $\log R$ over the input bits.

$\mathsf{SURJ}_{R,N}$ has been studied extensively in the contexts of quantum query complexity and approximate degree. Beame and Machmouchi [3] showed that computing $\mathsf{SURJ}_{R,N}$ for $R = N/2 + 1$ requires $\tilde{\Omega}(n)$ quantum queries, making it the only known AC^0 function with linear quantum query complexity. Meanwhile, the (1/3)-approximate degree of $\mathsf{SURJ}_{R,N}$ was

[3] As is standard, we associate -1 with logical TRUE and $+1$ with logical FALSE throughout.

recently shown to be $\tilde{\Theta}(R^{1/4} \cdot N^{1/2})$. The lower bound is from our prior work [8], while the upper bound was shown by Sherstov [35], with a different proof given in [8]. In particular, when $R = N/2$, $\widetilde{\deg}_{1/3}(\mathsf{SURJ}_{R,N}) = \tilde{\Theta}(N^{3/4})$. Our prior works [8, 13] built directly on the approximate degree lower bound for $\mathsf{SURJ}_{R,N}$ to give near-optimal lower bounds on the (1/3)-approximate degree of AC^0 (see Section 3.3 for details).

Our Result. In spite of the progress described above, the threshold degree $\mathsf{SURJ}_{R,N}$ remained open. For $R < N/2$, an upper bound of $\tilde{O}(\min\{R, N^{1/2}\})$ follows from standard techniques. The best known lower bound was $\Omega(\min\{R, N^{1/3}\})$, obtained by a reduction to Minsky and Papert's threshold degree lower bound for $\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}$. In this work, we settle the threshold degree of $\mathsf{SURJ}_{R,N}$, showing that the known upper bound is tight up to logarithmic factors.

▶ **Theorem 3.** *For $R < N/2$, the threshold degree of $\mathsf{SURJ}_{R,N}$ is $\tilde{\Theta}(\min\{R, N^{1/2}\})$. In particular, if $R = N^{1/2}$, $\deg_{\pm}(\mathsf{SURJ}_{R,N}) = \tilde{\Theta}(N^{1/2})$.*

In addition to resolving a natural question in its own right, Theorem 3 matches the best prior threshold degree lower bound for AC^0, previously proved in [34] for a much more complicated function computed by a circuit of strictly greater depth. Furthermore, with some extra effort, our lower bound for $\mathsf{SURJ}_{R,N}$ extends to give a $\tilde{\Omega}(n^{1/2})$ lower bound on the $\mathbf{UPP}^{\mathrm{cc}}$ complexity of a related AC^0 function, yielding progress on Open Question 1 (cf. Section 1). In contrast, Sherstov's $\Omega(n^{1/2})$ threshold degree lower bound for AC^0 [34] is not known to extend to $\mathbf{UPP}^{\mathrm{cc}}$ complexity. As stated in Section 1, the best previous $\mathbf{UPP}^{\mathrm{cc}}$ lower bound for an AC^0 function was $\Omega(n^{2/5})$.

▶ **Corollary 4.** *There is an AC^0 function $F\colon \{-1,1\}^{n\times n} \to \{-1,1\}$ such that $\mathbf{UPP}^{cc}(F) \geq \tilde{\Omega}(n^{1/2})$.*

1.1.2 AC^0 Has Nearly Maximal $\mathrm{PP}^{\mathrm{cc}}$ Complexity

In our second result, for any constant $\delta > 0$, we exhibit an AC^0 function $f\colon \{-1,1\}^n \to \{-1,1\}$ with $\widetilde{\deg}_{\varepsilon}(f) = \Omega(n^{1-\delta})$ for some $\varepsilon = 1 - 2^{-\Omega(n^{1-\delta})}$. This is a major strengthening of our prior works [8, 13], which proved a similar result for $\varepsilon = 1/3$. By combining this large-error approximate degree lower bound with a "query-to-communication lifting theorem" for $\mathbf{PP}$ [31], we obtain a $\Omega(n^{1-\delta})$ bound on the $\mathbf{PP}^{\mathrm{cc}}$ complexity of an AC^0 function, nearly resolving Open Question 2 from the previous section.

▶ **Theorem 5.** *For any constant $\delta > 0$, there is an AC^0 function $F\colon \{-1,1\}^{n\times n} \to \{-1,1\}$ with $\mathbf{PP}^{cc}(F) = \Omega(n^{1-\delta})$.*

The best previous lower bound for the $\mathbf{PP}^{\mathrm{cc}}$ complexity of an AC^0 function was $\Omega(n^{1/2})$ [34].

2 Algorithmic and Complexity-Theoretic Applications

To introduce the applications of our results, we begin by defining the query complexity quantities $\mathbf{UPP}^{\mathrm{dt}}$ and $\mathbf{PP}^{\mathrm{dt}}$ and the communication complexity quantities $\mathbf{UPP}^{\mathrm{cc}}$ and $\mathbf{PP}^{\mathrm{cc}}$.

Query Models. In randomized query complexity, an algorithm aims to evaluate a known Boolean function f on an unknown input $x \in \{-1,1\}^n$ by reading as few bits of x as possible. We say that the *query cost* of a randomized algorithm is the maximum number of bits it queries for any input x.

- $\mathbf{UPP}^{\mathrm{dt}}$ considers "unbounded error" randomized algorithms, which means that on any input x, the algorithm outputs $f(x)$ with probability strictly greater than $1/2$. $\mathbf{UPP}^{\mathrm{dt}}(f)$ is the minimum query cost of any unbounded error algorithm for f.
- $\mathbf{PP}^{\mathrm{dt}}(f)$ captures "large" (rather than unbounded) error algorithms. If a randomized query algorithm outputs $f(x)$ with probability $1/2+\beta$ for all x, then the $\mathbf{PP}$-cost of the algorithm is the sum of the query cost and $\log(1/\beta)$. $\mathbf{PP}^{\mathrm{dt}}(x)$ is the minimum $\mathbf{PP}$-cost of any randomized query algorithm for f.

Communication Models. $\mathbf{UPP}^{\mathrm{cc}}$ and $\mathbf{PP}^{\mathrm{cc}}$ consider the standard two-party setup where Alice holds an input x and Bob holds an input y, and they run a private-coin randomized communication protocol to compute a function $f(x,y)$, while minimizing the number of bits they exchange. In direct analogy to the query complexity measures above, we say that the *communication cost* of a randomized protocol is the maximum number of bits Alice and Bob exchange on any input (x,y).

- $\mathbf{UPP}^{\mathrm{cc}}(f)$ [26] is the minimum communication cost of any randomized protocol that outputs $f(x,y)$ with probability strictly greater than $1/2$ on all inputs (x,y).
- $\mathbf{PP}^{\mathrm{cc}}(f)$ [2] is the minimum $\mathbf{PP}$-cost of a protocol for f, where the $\mathbf{PP}$-cost of a protocol that outputs $f(x,y)$ with probability $1/2+\beta$ for all (x,y) is the sum of the communication cost and $\log(1/\beta)$.

We now give an overview of the applications of Theorem 5 and Corollary 4.

2.1 Applications of Theorem 5

$\mathbf{PP}^{\mathrm{cc}}$ is known to be equivalent to two measures of central importance in learning theory and communication complexity, namely *margin complexity* [22] and *discrepancy* [18]. Hence, Theorem 5 implies that AC^0 has nearly maximal complexity under both measures. Below, we highlight four additional applications.

- **Communication Complexity**. The $\mathbf{PP}^{\mathrm{cc}}$ communication model can efficiently simulate almost every two-party communication model, including $\mathbf{P}$ (i.e., deterministic communication), $\mathbf{BPP}$ (randomized communication), $\mathbf{BQP}$ (quantum), and $\mathbf{P}^{\mathbf{NP}}$. The only well-studied exceptions are $\mathbf{UPP}^{\mathrm{cc}}$, and communication analogs of the polynomial hierarchy (the latter of which we do not know how to prove lower bounds against). Hence, in showing that AC^0 has essentially maximal $\mathbf{PP}^{\mathrm{cc}}$ complexity, we subsume or nearly subsume all previous results on the communication complexity of AC^0.
- **Cryptography**. Bogdanov et al. [4] observed that for any $f\colon \{-1,1\}^n \to \{-1,1\}$ and $d>0$, if one shows that $\widetilde{\deg}_{\varepsilon}(f) \geq d$, then one obtains a scheme for sharing a secret bit $b \in \{-1,1\}$ among n parties such that any subset of d shares provides no reconstruction advantage, yet applying f to all n shares yields b with probability at least $1/2+\varepsilon/2$. They combined this with known approximate degree lower bounds for AC^0 functions to get secret sharing schemes with reconstruction procedures in AC^0. Via this connection, an immediate corollary of Theorem 5 is a nearly optimal secret sharing scheme in AC^0: for any desired constant $\delta>0$, any subset of $n^{1-\delta}$ shares provides no reconstruction advantage, yet all n shares can be successfully reconstructed (by applying an AC^0 function) with probability $1-2^{-n^{1-\delta}}$.
- **Learning Theory**. Valiant [38] introduced the *evolvability* model in an effort to quantify how (and which) mechanisms can evolve in realistic population sizes within realistic time periods. Feldman [15] showed that the "weak evolvability" of a class of functions $\mathcal{F}=\{\phi_1,\ldots,\phi_{|\mathcal{F}|}\}$ is characterized by the $\mathbf{PP}^{\mathrm{cc}}$ complexity of the function $F(x,y)=\phi_x(y)$. Hence, a consequence of Theorem 5 is that there are AC^0 functions that are nearly

maximally hard to evolve (i.e., for any constant $\delta > 0$, there are AC^0 functions that require either $2^{n^{1-\delta}}$ generations, or populations of size $2^{n^{1-\delta}}$ to evolve, even if one only wants to evolve a mechanism that has advantage just $2^{-n^{1-\delta}}$ over random guessing). We also obtain a nearly optimal $2^{n^{1-\delta}}$ lower bound on the *threshold weight* of an AC^0 function. Threshold weight is another central quantity underlying many algorithmic results in learning theory. Our results rule out the possibility that algorithms based on threshold weight bounds can PAC learn AC^0 in time significantly faster than 2^n.

- **Circuit Complexity**. If $\mathbf{PP}^{cc}(f) \geq d$, then f is not computable by Majority-of-Threshold circuits of size $2^{\Omega(d)}$ [24]. Hence, by showing that AC^0 has nearly maximal $\mathbf{PP}^{cc}$ complexity, we show that there are AC^0 functions that are not computed by Majority-of-Threshold circuits of size $2^{n^{1-\delta}}$. That is, AC^0 has essentially no non-trivial simulation by Majority-of-Threshold circuits (in contrast, AC^0 can be efficiently simulated by depth-three Majority circuits [1]).

2.2 Applications of Corollary 4

As indicated in Section 1, $\mathbf{UPP}^{cc}(F)$ is known to be characterized by (the logarithm of) of the *sign-rank* of the matrix $[F(x,y)]_{x,y\in\{-1,1\}^{n\times n}}$ [26].[4] Hence, Corollary 4 implies an $\exp(\tilde{\Omega}(n^{1/2}))$ lower bound on the sign-rank of AC^0 function. Below, we highlight two additional applications of Corollary 4, based on the following connections between communication complexity, circuit complexity, and learning theory.

In communication complexity, $\mathbf{UPP}^{cc}$ is the most powerful two-party model against which we know how to prove lower bounds. In circuit complexity, if $\mathbf{UPP}^{cc}(f) \geq d$, then f cannot be computed by Threshold-of-Majority circuits of size $2^{\Omega(d)}$ [17]. (Threshold-of-Majority circuits represent the most powerful class of threshold circuits against which we can prove superpolynomial lower bounds.) In learning theory, it is commonly assumed that data can be classified by a halfspace in many dimensions; the $\mathbf{UPP}^{cc}$-complexity of a concept class precisely captures how many dimensions are needed. To connect this to a previously mentioned example, Klivans and Servedio [19] observed that an upper bound of d on the $\mathbf{UPP}^{cc}$ complexity of a concept class $\mathcal{C}$ yields a PAC learning for $\mathcal{C}$ running in time $2^{O(d)}$. They used this result to give a $2^{\tilde{O}(n^{1/3})}$-time algorithm for PAC-learning CNFs. This remains the state-of-the-art algorithm for this fundamental problem. Accordingly, Corollary 4 has the following implications.

- **Circuit Complexity.** There are AC^0 functions that are not computable by Threshold-of-Majority Circuits of size $2^{\tilde{\Omega}(n^{1/2})}$.
- **Learning Theory.** $\mathbf{UPP}^{cc}$-based learning algorithms cannot learn AC^0 in time better than $2^{\tilde{\Omega}(n^{1/2})}$.

3 Techniques

3.1 The SURJECTIVITY Lower Bound

For a function f_n, let $f^{\leq N}$ denote the partial function obtained by restricting f to the domain of inputs of Hamming weight at most N. The ε-approximate degree of $f^{\leq N}$, denoted $\widetilde{\deg}_{\varepsilon}(f^{\leq N})$, is the least degree of a real polynomial p such that

$$|p(x) - f(x)| \leq \varepsilon \text{ for all inputs } x \text{ of Hamming weight at most } N. \tag{1}$$

[4] The sign-rank of a matrix M with entries in $\{\pm 1\}$ is the least rank of a real matrix M' that agrees in sign with M entry-wise.

Note that Property (1) allows p to *behave arbitrarily* on inputs x of Hamming weight more than N. Similarly, the threshold degree of $f^{\leq N}$ is the least degree of a real polynomial p such that

$$p(x) \cdot f(x) > 0 \text{ for all inputs } x \text{ of Hamming weight at most } N.$$

Our prior work [13] showed the ε-approximate (respectively, threshold) degree of $\mathsf{SURJ}_{R,N}$ is *equivalent* to the ε-approximate (respectively, threshold) degree of $(\mathsf{AND}_R \circ \mathsf{OR}_N)^{\leq N}$. Hence, the main technical result underpinning our threshold degree lower bound for SURJ is the following theorem about the threshold degree of $(\mathsf{AND}_R \circ \mathsf{OR}_N)^{\leq N}$ (we have made no effort to optimize the logarithmic factors).

▶ **Theorem 6.** *Let* $R = N^{1/2}$. *Then* $\deg_{\pm}\left((\mathsf{AND}_R \circ \mathsf{OR}_N)^{\leq N}\right) = \Omega(N^{1/2}/\log^{3/2} N)$.

Discussion. Theorem 6 is a substantial strengthening of the classic result of Minsky and Papert [23] mentioned above, which established that the total function $\mathsf{MP}_{N^{1/2},N} := \mathsf{AND}_{N^{1/2}} \circ \mathsf{OR}_N$ on $n = N^{3/2}$ inputs has threshold degree $\Omega(N^{1/2})$. Theorem 6 establishes that Minsky and Papert's lower bound holds even under the promise that the input has Hamming weight at most $N = n^{2/3}$. That is, any polynomial that sign-represents $\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}$ on inputs of Hamming weight at most $n^{2/3}$ has degree $\tilde{\Omega}(n^{1/3})$, *even when* p *is allowed to behave arbitrarily on inputs of Hamming weight larger than* $n^{2/3}$.

Proof overview for Theorem 6 and comparison to prior work. Like much recent work on approximate and threshold degree lower bounds, our proof makes use of *dual polynomials*. A dual polynomial is a dual solution to a certain linear program capturing the approximate or threshold degree of any function, and acts as a certificate of the high approximate or threshold degree of the function.

A dual polynomial that witnesses the fact that $\deg_{\pm}(f_M) \geq d$ is a function $\psi\colon \{-1,1\}^M \to \{-1,1\}$ satisfying three properties:

- $\psi(x) \cdot f(x) \geq 0$ for all $x \in \{-1,1\}^M$. If ψ satisfies this condition, we say ψ agrees in sign with f.
- $\sum_{x\in\{-1,1\}^M} |\psi(x)| = 1$. If ψ satisfies this condition, it is said to have ℓ_1-norm equal to 1.
- For all polynomials $p\colon \{-1,1\}^M \to \mathbb{R}$ of degree at most d, $\sum_{x\in\{-1,1\}^M} p(x) \cdot \psi(x) = 0$. If ψ satisfies this condition, it is said to have *pure high degree* at least d.

A dual witness for the fact that $\widetilde{\deg}_{\varepsilon}(f_M) \geq d$ is similar, except that the first condition is replaced with:

- $\sum_{x\in\{-1,1\}^M} \psi(x) \cdot f(x) > \varepsilon$. If ψ satisfies this condition, it is said to be ε-*correlated* with f. If $\psi(x) \cdot f(x) < 0$, we say that ψ *makes an error* at x.

Sherstov [34] reproved Minsky and Papert's result by constructing an explicit dual witness for $\mathsf{MP}_{N^{1/2},N}$, via a two-step process. First, Sherstov started with a dual witness ψ_{base} for the fact that

$$\widetilde{\deg}_{\varepsilon}(\mathsf{MP}_{N^{1/2},N}) = \Omega(N^{1/2}), \text{ for } \varepsilon = 1 - 2^{-N^{1/2}}.$$

The function ψ_{base} was introduced in our prior work [10], where it was constructed by combining a dual witness for $\mathsf{AND}_{N^{1/2}}$ with a dual witness for OR_N via a technique called dual block composition [21, 32, 37].

Unfortunately, ψ_{base} falls short of witnessing Minsky and Papert's threshold degree lower bound because it makes errors on some inputs. In the second step of Sherstov's construction [34], he adds in a correction term that zeros out the errors of ψ_{base}, without disturbing the sign of ψ_{base} on any other inputs, and without lowering its pure high degree.

Theorem 6 asserts that $\mathsf{MP}^{\leq N}_{N^{1/2},N}$ satisfies the same threshold degree lower bound as $\mathsf{MP}_{N^{1/2},N}$ itself. To prove Theorem 6, we need to construct a dual witness ψ that not only reproves Minsky and Papert's classic lower bound for $\mathsf{MP}_{N^{1/2},N}$, but also satisfies the extra condition that:

$$\psi(x) = 0 \text{ for all inputs } x \text{ of Hamming weight more than } N. \tag{2}$$

To accomplish this, we apply a novel strategy that can be thought of as a three-step process. First, like Sherstov, we start with ψ_{base}. Second, we modify ψ_{base} to obtain a dual witness ψ'_{base} that places significant mass on all inputs of Hamming weight at most d, for some $d = \tilde{\Omega}(N^{1/2})$ (details of the construction of ψ'_{base} are described two paragraphs hence). More specifically, we ensure that ψ'_{base} satisfies:

$$|\psi'_{\mathsf{base}}(x)| \gg n^{-d} \text{ for all inputs } x \text{ of Hamming weight at most } d. \tag{3}$$

We refer to this property by saying that ψ'_{base} is "smooth" or "large" on all inputs of Hamming weight at most d. Note that, in modifying ψ_{base} to obtain ψ'_{base}, we do *not* correct the errors that ψ_{base} makes, nor do we ensure that ψ'_{base} is supported on inputs of Hamming weight at most N.

Third, we add in a correction term, very different than Sherstov's correction term, that not only zeros out the errors of ψ'_{base}, but also zeros out any mass it places on inputs of Hamming weight more than N. While the general technique we use to construct this correction term appeared in our prior works [8, 13], the novelty in our construction and analysis is two-fold. First, the technique was used in our prior work only to zero out mass placed on inputs of Hamming weight more than N (i.e., to ensure that Equation (2) is satisfied), not to correct errors. Second, and more importantly, we crucially exploit the largeness of ψ'_{base} on inputs of Hamming weight at most d to ensure that the correction term does not disturb the sign of ψ'_{base} on any inputs other than those on which it is deliberately being zeroed out. This is what enables us to obtain a threshold degree lower bound, whereas our prior works [8, 13] were only able to obtain ε-approximate degree lower bounds for ε bounded away from 1.

Our "smoothing followed by correction" approach appears to be significantly more generic than the correction technique of [34]. For example, prior work of Bouland et al. [6] proved an $\Omega(n^{1/4})$ lower bound on the threshold degree of a certain function denoted $\mathsf{GAPMAJ}_{n^{1/4}} \circ \mathsf{PTP}_{n^{3/4}}$, and used this result to give an oracle separating the oracle complexity classes SZK and UPP, thereby answering an open question of Watrous from 2002. Our techniques can be used to give a much simpler proof of this result, as well as several others appearing in the literature (for brevity, we omit the details of these simpler proofs of prior results). We are confident that our technique will find additional applications in the future.

Details of the smoothing step. As stated above, the dual witness ψ_{base} from our prior work does not satisfy the property we need (cf. Equation (3)) of being "large" on all inputs of Hamming weight at most $d = \tilde{\Omega}(N^{1/2})$.

Fortunately, we observe that although ψ_{base} is *not* large on all inputs of Hamming weight at most d, it *is* large on one very special input of low Hamming weight, namely the $\mathsf{ALL\text{-}FALSE}$ input. That is, $\psi_{\mathsf{base}}(\mathbf{1}) \geq 2^{-d}$. So we just need a way to "bootstrap" this largeness property

on **1** to a largeness property on all inputs of Hamming weight at most d. Put another way, we need to be able to treat other inputs of Hamming weight at most d as if they actually have Hamming weight 0. But $\mathsf{MP}_{N^{1/2},N} := \mathsf{AND}_{N^{1/2}} \circ \mathsf{OR}_N$ has a property that enables precisely this: we can fix the inputs to any constant fraction c of the OR gates to an arbitrary value in $\mathsf{OR}^{-1}(-1)$, and the remaining function of the unrestricted inputs is $\mathsf{AND}_{(1-c)\cdot R} \circ \mathsf{OR}_N$. This is "almost" the same function as $\mathsf{AND}_R \circ \mathsf{OR}_N$; we have merely slightly reduced the top fan-in, which does not substantially lower the threshold degree of the resulting function.

We exploit the above observation to achieve the following: for each input x of Hamming weight at most d, we build a dual witness ν_x targeted at x (i.e., that essentially treats x as if it is the $\mathsf{ALL\text{-}FALSE}$ input). We do this as follows. Let T be the set of all OR gates that are fed one or more -1s by x, and let $S \subseteq [N^{1/2} \cdot N]$ be the union of the inputs to each of the OR gates in T. Let ψ_{base} be the dual witness for $\mathsf{AND}_{N^{1/2}-|T|} \circ \mathsf{OR}_N$ given in our prior work [10]. We let

$$\nu_x(y) = \begin{cases} \psi_{\mathsf{base}}(y_{\bar{S}}) & \text{if } y_S = x_S \\ 0 & \text{otherwise,} \end{cases}$$

where $y_{\bar{S}}$ denotes the set of all the coordinates of y other than those in S.

The dual witness ψ'_{base} is then defined to be the average of the ν_x's, over all inputs x of Hamming weight at most d. This averaged dual witness ψ'_{base} has all of the same useful properties as ψ_{base}, and additionally satisfies the key requirement captured by Equation (3).

3.2 Extension to $\mathrm{UPP}^{\mathrm{cc}}$: Proof of Corollary 4

Building on the celebrated framework of Forster [16], Razborov and Sherstov [28] developed techniques to translate threshold degree lower bounds into sign-rank lower bounds. Specifically, they showed that, in order for a threshold degree lower bound of the form $\deg_{\pm}(f_n) \geq d$ to translate into a $\mathbf{UPP}^{\mathrm{cc}}$ lower bound for a related function F, it suffices for the threshold degree lower bound for f_n to be exhibited by a dual witness ϕ satisfying the following smoothness condition:

$$|\phi(x)| \geq 2^{-O(d)} \cdot 2^{-n} \text{ for all but a } 2^{-\Omega(d)} \text{ fraction of inputs } x \in \{-1,1\}^n. \tag{4}$$

Note that this is a different smoothness condition than the one satisfied by the dual witness ψ'_{base} discussed above for $\mathsf{MP}_{N^{1/2},N}$ (cf. Equation (3)): on inputs x of Hamming weight at most d, $|\psi'_{\mathsf{base}}(x)|$ is always at least $n^{-d} \gg 2^{-d} \cdot 2^{-n}$, whereas on inputs x of Hamming weight more than d, $|\psi'_{\mathsf{base}}(x)|$ may be 0. In words, $|\psi'_{\mathsf{base}}(x)|$ is *very large* on inputs x of Hamming weight at most d, but may not be large at all on inputs of larger Hamming weight. In contrast, Equation (4) requires a dual witness to be "somewhat large" (within a $2^{-O(d)}$ factor of uniform) on *nearly all* inputs.

In summary, our construction of a dual witness for $\mathsf{MP}^{\leq N}_{N^{1/2},N}$ that is sketched in the previous subsection is not sufficient to apply Razborov and Sherstov's framework to $\mathsf{SURJ}_{R,N}$, for two reasons. First, the dual witness we construct for $\mathsf{MP}^{\leq N}_{N^{1/2},N}$ is not smooth in the sense of Equation (4), as it is only "large" on inputs of Hamming weight at most d. Second, to apply Razborov and Sherstov's framework to $\mathsf{SURJ}_{R,N}$, we actually need to give a smooth dual witness for $\mathsf{SURJ}_{R,N}$ itself, not for $\mathsf{MP}^{\leq N}_{N^{1/2},N}$. Note that $\mathsf{SURJ}_{R,N}$ is defined over the domain $\{-1,1\}^n$ where $n = N \log R$, while $\mathsf{MP}^{\leq N}_{N^{1/2},N}$ is defined over subset of $\{-1,1\}^{NR}$ consisting of inputs of Hamming weight at most N.

We address both of the above issues as follows. First, we show how to turn our dual witness μ for $\mathsf{MP}^{\leq N}_{N^{1/2},N}$ into a dual witness $\hat{\sigma}$ for the fact that $\deg_{\pm}(\mathsf{SURJ}_{R,N}) \geq d$, such that $\hat{\sigma}$ inherits the "largeness" property of μ on inputs of Hamming weight at most d. Second, we transform

$\hat{\sigma}$ into a dual witness τ for the fact that $\deg_{\pm}\left(\mathsf{SURJ}_{R,N} \circ \mathsf{AND}_{\log^2 n} \circ \mathsf{PARITY}_{\log^3 n}\right) \geq d$, such that τ satisfies the smoothness condition given in Equation (4). We conclude that $\mathsf{SURJ}_{R,N} \circ \mathsf{AND}_{\log^2 n} \circ \mathsf{PARITY}_{\log^3 n}$ can be transformed into a related function F (on $\tilde{O}(n)$ inputs, and which is also in AC^0) that has sign-rank $\exp(\tilde{\Omega}(n^{1/2}))$.

3.3 The $\mathrm{PP}^{\mathrm{cc}}$ Bound: Proof of Theorem 5

As mentioned in Section 1.1.2, the core of Theorem 5 is to exhibit an AC^0 function f such that $\widetilde{\deg}_{\varepsilon}(f) = \Omega(n^{1-\delta})$ for some $\varepsilon = 1 - 2^{-\Omega(n^{1-\delta})}$. To accomplish this, we prove a hardness amplification theorem that should be understood in the context of a weaker result from our prior work [13].

As stated in Section 3.1, for $\varepsilon = 1/3$, our prior work [13] showed how to take any Boolean function f_n in AC^0 with ε-approximate degree d and transform it into a related function g on roughly the same number of variables, such that g is still in AC^0, and g has significantly higher ε'-approximate degree for some $\varepsilon' \approx 1/3$. This was done in a two-step process. First, we showed that in order to construct a "harder" function g, it is sufficient to identify an AC^0 function G defined on $\mathrm{poly}(n)$ inputs such that for some $\ell = n \cdot \mathrm{polylog}(n)$, $\widetilde{\deg}_{\varepsilon'}(G^{\leq \ell}) \gg d$.[5] Second, we exhibited such a G. In our prior works [8, 13], for general functions f_n, the function G was $f_n \circ \mathsf{AND}_r \circ \mathsf{OR}_{m'}$, where $r = 10 \log n$, and $m' = \Theta(n/d)$.

We would like to prove a similar result, but we require that G have larger ε'-approximate degree than f_n, *where ε' is exponentially closer to 1 than is ε itself.* Unfortunately, the definition of G from our prior works [8, 13] does not necessarily result in such a function. For example, if $f_n = \mathsf{OR}_n$ (or any polylogarithmic DNF for that matter), then the function $G = f_n \circ \mathsf{AND}_r \circ \mathsf{OR}_{m'}$ is also a DNF of polylogarithmic width, and it is not hard to see that all such DNFs have ε-approximate degree at most $\mathrm{polylog}(n)$ for some $\varepsilon = 1 - 1/n^{\mathrm{polylog}(n)}$.

To address this situation, we change the definition of G. Rather than defining $G := f_n \circ \mathsf{AND}_r \circ \mathsf{OR}_{m'}$, we define $G = \mathsf{GAPMAJ}_t \circ f_z \circ \mathsf{AND}_r \circ \mathsf{OR}_m$ for appropriately chosen settings of the parameters t, z, r, and m. Here, GAPMAJ_t denotes any function evaluating to 1 on inputs of Hamming weight at most $t/3$, -1 on inputs of Hamming weight at least $2t/3$, and taking any value in $\{-1, 1\}$ on all other inputs (such functions are also called *approximate majorities*, and it is known that there are approximate majorities computable in AC^0). GAPMAJ has also played an important role in related prior work [6, 13].

In order to show that $\widetilde{\deg}_{\varepsilon'}(G^{\leq \ell}) \gg \widetilde{\deg}_{\varepsilon}(f_n)$ for an ε' that is exponentially closer to 1 than is ε, we require a more delicate construction of a dual witness than our prior works [8, 13]. After all, our prior works only required a dual witness for $G^{\leq \ell}$ with correlation at least $1/3$ with G^{ℓ}, while we require a dual witness achieving correlation with $G^{\leq \ell}$ that is exponentially close to 1. Roughly speaking, whereas our prior works [8, 13] were able to get away with exclusively using the simple and clean technique called dual block composition for constructing dual witnesses, we use a closely related but more involved construction introduced by Sherstov [30]. (Sherstov introduced his construction to prove that approximate degree satisfies a type of direct-sum theorem.)

More specifically, suppose that for some positive integer k, f_z has $\varepsilon(z)$-approximate degree at least $d(z) = z^{k/(k+1)}$, where $\varepsilon(z) = 1 - 2^{-z^{k/(k+1)}}$. In our definition of G, we set $t = n^{1/(k+2)}$, $z = n^{(k+1)/(k+2)}$, $r = 10 \log n$, and $m = n^{2/(k+2)}$, and we build a dual witness for $G^{\leq \ell}$ via a multi-step construction.

[5] This step was also used in the analysis of $\mathsf{SURJ}_{R,N}$ outlined in Section 3.2 above, where G was the function $\mathsf{AND}_R \circ \mathsf{OR}_N$.

In Step 1, we take dual witnesses ψ_{f_z}, ψ_{AND_r}, and ψ_{OR_m} for f_z, AND_r, and OR_m respectively, and we combine them using the technique of Sherstov [30], to give a dual witness γ for $f_z \circ \mathsf{AND}_r \circ \mathsf{OR}_m$ satisfying the following properties: γ has pure high degree at least $D(n) = n^{(k+1)/(k+2)} = d(n)^{(k+1)/k} \gg d(n)$, and γ's correlation with with $f_z \circ \mathsf{AND}_r \circ \mathsf{OR}_m$ is $\varepsilon'' \approx \varepsilon(z)$. That is, γ witnesses the fact that the ε''-approximate degree of $f_z \circ \mathsf{AND}_r \circ \mathsf{OR}_m$ is much larger than the $\varepsilon(n)$-approximate degree of f_n itself.

This step of the construction is in contrast to our prior work, which constructed a dual witness for $f_n \circ \mathsf{AND}_r \circ \mathsf{OR}_{m'}$ via direct dual block composition of ψ_{f_n}, ψ_{AND_r}, and ψ_{OR_m}. Direct dual block composition does not suffice for us because it would yield a dual witness with significantly worse correlation with $f_z \circ \mathsf{AND}_r \circ \mathsf{OR}_m$ than $\varepsilon(z)$.

While achieving correlation $\varepsilon'' \approx \varepsilon(z)$ is an improvement over what would obtain from direct dual block composition, it is still significantly farther from 1 than is $\varepsilon(n)$, i.e., $1 - \varepsilon'' \gg 1 - \varepsilon(n)$. And we ultimately need to construct a dual witness for $G^{\leq \ell}$ that is significantly *closer* to 1 than is $\varepsilon(n)$. To address this issue, in Step 2 of our construction, we use dual block composition to turn γ into a dual witness η for $G = \mathsf{GAPMAJ}_t \circ f_z \circ \mathsf{AND}_r \circ \mathsf{OR}_m$ satisfying the following properties: η has the same pure high degree as γ, and moreover η has correlation at least $\varepsilon' = 1 - 2^{-\Omega(n^{(k+1)/(k+2)})}$ with G.

However, after Step 2, we are still not done, because η places some mass on inputs of Hamming weight as large as $t \cdot z \cdot r \cdot m \gg \ell$. Hence η is only a dual witness to the high ε'-approximate degree of G, not the high ε'-approximate degree of $G^{\leq \ell}$ (recall that any dual witness witness for $G^{\leq \ell}$, must evaluate to 0 on all inputs of Hamming weight larger than ℓ, cf. Equation (2)). Nonetheless, as in our prior work [8, 13], we are able to argue that η places *very little* mass on inputs of Hamming weight more than ℓ, and thereby invoke techniques from our prior work [8, 13] to zero out this mass. The reason this final step of the argument is not immediate from our prior work [8, 13] is as follows. Although prior work has developed a precise understanding of how much mass is placed on inputs of Hamming weight more than ℓ by dual witnesses constructed via basic dual block composition, the dual witness γ for $f_z \circ \mathsf{AND}_r \circ \mathsf{OR}_m$ that we constructed in Step 1 was *not* built by invoking pure dual block composition. Our key observation is that Sherstov's technique that we invoked to construct γ is "similar enough" to vanilla dual block composition that the precise understanding of dual block composition developed in our prior work can be brought to bear on our dual witness η.

In summary, there are two main technical contributions in our proof of Theorem 5. The first is the identification of a hardness amplification construction for ε-approximate degree that not only amplifies the degree against which the lower bound holds, but also the error parameter ε. The second is constructing a dual polynomial to witness the claimed lower bound, using techniques more involved and delicate than the vanilla dual block composition technique that sufficed in our prior works [8, 13].

4 Subsequent Work and Discussion

Subsequent to our work, Sherstov and Wu [36] have made major progress toward resolving Open Problem 1 by showing nearly optimal threshold degree and sign-rank lower bounds for AC^0. Specifically, for every $k \geq 1$, they exhibit a family of depth-k AC^0 circuits with threshold degree $\tilde{\Omega}(n^{(k-1)/(k+1)})$. This generalizes Minsky and Papert's lower bound of $\Omega(n^{1/3})$ on the threshold degree of DNF, as well as our lower bound of $\tilde{\Omega}(n^{1/2})$ for the depth-3 $\mathsf{SURJECTIVITY}$ function. Sherstov and Wu, moreover, show that for any positive constant $\delta > 0$ there is a family of AC^0 circuits with depth $O(1/\delta)$ and sign-rank $\exp\left(\tilde{\Omega}(n^{1-\delta})\right)$. This gives an almost optimal improvement to our sign-rank lower bound of $\exp\left(\tilde{\Omega}(n^{1/2})\right)$ on an AC^0 function.

As in our proof of Theorem 5, as well as our prior work [13], Sherstov and Wu obtain their threshold degree lower bound for AC^0 by recursively applying a new hardness amplification theorem. Their hardness amplification theorem shows how to convert a function f_z into a new function g_n, computable by circuits with slightly higher depth and roughly the same size, but with polynomially larger threshold degree. Again as in the proof of Theorem 5, in order to obtain such a g, it suffices to construct a function G with $\deg_{\pm}(G^{\leq n}) \gg \deg_{\pm}(f)$. Starting from a function f_z with threshold degree $z^{(k-1)/(k+1)}$, the function G that they identify as sufficient for this purpose is $G = f_z \circ \mathsf{MP}_{r,r^2}$, where $z = n^{(k+1)/(k+3)}$ and $r = n^{2/(k+3)}$. When f_z is a trivial function, this recovers our lower bound of $\tilde{\Omega}(n^{1/2})$ for $\mathsf{SURJECTIVITY}$. Hence, their construction in full can be viewed as a generalization of our Theorem 6 that is amenable to recursive application. This requires several technical new ideas in the construction of the dual witness. However, we remain optimistic that the simplicity of our analysis for $\mathsf{SURJECTIVITY}$ will nonetheless lead to future applications of our techniques.

Sherstov and Wu's sign-rank lower bound follows from a similar high-level (though more technically demanding) strategy, where they show that *smooth* threshold degree also obeys such a hardness amplification theorem.

While these new results resolve the most glaring question raised in the initial version of this work, a number of interesting directions remain for further study. A common feature of our large-error approximate degree lower bound and Sherstov and Wu's threshold degree and sign-rank lower bounds for AC^0 is that, in order to obtain lower bounds of the form $\Omega(n^{1-\delta})$, we must consider functions computed by circuits of depth $\Theta(1/\delta)$. This contrasts with the situation for bounded error approximate degree [13], where a lower bound of $\Omega(n^{1-\delta})$ can be obtained at depth only $O(\log(1/\delta))$. Can one show that there are AC^0 functions f of depth $O(\log(1/\delta))$ with $\widetilde{\deg}_{\varepsilon}(f) = \Omega(n^{1-\delta})$ for $\varepsilon = 1 - 2^{-\Omega(n^{1-\delta})}$ or with $\deg_{\pm}(f) = \Omega(n^{1-\delta})$? There is a common underlying reason why our construction and Sherstov and Wu's construction both require circuits of depth $\Theta(1/\delta)$ and not $\Theta(\log(1/\delta))$: a component of the hardness amplifier in both constructions (in our case, $\mathsf{GAPMAJ}_{n^{1/(k+1)}}$, and in Sherstov and Wu's case, the top gate of MP_{r,r^2}) is used to amplify error but does not amplify degree. In contrast, in the construction of [13] for lower bounding bounded-error approximate degree, up to a logarithmic factor, all of the hardness amplifier is used to amplify degree.

We would also like to highlight the question of proving sublinear *upper bounds* on the threshold degree of AC^0. Given the surprising $O(R^{1/4} \cdot N^{1/2})$ upper bound on the $(1/3)$-approximate degree of $\mathsf{SURJ}_{R,N}$ from recent works [8,35], we have begun to seriously entertain the possibility that for every function f computable by AC^0 of depth k, there is some constant $\delta(k) > 0$ such that the threshold degree (and possibly even $(1/3)$-approximate degree) of f is $O(n^{1-\delta})$. Unfortunately, we cannot currently even show that this is true for depth three circuits of quadratic size. Any progress in this direction would be very interesting, and we believe that such progress would likely lead to new circuit lower bounds.

References

1 Eric Allender. A note on the power of threshold circuits. In *Foundations of Computer Science, 1989., 30th Annual Symposium on*, pages 580–584. IEEE, 1989.

2 László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory (preliminary version). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 337–347. IEEE Computer Society, 1986. doi:10.1109/SFCS.1986.15.

3 Paul Beame and Widad Machmouchi. The quantum query complexity of AC^0. *Quantum Information & Computation*, 12(7-8):670–676, 2012. URL: http://www.rintonpress.com/xxqic12/qic-12-78/0670-0676.pdf.

4 Andrej Bogdanov, Yuval Ishai, Emanuele Viola, and Christopher Williamson. Bounded Indistinguishability and the Complexity of Recovering Secrets. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 593–618. Springer, 2016. doi:10.1007/978-3-662-53015-3_21.

5 Andrej Bogdanov and Christopher Williamson. Approximate Bounded Indistinguishability. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 53:1–53:11, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICALP.2017.53.

6 Adam Bouland, Lijie Chen, Dhiraj Holden, Justin Thaler, and Prashant Nalini Vasudevan. On The Power of Statistical Zero Knowledge. In *To Appear In Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS)*, 2017. Preliminary version available at http://eccc.hpi-web.de/report/2016/140.

7 Harry Buhrman, Nikolai K. Vereshchagin, and Ronald de Wolf. On Computation and Communication with Small Bias. In *22nd Annual IEEE Conference on Computational Complexity (CCC 2007), 13-16 June 2007, San Diego, California, USA*, pages 24–32. IEEE Computer Society, 2007. doi:10.1109/CCC.2007.18.

8 Mark Bun, Robin Kothari, and Justin Thaler. The polynomial method strikes back: Tight quantum query bounds via dual polynomials. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 297–310. ACM, 2018.

9 Mark Bun and Justin Thaler. Dual Lower Bounds for Approximate Degree and Markov-Bernstein Inequalities. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *ICALP (1)*, volume 7965 of *Lecture Notes in Computer Science*, pages 303–314. Springer, 2013. doi:10.1007/978-3-642-39206-1_26.

10 Mark Bun and Justin Thaler. Hardness Amplification and the Approximate Degree of Constant-Depth Circuits. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 268–280. Springer, 2015. Full version available at http://eccc.hpi-web.de/report/2013/151. doi:10.1007/978-3-662-47672-7_22.

11 Mark Bun and Justin Thaler. Approximate Degree and the Complexity of Depth Three Circuits. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:121, 2016. URL: http://eccc.hpi-web.de/report/2016/121.

12 Mark Bun and Justin Thaler. Improved Bounds on the Sign-Rank of AC^0. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 37:1–37:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.37.

13 Mark Bun and Justin Thaler. A Nearly Optimal Lower Bound on the Approximate Degree of AC^0. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 1–12, 2017. doi:10.1109/FOCS.2017.10.

14 Kuan Cheng, Yuval Ishai, and Xin Li. Near-Optimal Secret Sharing and Error Correcting Codes in AC^0. In *Theory of Cryptography Conference*, pages 424–458. Springer, 2017.

15 Vitaly Feldman. Evolvability from learning algorithms. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 619–628. ACM, 2008.

16 Jürgen Forster. A Linear Lower Bound on the Unbounded Error Probabilistic Communication Complexity. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity, Chicago, Illinois, USA, June 18-21, 2001*, pages 100–106. IEEE Computer Society, 2001. doi:10.1109/CCC.2001.933877.

17 Jürgen Forster, Matthias Krause, Satyanarayana V. Lokam, Rustam Mubarakzjanov, Niels Schmitt, and Hans Ulrich Simon. Relations Between Communication Complexity, Linear Arrangements, and Computational Complexity. In Ramesh Hariharan, Madhavan Mukund, and V. Vinay, editors, *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science, 21st Conference, Bangalore, India, December 13-15, 2001, Proceedings*, volume 2245 of *Lecture Notes in Computer Science*, pages 171–182. Springer, 2001. doi:10.1007/3-540-45294-X_15.

18 Hartmut Klauck. Lower bounds for quantum communication complexity. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 288–297. IEEE, 2001.

19 Adam R. Klivans and Rocco A. Servedio. Learning DNF in time $2^{\tilde{O}(n^{1/3})}$. *J. Comput. Syst. Sci.*, 68(2):303–318, 2004. doi:10.1016/j.jcss.2003.07.007.

20 Matthias Krause and Pavel Pudlák. On the Computational Power of Depth-2 Circuits with Threshold and Modulo Gates. *Theor. Comput. Sci.*, 174(1-2):137–156, 1997. doi:10.1016/S0304-3975(96)00019-9.

21 Troy Lee. A note on the sign degree of formulas. *CoRR*, abs/0909.4607, 2009. arXiv:0909.4607.

22 Nati Linial and Adi Shraibman. Learning complexity vs communication complexity. *Combinatorics, Probability and Computing*, 18(1-2):227–245, 2009.

23 Marvin Minsky and Seymour Papert. *Perceptrons - an introduction to computational geometry*. MIT Press, 1969.

24 Noam Nisan. The Communication Complexity of Threshold Gates. In *Combinatorics, Paul Erdos is Eighty*, pages 301–315, 1994.

25 Ryan O'Donnell and Rocco A. Servedio. New degree bounds for polynomial threshold functions. *Combinatorica*, 30(3):327–358, 2010. Preliminary version in STOC 2003. doi:10.1007/s00493-010-2173-3.

26 Ramamohan Paturi and Janos Simon. Probabilistic Communication Complexity. *J. Comput. Syst. Sci.*, 33(1):106–123, 1986. doi:10.1016/0022-0000(86)90046-2.

27 Vladimir V Podolskii. Perceptrons of large weight. In *International Computer Science Symposium in Russia*, pages 328–336. Springer, 2007.

28 Alexander A. Razborov and Alexander A. Sherstov. The Sign-Rank of AC^0. *SIAM J. Comput.*, 39(5):1833–1855, 2010. doi:10.1137/080744037.

29 Alexander A. Sherstov. Separating AC^0 from Depth-2 Majority Circuits. *SIAM J. Comput.*, 38(6):2113–2129, 2009. doi:10.1137/08071421X.

30 Alexander A. Sherstov. Strong direct product theorems for quantum communication and query complexity. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 41–50. ACM, 2011. doi:10.1145/1993636.1993643.

31 Alexander A. Sherstov. The Pattern Matrix Method. *SIAM J. Comput.*, 40(6):1969–2000, 2011. Preliminary version in *STOC* 2008. doi:10.1137/080733644.

32 Alexander A. Sherstov. The Intersection of Two Halfspaces Has High Threshold Degree. *SIAM J. Comput.*, 42(6):2329–2374, 2013. Preliminary version in FOCS 2009. doi:10.1137/100785260.

33 Alexander A. Sherstov. Breaking the Minsky-Papert barrier for constant-depth circuits. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 223–232. ACM, 2014. doi:10.1145/2591796.2591871.

34 Alexander A. Sherstov. The Power of Asymmetry in Constant-Depth Circuits. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 431–450, 2015. doi:10.1109/FOCS.2015.34.

35 Alexander A. Sherstov. Algorithmic polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:10, 2018. To appear in STOC 2018. URL: `https://eccc.weizmann.ac.il/report/2018/010`.

36 Alexander A. Sherstov and Pei Wu. Near-Optimal Lower Bounds on the Threshold Degree and Sign-Rank of AC^0. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:3, 2019. URL: `https://eccc.weizmann.ac.il/report/2019/003`.

37 Yaoyun Shi and Yufan Zhu. Quantum communication complexity of block-composed functions. *Quantum Information & Computation*, 9(5):444–460, 2009. URL: `http://www.rintonpress.com/xxqic9/qic-9-56/0444-0460.pdf`.

38 Leslie G Valiant. Evolvability. *Journal of the ACM (JACM)*, 56(1):3, 2009.

String Matching: Communication, Circuits, and Learning

Alexander Golovnev
Harvard University, Cambridge, MA, USA
alex.golovnev@gmail.com

Mika Göös
Institute for Advanced Study, Princeton, NJ, USA
mika@ias.edu

Daniel Reichman
Department of Computer Science, Princeton University, NJ, USA
daniel.reichman@gmail.com

Igor Shinkar
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
ishinkar@sfu.ca

Abstract

String matching is the problem of deciding whether a given n-bit string contains a given k-bit pattern. We study the complexity of this problem in three settings.

- **Communication complexity.** For small k, we provide near-optimal upper and lower bounds on the communication complexity of string matching. For large k, our bounds leave open an exponential gap; we exhibit some evidence for the existence of a better protocol.
- **Circuit complexity.** We present several upper and lower bounds on the size of circuits with threshold and DeMorgan gates solving the string matching problem. Similarly to the above, our bounds are near-optimal for small k.
- **Learning.** We consider the problem of learning a hidden pattern of length at most k relative to the classifier that assigns 1 to every string that contains the pattern. We prove optimal bounds on the VC dimension and sample complexity of this problem.

2012 ACM Subject Classification Theory of computation → Communication complexity; Theory of computation → Circuit complexity; Theory of computation → Boolean function learning

Keywords and phrases string matching, communication complexity, circuit complexity, PAC learning

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.56

Category RANDOM

Related Version All proofs are deferred to the full version of the paper available at `https://arxiv.org/abs/1709.02034`.

Funding *Alexander Golovnev*: supported by a Rabin Postdoctoral Fellowship.
Mika Göös: supported by the NSF grant No. CCF-1412958.
Igor Shinkar: supported by NSERC discovery grant.

Acknowledgements We thank Paweł Gawrychowski for his useful feedback and Gy. Turán for sharing [16] with us. We are also very grateful to anonymous reviewers for their insightful comments.

1 Introduction

One of the most fundamental and frequently encountered tasks by minds and machines is that of detecting patterns in perceptual inputs. A basic example is the *string matching* problem, where given a string $x \in \{0,1\}^n$ and a pattern $y \in \{0,1\}^k$, $k \leq n$, the goal is to


© Alexander Golovnev, Mika Göös, Daniel Reichman, and Igor Shinkar;
licensed under Creative Commons License CC-BY



decide whether x contains y as a substring. Formally, denoting by $x[i,j]$ the bits of x in the interval $[i,j] \coloneqq \{i, i+1, \ldots, j\}$, we define a Boolean function by

$$\mathrm{SM}_{n,k}(x,y) \coloneqq 1 \qquad \text{iff} \qquad x[i,i+k-1] = y \text{ for some } i \in [n-k+1].$$

String matching is well-studied in the context of traditional algorithms: it can be computed in linear time [7, 25, 15] (with some lower bounds given by [40]). It has also been studied in more modern algorithmic frameworks such as streaming [37], sketching [3], and property testing [5]. See Section 2 for more related work.

In this work we study the $\mathrm{SM}_{n,k}$ problem in three models of computation, where it appears to have received relatively little attention.

1. *Communication complexity:* How many bits of communication are required to compute $\mathrm{SM}_{n,k}$ when the input (x,y) is adversarially split between two players?
2. *Circuit complexity:* How many gates are needed to compute $\mathrm{SM}_{n,k}$ by DeMorgan circuits (possibly in low depth)? How about threshold circuits?
3. *Learning:* How many labeled samples of strings must be observed in order to (PAC) learn a classifier assigning 1 to a string if and only if it contains a (fixed) hidden pattern y? What is the VC dimension of this problem?

1.1 Results: Communication Complexity

We first show bounds on the randomized two-party communication complexity of $\mathrm{SM}_{n,k}$. (For standard textbooks on communication complexity, see [26, 22].) The only related prior work we are aware of is Bar-Yossef et al. [3] who studied the *one-way* communication complexity of string matching; our focus is on *two-way* communication. Our bounds are near-optimal for small k, but for large $k \geq \Omega(n)$, we leave open a mysterious exponential gap. Our protocols work regardless of how the input bits (x,y) are bipartitioned between the players, whereas our lower bound is proved relative to some fixed hard partition.

▶ **Theorem 1** (Communication Complexity). *For the* $\mathrm{SM}_{n,k}(x,y)$ *problem:*

- ***Upper bound:*** *Under any bipartition of the input bits, there is a protocol of cost*

  Deterministic: $O(\log k \cdot n/k)$ *if* $k \leq \sqrt{n}$ *;*
  Randomized: $O(\log n \cdot \sqrt{n})$ *if* $k \geq \sqrt{n}$*.*

- ***Lower bound:*** *For* $k \geq 2$ *there is a bipartition of the input bits such that every randomized protocol requires* $\Omega(\log\log k \cdot n/k)$ *bits of communication, even for the fixed pattern* $y = 1^k$*.*

▶ Remark 2. Note that the most natural bipartition, where Alice gets x and Bob gets y, is easy. Indeed, for such partition there is a randomized $O(\log n)$-bit protocol, where Bob sends to Alice a hash of y, and Alice compares it with the hashes of the substrings $x[1,k]$, $x[2,k+1],\ldots, x[n-k+1,n]$. Under this bipartition, by setting $k=n$, one can also recover the usual *equality* problem, which is well-known to have deterministic communication complexity $\Omega(n)$. This explains why nontrivial protocols for large k need randomness.

A better protocol?

For simplicity of discussion, consider the case $k = n/2$.

What is the randomized communication complexity of $\mathrm{SM}_{n,n/2}$*?*

Our bounds, $\Omega(\log\log n)$ and $O(\log n \cdot \sqrt{n})$, leave open a huge gap. We conjecture that the answer is closer to the lower bound. As formal evidence we show that problems closely related to $\mathrm{SM}_{n,n/2}$ admit efficient "unambiguous randomized" (aka U·BPP) communication protocols.

A classic result [51] says that any "unambiguous deterministic" (aka $\mathsf{U{\cdot}P}$) protocol can be efficiently simulated by a deterministic one, that is, $\mathsf{U{\cdot}P} = \mathsf{P}$ in communication complexity. A randomized analogue of this, $\mathsf{U{\cdot}BPP} = \mathsf{BPP}$, turns out to be false as a consequence of the recent breakthrough of Chattopadhyay et al. [9]. One can nevertheless interpret our $\mathsf{U} \cdot \mathsf{BPP}$ protocols as evidence for the existence of improved randomized protocols.

Techniques

Our lower bound in Theorem 1 requires proving a tight randomized lower bound for composed functions of the form $\mathrm{OR} \circ \mathrm{GT}$ (where GT is the *greater-than* function), which answers a question of Watson [50]. We observe that the lower bound follows by a minor modification of existing *information complexity* techniques [8]. For upper bounds, the role of periods in strings plays a central role (Section 3.1). We go on to discuss a natural *period finding* problem, and conjecture that it is easy for randomized protocols. See Section 3.4 for details.

The communication complexity and circuit complexity of $\mathrm{SM}_{n,k}$ are related. As we soon demonstrate, our study of the communication complexity of $\mathrm{SM}_{n,k}$ results with circuit lower bounds for threshold circuits computing $\mathrm{SM}_{n,k}$.

1.2 Results: Circuit Complexity

Threshold circuits

A threshold circuit is a circuit whose gates compute *linear threshold functions* (LTFs). Recall that an LTF outputs 1 on an m-bit input x if and only if $\sum_{i\in[m]} a_i x_i \geq \theta$ for some fixed coefficient vector $a \in \mathbb{R}^m$, and $\theta \in \mathbb{R}$. The study of threshold circuits is often motivated by its connection to neural networks [17, 36, 35, 32, 33]. The case of *low-depth* threshold circuits is also interesting. In particular, one line of work [47, 38, 46] has focused on efficient low-depth threshold implementations of arithmetic primitives (addition, comparison, multiplication). As for lower bounds, [17] show an exponential-in-n lower bound for the *mod-2 inner-product* function against depth-2 threshold circuits of low weight (see [12] for an extension). Superlinear lower bounds on the number of gates of arbitrary depth-2 as well as low-weight depth-3 threshold circuits were proven recently by Kane and Williams [24].

It is important that we measure the size of a threshold circuit as the *number of gates* (excluding inputs), in which case even superconstant lower bounds are meaningful. For example, it is easy to implement the equality function (namely $\mathrm{SM}_{n,n}$) using three threshold gates (albeit, with exponential weights). Thus, in contrast to the case of bounded fanin circuits, proving linear or even nonconstant lower bounds on the number of gates is not straightforward. Indeed, there are few explicit examples of functions with superconstant lower bounds [16], and proving them is considered challenging [43]. Indeed, Jukna [22] writes "even proving non-constant lower bounds ... is a nontrivial task".

We show that $\mathrm{SM}_{n,k}$ admits a linear-size implementation at low depth. Thereafter we focus on its fine-grained complexity, seeking to establish lower bounds as close to $\Omega(n)$ as possible.

▶ **Theorem 3** (Threshold circuits). *For the* $\mathrm{SM}_{n,k}(x,y)$ *problem:*

- ***Upper bound:*** *There is a depth-2 threshold circuit of size* $O(n-k)$.
- ***Lower bound for unbounded depth:*** *Any threshold circuit must be of size*

$\Omega(\frac{n \log\log k}{k \log n})$ *if* $k > 1$;
$\Omega(\sqrt{n/k})$ *if* $k \geq 2.1 \cdot \log n$.

The second lower bound is stronger than the first one in the regime $k = \Omega(n \cdot (\frac{\log\log n}{\log n})^2)$. We note that for $k \leq \text{polylog}(n)$, we have nearly linear lower bounds for unbounded-depth threshold circuits computing $\text{SM}_{n,k}$. We stress that there are no restrictions on the weights of the threshold gates in these lower bounds. We are not able to prove $\Omega(n)$ lower bounds even for depth-2 threshold circuits. Proving such lower bounds (or constructing a threshold circuit of size $o(n)$) remains open. We can prove strong lower bounds for depth-2 circuits in some special cases (see Section 4.3).

Techniques

In Section 4.2 we obtain lower bounds for threshold circuits from the lower bounds on communication complexity of $\text{SM}_{n,k}$ using a connection between threshold complexity and circuit complexity outlined by [34]. We also prove lower bounds for threshold circuits by reducing the problem of computing a "sparse hard" function to computing $\text{SM}_{n,k}$. Perhaps surprisingly, we show that the string matching problem can encode a truth table of an arbitrary sparse (few preimages of 1) Boolean function.

DeMorgan circuits

We consider usual DeMorgan circuits (AND, OR, NOT gates) of *unbounded fan-in* and show upper and lower bounds on the circuit complexity of $\text{SM}_{n,k}$. We emphasize again that we measure the size of a circuit as the *number of gates* (excluding inputs). For example, the n-bit AND can be computed with a circuit of size 1.

We start by analyzing the case of low-depth circuits.

▶ **Theorem 4** (Depth-2 DeMorgan circuits). *For the* $\text{SM}_{n,k}(x, y)$ *problem:*

- ***Depth-2 upper bound:*** *There is a depth-2 DeMorgan circuit of size* $O(n \cdot 2^k)$.
- ***Depth-2 lower bound:*** *Any depth-2 DeMorgan circuit must be of size*

$$\begin{array}{ll} \Omega(n \cdot 2^k) & \text{if } 1 < k \leq \sqrt{n}\ ; \\ \Omega(2^{2\sqrt{n-k+1}}) & \text{if } k \geq \sqrt{n}. \end{array}$$

For $k \leq \sqrt{n}$, our depth-2 results are optimal (up to a constant factor). For large k, say $k = n/2$, there is (similarly as for communication) a huge gap in our bounds: $2^{\Omega(\sqrt{n})}$ versus $2^{O(n)}$. We do not know what bound to conjecture here as the correct answer.

For DeMorgan circuits, the celebrated Håstad's switching lemma [19] established exponential lower bounds for bounded depth circuits computing explicit functions (e.g., majority, parity). We note that in contrast to the parity function, the string matching function admits a polynomial size circuit of depth 3. It is unclear (to us) how to leverage known tools for proving lower bounds for small depth circuits (such as the switching lemma) towards proving super linear lower bounds for small depth DeMorgan circuits computing $\text{SM}_{n,k}$. Whether the string matching problem can be computed by a depth 3 (or even unrestricted) DeMorgan circuit of size $O(n)$ remains open.

Next, we prove that the circuit complexity of $\text{SM}_{n,k}$ for general DeMorgan circuits (unrestricted depth and fan-in) must be $\Omega(n)$. We also include a relatively straightforward upper bound (which may have been discovered before; [14] claims an upper bound $O(n \log^2 n)$ without a proof).

▶ **Theorem 5** (General DeMorgan circuits). *For the* $\text{SM}_{n,k}(x, y)$ *problem:*

- ***Upper bound:*** *There is a DeMorgan circuit of size* $O(nk)$ *and depth* 3.
- ***Lower bound:*** *Any DeMorgan circuit must be of size at least* $n/2$.

Techniques

We prove the lower bound on DNF by exhibiting an explicit set of inputs to $\mathrm{SM}_{n,k}$ each of which requires a separate clause in any DNF. Our lower bound for CNF involves estimating the size of maxterms of $\mathrm{SM}_{n,k}$. For the lower bound against circuits of unrestricted depth, we adjust the gate elimination technique to the case of unbounded fan-in circuits. See Section 5 for details.

1.3 Results: Learning

Finally, we seek to understand the sample complexity of PAC-learning the string matching function $\mathrm{SM}_{n,\ell}(x,\sigma)$, where x is an arbitrary string of length n and σ is a *fixed* pattern of length $\ell \leq k$. Towards this goal we prove (almost) tight bounds on the VC dimension of the class of these functions. The VC dimension essentially determines the sample complexity needed to learn the pattern σ from a set of i.i.d. samples in the PAC learning framework. We formalize these notions below.

Let Σ be a fixed finite alphabet of size $|\Sigma| \geq 2$.[1] By Σ^n we denote the set of strings over Σ of length n, and by $\Sigma^{\leq k}$ we denote the set of strings of length at most k. We study the VC dimension of the class of functions, where each function is identified with a pattern of length at most k, and outputs 1 only on the strings containing this pattern. Recall that the length of the pattern $k = k(n) \leq n$ can be a function of n. We now define the set of functions we wish to learn:

▶ **Definition 6.** *For a fixed* finite *alphabet Σ and an integer $k > 0$, let us define the class of Boolean functions $\mathcal{H}_{k,\Sigma}$ over Σ^n as follows. Every function $h_\sigma \in \mathcal{H}_{k,\Sigma}$ is parameterized by a pattern $\sigma \in \Sigma^{\leq k}$ of length at most k. Hence, $|\mathcal{H}_{k,\Sigma}| = \frac{|\Sigma|^{k+1}-1}{|\Sigma|-1}$. For a string $s \in \Sigma^n$, $h_\sigma(s) = 1$ if and only if s contains σ as a substring.*

To analyze the sample complexity required to learn a function from $\mathcal{H}_{k,\Sigma}$ we first define *VC dimension*.

▶ **Definition 7.** *Let $\mathcal{F}$ be a class of functions from a set D to $\{0,1\}$, and let $S \subseteq D$. A* dichotomy *of S is one of the possible labellings of the points of S using a function from $\mathcal{F}$. S is* shattered *by $\mathcal{F}$ if $\mathcal{F}$ realizes all $2^{|S|}$ dichotomies of S. The* VC dimension *of $\mathcal{F}$, $\mathrm{VC}(\mathcal{F})$, is the size of the largest set S shattered by $\mathcal{F}$.*

In particular, $\mathrm{VC}(\mathcal{H}_{k,\Sigma}) = d$ if and only if there is a set S of d strings of length n such that for every $S' \subseteq S$, there exists a pattern P_S of length at most k occurring in all the strings in S' and not occurring in all the strings in $S \setminus S'$.

A class of functions $\mathcal{F}$ is PAC-learnable[2] with accuracy ε and confidence $1-\delta$ in $\Theta\left(\frac{\mathrm{VC}(\mathcal{F})+\log(1/\delta)}{\varepsilon}\right)$ samples [6, 11, 18], and is agnostic PAC-learnable in $\Theta\left(\frac{\mathrm{VC}(\mathcal{F})+\log(1/\delta)}{\varepsilon^2}\right)$ samples [2, 44]. Thus, tight bounds on the VC dimension of a class of functions give tight bounds on its sample complexity.

Our main result is a tight bound on the VC dimension of $\mathcal{H}_{k,\Sigma}$ (up to low order terms). That is:

[1] In contrast to the circuit and communication setting, for the learning problem we consider nonbinary alphabets.

[2] For a precise definition of PAC learning, see Definition 36.

▶ **Theorem 8.** *Let Σ be a finite alphabet of size $|\Sigma| \geq 2$, then*

$$\mathrm{VC}(\mathcal{H}_{k,\Sigma}) = \min(\log|\Sigma|(k - O(\log k)), \log n + O(\log\log n)) \ .$$

It follows that the sample complexity of learning patterns is $O(\log n)$. We also show that there are efficient polynomial time algorithms solving this learning problem. See Corollary 37 for details.

Techniques

We prove our upper bound on the VC dimension by a double counting argument. This argument uses Sperner families to show that shattering implies a "large" family of non-overlapping patterns, which, on the other hand, is constrained by the length n of the strings that we shatter. The lower bound is materialized by the idea to have 2^d patterns $P = \{p_0 \dots p_{2^d-1}\}$ and d strings such that the ith string is a concatenation of all patterns with the binary expansion of their index having the ith bit equal 1. We construct a family of patterns T with the property that for any pair of distinct strings $\alpha, \beta \in T$, their concatenation $\alpha\beta$ does not contain a string $\gamma \in T, \gamma \neq \alpha, \beta$. Using this family (with some additional technical requirements) we are able to show that P shatters a set of d strings implying our lower bound on the VC dimension.

2 More related work

Circuit complexity

Upper bounds on the circuit complexity of 2D image matching problem under projective transformations was studied in [42]. In this problem, which is considerably more complicated than the pattern matching problems we study, the goal is to find a projective transformation f such that $f(A)$ "resembles"[3] B for two images A, B. Here, images are 2D square arrays of dimension n containing discrete values (colors). In particular, it is proven that this image matching problem is in TC^1 (it admits a threshold circuit of polynomial size and logarithmic depth in n). These results concern a different problem than the string matching considered here, and do not seem to imply the upper bounds we obtain for circuits solving the string matching problem.

The idea to lower bound the circuit complexity of Boolean functions that arise in feature detection was studied in [29, 30]. These works assumed a setting with two types of features, a and b, with detectors corresponding to the two types situated on a 1D or 2D grid. The binary outputs of these features are represented by an array of n positions: $a_1, ..., a_n$ (where $a_i = 1$ if the feature a is detected in position i, and $a_i = 0$ otherwise) and an array $b_1, ..., b_n$ which is analogously defined with respect to b. The Boolean function P^n_{LR} outputs 1 if there exist i, j with $i < j$ such that $a_i = b_j = 1$, and 0 otherwise. This function is advocated in [30] as a simple example of a detection problem in vision that requires to identify spatial relationship among features. It is shown that this problem can be solved by $O(\log n)$ threshold gates. A 2-dimensional analogue where the indices $i = (i_1, i_2)$ and $j = (j_1, j_2)$ represent two-dimensional coordinates and one is interested whether there exist indices i and j such that $a_i = b_j = 1$ and j is above and to the right of the location i is studied in [30]. Recently, the two-dimensional version was studied in [48] where a $O(\sqrt{n})$-gate threshold implementation

[3] We refer to [42] for the precise definition of distance used there.

was given along with a lower bound of $\Omega(\sqrt{n/\log n})$ for the size of any threshold circuit for this problem. We remark that the problem studied in [29, 30, 48] is different from ours, and different proof ideas are needed for establishing lower bounds in our setting.

Learning patterns

The language of all strings (of arbitrary length) containing a fixed pattern is regular and can be recognized by a finite automaton. There is a large literature on learning finite automata (e.g., [1, 13, 41]). This literature is mostly concerned with various active learning models and it does not imply our bounds on the sample complexity of learning $\mathcal{H}_{k,\Sigma}$.

Motivated by computer vision applications, several works have considered the notion of *visual concepts*: namely a set of shapes that can be used to classify images in the PAC-learning framework [27, 45]. Their main idea is that occurrences of shapes (such as lines, squares etc.) in images can be used to classify images and that furthermore the representational class of DNF's can represent occurrences of shapes in images. For example, it is easy to represent the occurrence of a fixed pattern of length k in a string of size n as a DNF with $n-k$ clauses (see e.g., Lemma 24). We note that these works do not study the VC dimension of our pattern matching problems (or VC bounds in general). We also observe that no polynomial algorithm is known for learning DNF's and that there is some evidence that the problem of learning DNF is intractable [10]. Hence the result in [27, 45] do not imply that our pattern learning problem (represented as a DNF) can be done in polynomial time.

3 Communication Complexity

In this section we prove Theorem 1, and also discuss the possibility of a better upper bound.

▶ **Theorem 1** (Communication Complexity). *For the* $\mathrm{SM}_{n,k}(x,y)$ *problem:*

- ***Upper bound:*** *Under any bipartition of the input bits, there is a protocol of cost*

  Deterministic: $O(\log k \cdot n/k)$ *if* $k \le \sqrt{n}$;
  Randomized: $O(\log n \cdot \sqrt{n})$ *if* $k \ge \sqrt{n}$.

- ***Lower bound:*** *For* $k \ge 2$ *there is a bipartition of the input bits such that every randomized protocol requires* $\Omega(\log\log k \cdot n/k)$ *bits of communication, even for the fixed pattern* $y = 1^k$.

3.1 Periods in strings

We say a string $x \in \{0,1\}^n$ has *period* $p \in \{0,1\}^i$ *of order* i if x is a prefix of a high enough power p^m (for some $m \ge 1$). Equivalently, x has a period of order i iff $x[i{+}1, n] = x[1, n{-}i{-}1]$. A classic lemma characterizes the orders of short periods in a string.

▶ **Lemma 9** ([31]). *If* x *has periods of orders* i, j, $i + j \le |x|$, *then there is one of order* $\gcd(i,j)$.

In particular, all periods of order $\le n/2$ are powers of some *primitive period* (shortest period of order $\le n/2$). It is natural to ask: how many bits of communication are required to decide whether a string has a primitive period? We will discuss this in Section 3.4.

3.2 Upper bound

We start by describing an $O(\log k \cdot n/k)$-bit deterministic protocol for $\mathrm{SM}_{n,k}$ assuming the pattern y is fixed (known to both players). This immediately gives a protocol of cost $O(k + \log k \cdot n/k)$ when y is *not* fixed: Alice and Bob simply exchange all bits of the k-bit pattern and then run the protocol that assumes y is fixed. When $k \le \sqrt{n}$ this yields the first upper bound claimed in Theorem 1.

▶ **Lemma 10.** *For every fixed pattern $y \in \{0,1\}^k$ the function $x \mapsto \mathrm{SM}_{n,k}(x,y)$ admits a deterministic protocol of cost $O(\log k \cdot n/k)$ under any bipartition of the input x.*

Next we supply the protocol for the second upper bound in Theorem 1.

▶ **Lemma 11.** *For $k \geq \sqrt{n}$ the function $\mathrm{SM}_{n,k}$ admits a randomized protocol of cost $O(\log n \cdot \sqrt{n})$ under any bipartition of the input (x,y).*

▶ Remark 12. For $k \geq \sqrt{n \log n}$ the above protocol can be optimized to have cost $O(\sqrt{n \log n})$. Namely, consider a prefix p (and intervals) of length $\Theta(\sqrt{n \log n})$ rather than $\Theta(\sqrt{n})$.

3.3 Lower bound

Next we prove a lower bound of $\Omega(\log \log k \cdot n/k)$, for every $k \leq n$, on the randomized communication complexity of $\mathrm{SM}_{n,k}$. As a warm-up, we first observe that a reduction from the ubiquitous set-disjointness function yields a randomized lower bound of $\Omega(n/k)$ for $\mathrm{SM}_{n,k}$. We then show how to improve this by a factor of $\log \log k$.

Recall that in the m-bit set-disjointness problem, Alice is given $a \in \{0,1\}^m$, Bob is given $b \in \{0,1\}^m$, and their goal is to compute $\mathrm{Disj}_m(a,b) \coloneqq (\mathrm{OR}_m \circ \mathrm{AND}_2)(a,b) = \bigvee_{i \in [m]} (a_i \wedge b_i)$. It is well known that this function has communication complexity $\Omega(m)$ even against randomized protocols [23, 39, 4].

▶ **Observation 13.** $\mathrm{Disj}_{\Omega(n/k)}$ *reduces to* $\mathrm{SM}_{n,k}$ *(under some bipartition of input bits).*

To improve the above, we give a reduction from a slightly harder function, $\mathrm{OR}_m \circ \mathrm{GT}_\ell \colon [\ell]^m \times [\ell]^m \to \{0,1\}$, which maps $(a,b) \mapsto \bigvee_{i \in [m]} \mathrm{GT}(a_i, b_i)$ where $\mathrm{GT}_\ell \colon [\ell] \times [\ell] \to \{0,1\}$ is the *greater-than* function given by $\mathrm{GT}_\ell(a,b) \coloneqq 1$ iff $a \geq b$. The claimed lower bound $\Omega(\log \log k \cdot n/k)$ for $\mathrm{SM}_{n,k}$ follows from the following two lemmas. As mentioned in the introduction, Lemma 15 was conjectured by [50].

▶ **Lemma 14.** $\mathrm{OR}_{\Omega(n/k)} \circ \mathrm{GT}_{\Omega(k)}$ *reduces to* $\mathrm{SM}_{n,k}$ *(under some bipartition of input bits).*

▶ **Lemma 15.** $\mathrm{OR}_m \circ \mathrm{GT}_\ell$ *has randomized communication complexity $\Omega(m \cdot \log \log \ell)$ for any m, ℓ.*

3.4 A better protocol?

As bonus results, we give some evidence for the existence of an improved randomized protocol for $\mathrm{SM}_{n,k}$ when k is large. We first define what *unambiguous randomized* (aka $\mathsf{U} \cdot \mathsf{BPP}$, or unambiguous Merlin–Arthur) protocols are; they generalize the notion of unambiguous deterministic protocols (aka $\mathsf{U} \cdot \mathsf{P}$) introduced by Yannakakis [51].

▶ **Definition 16** ($\mathsf{U} \cdot \mathsf{BPP}$ protocols)**.** *An* unambiguous randomized *protocol Π computes a function $F(x,y)$ as follows. In the first phase the players nondeterministically guess a witness string $z \in \{0,1\}^{c_1}$, and then in the second phase they run a randomized (error $\leq 1/3$) protocol of cost c_2 to decide whether to accept the witness z. The correctness requirement is that for every $(x,y) \in F^{-1}(1)$ there needs to be a unique witness that is accepted; for every $(x,y) \in F^{-1}(0)$ no witness should be accepted. The cost of Π is defined as $c_1 + c_2$.*

Unambiguous randomized protocols have not been studied before in communication complexity. However, the recent breakthrough of Chattopadhyay et al. [9] (who disproved the log-approximate-rank conjecture of [28]) is closely related. It is not hard to see that the function $F(x,y)$ they study (of the form Sink$\circ$XOR) admits an $O(\log n)$-cost $\mathsf{U} \cdot \mathsf{BPP}$ protocol.

The authors proved that the usual randomized (aka BPP) communication complexity of F is high, $n^{\Omega(1)}$. Consequently, there is no generic simulation of a $\mathsf{U}\cdot\mathsf{BPP}$ protocol by a BPP protocol. By contrast, Yannakakis [51, Lemma 1] showed that $\mathsf{U}\cdot\mathsf{P}$ protocols can be made deterministic efficiently.

Our first bonus result is an efficient $\mathsf{U}\cdot\mathsf{BPP}$ protocol for determining if a given string has a primitive period. We do not know whether there is an efficient randomized protocol.

▶ **Lemma 17.** *Suppose the bits of $x \in \{0,1\}^n$ are split between two players. There is an $\mathsf{U}\cdot\mathsf{BPP}$ protocol of cost $O(\log^2 n)$ for deciding whether x has a primitive period (and to compute its order).*

If we let R_{pf} denote the randomized communication complexity of the above period finding problem, then we can interpret Lemma 17 as evidence that $R_{\mathrm{pf}} \leq \mathrm{polylog}(n)$. Assuming period finding is indeed easy, we can then provide similar evidence for the easiness of $\mathrm{SM}_{n,k}$ for large k.

▶ **Lemma 18.** $\mathrm{SM}_{n,0.9n}$ *admits an $\mathsf{U}\cdot\mathsf{BPP}$ protocol of cost $O(\log n) + R_{\mathrm{pf}}$.*

4 Threshold Circuits

In this section we prove Theorem 3.

▶ **Theorem 3** (Threshold circuits). *For the $\mathrm{SM}_{n,k}(x,y)$ problem:*

- ***Upper bound:*** *There is a depth-2 threshold circuit of size $O(n-k)$.*
- ***Lower bound for unbounded depth:*** *Any threshold circuit must be of size*

$$\begin{array}{ll} \Omega\left(\frac{n \log\log k}{k \log n}\right) & \text{if } k > 1; \\ \Omega\left(\sqrt{n/k}\right) & \text{if } k \geq 2.1 \cdot \log n. \end{array}$$

In Section 4.1 we prove the upper bound, in Section 4.2 we give the lower bounds. Finally, in Section 4.3 we study the complexity of $\mathrm{SM}_{n,k}$ in the models of restricted threshold circuits.

4.1 Upper bound

We start with a construction giving the upper bound of Theorem 3.

▶ **Lemma 19.** *There is a depth-2 threshold circuit of size $O(n-k)$ computing $\mathrm{SM}_{n,k}$.*

4.2 Lower bounds

In order to prove the first lower bound of $\Omega(\frac{n \log\log k}{k \log n})$ we use the classical result on communication complexity of threshold gates [34], and the lower bound on communication complexity of $\mathrm{SM}_{n,k}$ from Theorem 1.

Nisan and Safra [34] proved that for *any* bipartition of the n input bits, the ϵ-error randomized communication complexity of a threshold gate (with arbitrary weights) has communication complexity $O(\log n/\epsilon)$. From this they concluded that for any function f, a lower bound of m on the randomized communication complexity for *some* bipartition of the input implies a lower bound of $\Omega(m/\log n)$ on the threhold complexity of f. Now the lower bound of $\Omega(n \log\log k/k)$ from Theorem 1 implies the lower bound of $\Omega(\frac{n \log\log k}{k \log n})$ on the size of an unbounded depth threshold circuit computing $\mathrm{SM}_{n,k}$.

Below we prove the second lower bound stated in Theorem 3. The lower bound is shown via a reduction from a hard function $f\colon \{0,1\}^{k/2-1} \to \{0,1\}$ which has n/k preimages of 1: $|f^{-1}(1)| = n/k$. First, we prove the desired lower bound for the case where k is even

and n is a multiple of k. In the end of this section we explain how to adjust the proof to the remaining cases. Let ℓ and t be integers such that $k = 2\ell + 2$ and $n = t \cdot k$. Let $F_{\ell,t} = \{f \colon \{0,1\}^\ell \to \{0,1\} : |f^{-1}(1)| = t\}$ be the class of Boolean functions of ℓ inputs which have exactly t preimages of 1.

We prove this lower bound via a reduction from a hard function $f \in F_{\ell,t}$. Specifically, we show that if $\mathrm{SM}_{n,k}$ can be solved by a circuit of size s, then every function $f \in F_{\ell,t}$ also has a circuit of size s computing it. Then, we show that there are functions in $F_{\ell,t}$ that require large threshold circuits, which implies the corresponding lower bound for the $\mathrm{SM}_{n,k}$ function.

The reduction

Given a string $a \in \{0,1\}^\ell$ define $dup(a) \in \{0,1\}^k$ to be the string obtained from a by repeating each bit of a twice, and concatenating it with 01 in the end. (Note that $2\ell + 2 = k$ by the choice of ℓ). For example $dup(010) = 00110001$.

▶ **Observation 20.** *Given a function $f \in F_{\ell,t}$ define $x_f \in \{0,1\}^{tk}$ to be the concatenation of $dup(a)$ for all $a \in f^{-1}(1)$ in the lexicographic order on $\{0,1\}^\ell$. Note that $|x_f| = tk = n$. Then, for any $y \in \{0,1\}^\ell$ it holds that $f(y) = 1$ if and only if $\mathrm{SM}_{n,k}(x_f, dup(y)) = 1$.*

Indeed, it is immediate to see that if $f(y) = 1$ then $\mathrm{SM}_{n,k}(x_f, dup(y)) = 1$. Duplicating every bit in a and adding 01 to the end of the resulting pattern are done to ensure that if $f(y) = 0$ there will not be a copy of $dup(y)$ in x_f.

Given the observation above, it is not difficult to see that any lower bound on the size of a circuit computing $f \in F_{\ell,t}$ implies a lower bound on $\mathrm{SM}_{n,k}$.

▶ **Proposition 21.** *Let C be a threshold circuit computing $\mathrm{SM}_{n,k}$. Then for every $f \in F_{\ell,t}$, there exists a threshold circuit C' computing f such that $|C'| \leq |C|$.*

In order to complete the proof of Theorem 3, we need to show that there exists a function $f \in F_{\ell,t}$ that requires large threshold circuits. For this, we compare the number of small threshold circuits (see, for example, [22, 24]) with the number of functions in $F_{\ell,t}$.

▶ **Proposition 22.** *Let $\ell \in \mathbb{N}$ be sufficiently large, and let $t \in \mathbb{N}$. There exists a function $f \in F_{\ell,t}$ such that any threshold circuit (with no restrictions on its depth) computing f must be of size at least $\Omega(\sqrt{t - t\log t/\ell})$.*

We now derive the desired lower bound on the size of threshold circuits computing the string matching function. Plugging in $k = 2\ell + 2$ and $n = tk$, we get the lower bound of $s \geq \Omega\big(\sqrt{\frac{n}{k} - \frac{2n}{k^2} \cdot \log(\frac{n}{k})}\,\big) = \Omega(\sqrt{\frac{n}{k}})$ assuming $k \geq \Omega(\log n)$.

Now we describe how this proof can be adopted for the case when n is not a multiple of k and the case of odd k. First, in order to handle the case of pattern of *odd* length, one can add the string 010 (instead of 01) to the end of $dup(a)$. If n is not a multiple of k, then in the reduction above we can pad the string x_f with zeros in the end, and the reduction still satisfies the property that $f(y) = 1$ if and only if $\mathrm{SM}_{n,k}(x_f, dup(y)) = 1$ as in Observation 20, and the same lower bound holds (up to a constant factor in the asymptotics).

4.3 Depth-2 Circuits

In Theorem 23 we prove lower bounds for some restricted classes of depth-2 circuits computing $\mathrm{SM}_{n,k}$. These results should be contrasted with the upper bounds of Theorem 3 and Theorem 5. Namely, there exists an LTF$\circ$LTF circuit of size $O(n-k)$ and an OR$\circ$AND$\circ$OR circuit of size $O(nk)$ computing $\mathrm{SM}_{n,k}$.

We recall a few definitions. Let ELTF denote the class of *exact* threshold functions (that is, the functions which output 1 on an m-bit input x if and only if $\sum_{i\in[m]} a_i x_i = \theta$ for some fixed coefficient vector $a \in \mathbb{R}^m$, and $\theta \in \mathbb{R}$). Similarly, EMAJ denotes the class of exact majorities which output 1 if and only if the sum of their m Boolean inputs is exactly $m/2$. By SYM we denote the class of all symmetric Boolean functions. For two classes of functions $\mathcal{C}_1$ and $\mathcal{C}_2$, by $\mathcal{C}_1 \circ \mathcal{C}_2$ we denote the class of depth-2 circuits where the output gate is from $\mathcal{C}_1$ and the gates of the first layer are from $\mathcal{C}_2$. For a class of circuits $\mathcal{C}$ and a function f, be $\mathcal{C}(f)$ we denote the minimal size of a circuit from $\mathcal{C}$ computing f.

In proving lower bounds for $\mathrm{SM}_{n,k}$ a simple yet useful property is that Observation 13 can be applied to circuits as well. This allows to reduce the disjointness problem to string matching, and get lower bounds for $\mathrm{SM}_{n,k}$ via known circuit lower bounds for disjointness. The point is that a circuit C with strings of length roughly mk for $\mathrm{SM}_{n,k}$ (and patterns of length k) can be used to solve disjointness on strings of length m by feeding C with the string $x := a_1 b_1 1^{k-2} 0 a_2 b_2 1^{k-2} 0 \dots a_n b_n 1^{k-2} 0$ and the pattern $y = 1^k$. Hence a lower bound of $s(n)$ for circuits computing disjointness implies a lower bound of $\Omega(s(n/k))$ for circuits computing $\mathrm{SM}_{n,k}$.

▶ **Theorem 23.** *For every* $1 < k \le n$,
1. $\mathrm{OR} \circ \mathrm{LTF}(\mathrm{SM}_{n,k}) \ge \Omega(n-k)$;
2. $\mathrm{AND} \circ \mathrm{LTF}(\mathrm{SM}_{n,k}) \ge 2^{\Omega(n/k)}$;
3. $\mathrm{AND} \circ \mathrm{OR} \circ \mathrm{XOR}(\mathrm{SM}_{n,k}) \ge 2^{\Omega(n/k)}$;
4. $\mathrm{ELTF} \circ \mathrm{SYM}(\mathrm{SM}_{n,k}) \ge 2^{\Omega(n/k)}$;
5. $\mathrm{EMAJ} \circ \mathrm{ELTF}(\mathrm{SM}_{n,k}) \ge 2^{\Omega(n/k)}$.

5 DeMorgan Circuits

In this section we prove Theorem 4 and Theorem 5.

▶ **Theorem 4** (Depth-2 DeMorgan circuits). *For the* $\mathrm{SM}_{n,k}(x,y)$ *problem:*
- ***Depth-2 upper bound:*** *There is a depth-2 DeMorgan circuit of size* $O(n \cdot 2^k)$.
- ***Depth-2 lower bound:*** *Any depth-2 DeMorgan circuit must be of size*

$$\begin{array}{ll} \Omega(n \cdot 2^k) & \text{if } 1 < k \le \sqrt{n}\ ; \\ \Omega(2^{2\sqrt{n-k+1}}) & \text{if } k \ge \sqrt{n}. \end{array}$$

▶ **Theorem 5** (General DeMorgan circuits). *For the* $\mathrm{SM}_{n,k}(x,y)$ *problem:*
- ***Upper bound:*** *There is a DeMorgan circuit of size* $O(nk)$ *and depth* 3.
- ***Lower bound:*** *Any DeMorgan circuit must be of size at least* $n/2$.

In Section 5.1 we give upper bounds for both theorems, in Section 5.2 we prove lower bounds for depth-2 circuits, and in Section 5.3 we provide a lower bound for the unbounded depth case.

5.1 Upper Bounds

We first give a DNF with $2^k(n-k+1)$ clauses computing $\mathrm{SM}_{n,k}$, and in Lemma 26 we will prove that this DNF is essentially optimal.

▶ **Lemma 24.** *For any* $k \le n$ *there exists a DeMorgan circuit of depth 2 and size* $(n-k+1)\cdot 2^k + 1$ *computing* $\mathrm{SM}_{n,k}$.

Now we show that already in depth 3, one can compute $\mathrm{SM}_{n,k}$ by a much smaller circuit. This Lemma is likely to have been discovered multiple times, we attribute it to folklore.

▶ **Lemma 25.** *There exists a DeMorgan circuit of depth 3 and size $O(nk)$ computing* $\mathrm{SM}_{n,k}$.

5.2 Lower bounds for depth 2

We may assume wlog that every optimal circuit of depth 2 is either a CNF or a DNF. First, we show that in the class of DNFs, the construction from Lemma 24 is optimal (up to a constant factor).

▶ **Lemma 26.**
For every $k > 2$, the DNF*-size of* $\mathrm{SM}_{n,k}$ *is at least*

$$\mathrm{DNF}(\mathrm{SM}_{n,k}) \geq 2^{k-2}(n-k+1)\ .$$

Now we will prove lower bounds for CNFs computing $\mathrm{SM}_{n,k}$. We will need the following definition.

▶ **Definition 27.** *A* maxterm *of a Boolean function f is a set of variables of f, such that some assignment to those variables makes f output 0 irrespective of the assignment to the other variables. The* width *of a maxterm is the number of variables in it.*

First we find the minimal width of maxterms of $\mathrm{SM}_{n,k}$.

▶ **Lemma 28.** *For any $k \leq n$, every maxterm of* $\mathrm{SM}_{n,k}$ *has width at least*

$$\begin{array}{ll} 2\sqrt{n-k+1} & \text{for all } k\ ; \\ k+\frac{n-k+1}{k} & \text{if } k \leq \sqrt{n-k+1}. \end{array}$$

Next we prove tight bounds on the number of non-satisfying inputs of $\mathrm{SM}_{n,k}$.

▶ **Lemma 29.** *For $k \leq n$, let Z denote the set of preimages of 0 of* $\mathrm{SM}_{n,k}$. *That is,*

$$Z = \{(x,y) \in \{0,1\}^{n+k} : \mathrm{SM}_{n,k}(x;y) = 0\}.$$

Then

$$\begin{array}{ll} |Z| = \Theta\left(2^{n+k}\right) & \text{if } k \geq \log n + 1; \\ |Z| \geq \Omega\left(2^n(1-2^{-k})^n\right) & \text{for all } k. \end{array}$$

▶ **Lemma 30.** *For every k, the* CNF*-size of* $\mathrm{SM}_{n,k}$ *is at least*

$$\begin{array}{ll} \mathrm{CNF}(\mathrm{SM}_{n,k}) \geq \Omega\left(2^{\frac{n}{10k}}\right) & \text{if } 1 < k \leq \log n + 1; \\ \mathrm{CNF}(\mathrm{SM}_{n,k}) \geq \Omega\left(2^{k+n/k}\right) & \text{if } \log n + 1 \leq k \leq \sqrt{n}; \\ \mathrm{CNF}(\mathrm{SM}_{n,k}) \geq \Omega\left(2^{2\sqrt{n-k+1}}\right) & \text{if } k \geq \sqrt{n}. \end{array}$$

Discussion

Lemma 26 and Lemma 30 together give the lower bounds of Theorem 4. We observe a curious behavior of CNFs and DNFs for $\mathrm{SM}_{n,k}$. For $k \leq \sqrt{n}$, an optimal depth-2 circuit for $\mathrm{SM}_{n,k}$ is a DNF. It can also be shown that for $k \geq n - O(\frac{n}{\log n})$, an optimal circuit is a CNF. (Indeed, in order to certify that $\mathrm{SM}_{n,k}(x,y) = 0$, it suffices to give mismatches for each of the $(n-k+1)$ shifts of the pattern y in x. This amounts to $k^{O(n-k+1)} < n \cdot 2^k$ clauses.) We leave the exact CNF complexity of $\mathrm{SM}_{n,k}$ for the regime $k > \sqrt{n}$ as an open problem. One way to prove a stronger lower bound in this regime would be to give a lower bound on the width of every maxterm. This approach does not lead to stronger lower bounds

because there exist maxterms of width $2\sqrt{n}$. To see this, consider an assignment where the first $\sqrt{n}$ characters of the pattern y are fixed to zeros, and all indices divisible by $\sqrt{n}$ in the text x are fixed to ones. While we cannot prove a stronger lower bound on the width of "most" maxterms, we know that some maxterms must have width at least $n-k+1$. Indeed, consider the text $x = 0^n$ and pattern $y = 10^{k-1}$. Every clause which outputs 0 on this pair, must assign the first $(n-k+1)$ positions of x to 0.

We remark that weaker lower bounds of $2^{\Omega(\sqrt{n/k})}$ and $2^{0.08n/k}$ on the size of CNF computing $\mathrm{SM}_{n,k}$ follow from the reduction from Disjointness in Observation 13 and the known lower bound on the depth-3 complexity of Iterated Disjointness [20] and Disjointness [21].

5.3 Lower bound for unbounded depth

Now we prove the lower bound of Theorem 5. For circuits with fan-in 2, a linear lower bound follows from the observation that $\mathrm{SM}_{n,k}$ essentially depends on all of its inputs. In the next lemma, we use an extension of the gate elimination technique to show that even in the class of DeMorgan circuits with *unbounded fan-in*, $\mathrm{SM}_{n,k}$ still requires linear size.

▶ **Lemma 31.** *For $k > 1$, any DeMorgan circuit computing $\mathrm{SM}_{n,k}$ has size at least $n/2$.*

6 Learning

6.1 VC dimension

In this section we prove Theorem 8.

▶ **Theorem 8.** *Let Σ be a finite alphabet of size $|\Sigma| \geq 2$, then*

$$\mathrm{VC}(\mathcal{H}_{k,\Sigma}) = \min(\log|\Sigma|(k - O(\log k)), \log n + O(\log\log n)) \ .$$

We begin by upper bounding the VC dimension. In the proof we will use the following folklore construction of a Sperner system.

▶ **Definition 32.** *A system $\mathcal{F}$ of subsets of $\{1,\dots,n\}$ is called a* Sperner system *if no set in $\mathcal{F}$ contains another one:*

$$\forall A, B \in \mathcal{F}\colon A \neq B \implies A \not\subseteq B \ .$$

For any n, there exists a Sperner system of size $\binom{n}{\lfloor n/2 \rfloor}$. Indeed, one can take $\mathcal{F}$ to be the family of all sets of size exactly $\lfloor n/2 \rfloor$.

▶ **Lemma 33.** *Let Σ be a finite alphabet of size $|\Sigma| \geq 2$, then*

$$\mathrm{VC}(\mathcal{H}_{k,\Sigma}) \leq \min(\lceil k\log|\Sigma| \rceil, \log n + 0.5\log\log n + 2) \ .$$

To lower bound the VC dimension of $\mathcal{H}_{k,\Sigma}$ we need the following lemma.

▶ **Lemma 34.** *Let m be an integer $m \geq 1$, and Σ be an alphabet of size $|\Sigma| \geq 2$. There exists a set T_m of at least $|\Sigma|^{m-1}$ strings from $\Sigma^{m+\lceil \log m \rceil + 2}$ with the following property. For any two distinct strings $\tau_1, \tau_2 \in T_m$, their concatenation $\tau = \tau_1 \circ \tau_2$ doesn't contain any string from $T_m \setminus \{\tau_1, \tau_2\}$ as a substring.*

▶ **Lemma 35.** *Let Σ be a finite alphabet of size $|\Sigma| \geq 2$, then*

$$\mathrm{VC}(\mathcal{H}_{k,\Sigma}) \geq \min((k - \log k - 5)\log|\Sigma|, \log n - \log\log n) \ .$$

This concludes the proof of Theorem 8.

6.2 Learning $\mathcal{H}_{k,\Sigma}$

In this section we discuss an efficient algorithm for learning the hypothesis class $\mathcal{H}_{k,\Sigma}$. For completeness we state the definition of PAC learning:

Let $\mathcal{D}$ be a distribution over Σ^n. Suppose we are trying to learn h_σ for $\sigma \in \Sigma^{\leq k}$. Given $\tau \in \Sigma^{\leq k}$, the loss of h_τ with respect to h_σ is defined as

$$L_{\mathcal{D},\sigma}(\tau) = \Pr_{x\sim D}[h_\tau(x) \neq h_\sigma(x)] \ .$$

Following the notion of PAC-learning [49, 44], we can now define what we mean by learning $\mathcal{H}_{k,\Sigma}$.

▶ **Definition 36.** *An algorithm $\mathcal{A}$ is said to PAC-learn $\mathcal{H}_{k,\Sigma}$ if for every distribution $\mathcal{D}$ over Σ^n and every $h_\sigma \in \mathcal{H}_{k,\Sigma}$ for all $\epsilon, \delta \in (0, 1/2)$ the following holds. Given $m := m(\epsilon, \delta, n, k)$ i.i.d. samples $(x_1, h_\sigma(x_1)), \ldots, (x_m, h_\sigma(x_m))$ where each x_i is sampled according to the distribution $\mathcal{D}$, $\mathcal{A}$ returns with probability at least $1 - \delta$ a function $h_\tau \in \mathcal{H}_{k,\Sigma}$ such that $L_{\mathcal{D},\sigma}(\tau) \leq \epsilon$. Here the probability is taken with respect to the m i.i.d. samples as well as the possible random choices made by the algorithm $\mathcal{A}$.*

Throughout, we refer to δ as the confidence parameter and ϵ as the accuracy parameter.

In Definition 36 we consider the *realizable* case. Namely there exists $h_\sigma \in \mathcal{H}_{k,\Sigma}$ that we want to learn. One can also consider the *agnostic* case. Consider a distribution $\mathcal{D}$ over $\Sigma^n \times \{0, 1\}$. We now define the loss of h_τ as

$$L_{\mathcal{D}}(\tau) = \Pr_{x\sim D}[h_\tau(x) \neq y] \ ,$$

namely the measure under $\mathcal{D}$ of all pairs $(x, y) \in \Sigma^n \times \{0, 1\}$ with $h_\tau(x) \neq y$ [44]. In the agnostic case we wish to find, given m i.i.d. samples $(x_1, h(x_1)), \ldots, (x_m, h(x_m))$, a pattern $\sigma' \in \Sigma^{\leq k}$ such that $L_{\mathcal{D}}(\sigma') \leq \min_\tau L_{\mathcal{D}}(\tau) + \epsilon$ (where the minimum is taken over all $\tau \in \Sigma^{\leq k}$). Thus agnostically PAC-learning generalizes the realizable case where $\min_\tau L_{\mathcal{D}}(\tau) = 0$.

Recall that a function $h_\sigma \in \mathcal{H}_{k,\Sigma}$ (parameterized by the pattern σ of length at most k) can be learned with error ϵ and confidence δ by considering $m = O(\mathrm{VC}(\mathcal{H}_{k,\Sigma}))$ samples $(x_1, h_\sigma(x_1)), \ldots, (x_m, h_\sigma(x_m))$ (where the constant in the O term depends on ϵ, δ) and following the ERM (expected risk minimization) rule: Finding σ' that minimizes the loss

$$L(h_{\sigma'}) := \frac{|\{i \in [m] : h_{\sigma'}(x_i) \neq h_\sigma(x_i)\}|}{m} \ .$$

In words, to PAC learn h_σ we simply look for a string σ' of length at most k such that the fraction of sample points that are misclassified by $h_{\sigma'}$ is minimized (the ERM rule applies both for the agnostic and realizable settings).

By Lemma 33, the number of samples needed to PAC-learn h_σ is at most $O(\log n)$ (ignoring the dependency on ϵ, δ). Clearly we can implement the ERM by considering all possible substrings of length at most k that occur in the $m = O(\log n)$ strings $x_1 \ldots x_m$ and finding the substring σ' minimizing $L(h_{\sigma'})$. The number of such substrings is at most $O(\log n \sum_{i=1}^{k}(n - k + 1)) \leq O(kn \log n)$. Since for every substring we can check whether it occurs in a string of length n in time $O(n)$, we can implement the ERM rule by going over every substring η of length at most k and checking for every string x_i (with $i \in [m]$) whether η occurs in x_i. By keeping track of the pattern which has minimal classification error with respect to the sample $(x_1, h_\sigma(x_1)), \ldots, (x_m, h_\sigma(x_m))$ we can thus implement the ERM rule in time $O(kn^2 \log^2 n)$.

We can do better if the number of substrings of length at most k which is upper bounded by $2|\Sigma|^k$ is smaller than $(n-k+1)\log n$. Suppose for example, that $k \leq \frac{\log n}{\log|\Sigma|}$. By Lemma 33, the VC-dimension of $\mathcal{H}_{k,\Sigma}$ is then upper bounded by $k\log|\Sigma|$. Hence in this case we can assume the number of strings m in our sample is at most $k\log|\Sigma|$, and we can implement the ERM rule in time $O(|\Sigma|^k kn\log|\Sigma|)$. When $k, |\Sigma|$ are constants independent of n we can thus learn h_σ in time $O(n)$.

We summarize this discussion with the following corollary:

▶ **Corollary 37.** *The hypothesis class $\mathcal{H}_{k,\Sigma}$ is PAC-learnable in time $O(kn^2\log^2 n)$, where the O symbol contains constants depending on ϵ, δ but not on n, k. If $k, |\Sigma|$ are constants independent of n, then $\mathcal{H}_{k,\Sigma}$ can be learned in time $O(n)$.*

References

1 Dana Angluin. Learning regular sets from queries and counterexamples. *Information and computation*, 75(2):87–106, 1987.

2 Martin Anthony and Peter L. Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 2009.

3 Ziv Bar-Yossef, T. S. Jayram, Robert Krauthgamer, and Ravi Kumar. The sketching complexity of pattern matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 261–272. Springer, 2004.

4 Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.

5 Omri Ben-Eliezer, Simon Korman, and Daniel Reichman. Deleting and testing forbidden patterns in multi-dimensional arrays. In *International Proceedings in Informatics*, volume 80. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

6 Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

7 Robert S. Boyer and J. Strother Moore. A fast string searching algorithm. *Communications of the ACM*, 20(10):762–772, 1977.

8 Mark Braverman and Omri Weinstein. A Discrepancy Lower Bound for Information Complexity. *Algorithmica*, 76(3):846–864, 2016. doi:10.1007/s00453-015-0093-8.

9 Arkadev Chattopadhyay, Nikhil Mande, and Suhail Sherif. The Log-Approximate-Rank Conjecture is False. In *Proceedings of the 51st Symposium on Theory of Computing*, 2019. To appear.

10 Amit Daniely and Shai Shalev-Shwartz. Complexity theoretic limitations on learning DNF's. In *Conference on Learning Theory*, pages 815–830, 2016.

11 Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–261, 1989.

12 Jürgen Forster, Matthias Krause, Satyanarayana V. Lokam, Rustam Mubarakzjanov, Niels Schmitt, and Hans Ulrich Simon. Relations between communication complexity, linear arrangements, and computational complexity. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 171–182. Springer, 2001.

13 Yoav Freund, Michael Kearns, Dana Ron, Ronitt Rubinfeld, Robert E Schapire, and Linda Sellie. Efficient learning of typical finite automata from random walks. *Information and Computation*, 138(1):23–48, 1997.

14 Zvi Galil. Optimal parallel algorithms for string matching. *Information and Control*, 67(1-3):144–157, 1985.

15 Zvi Galil and Joel Seiferas. Time-space-optimal string matching. *Journal of Computer and System Sciences*, 26(3):280–294, 1983.

16 Hans Dietmar Groeger and György Turán. A linear lower bound for the size of threshold circuits. *Bulletin-European Association For Theoretical Computer Science*, 50:220–220, 1993.

17 András Hajnal, Wolfgang Maass, Pavel Pudlák, Mario Szegedy, and György Turán. Threshold circuits of bounded depth. *Journal of Computer and System Sciences*, 46(2):129–154, 1993.

18 Steve Hanneke. The optimal sample complexity of PAC learning. *The Journal of Machine Learning Research*, 17(1):1319–1333, 2016.

19 Johan Håstad. *Computational Limitations of Small-depth Circuits*. MIT Press, 1987.

20 Johan Håstad, Stasys Jukna, and Pavel Pudlák. Top-down lower bounds for depth-three circuits. *Computational Complexity*, 5(2):99–112, 1995.

21 Stasys Jukna. On graph complexity. *Combinatorics, Probability and Computing*, 15(6):855–876, 2006.

22 Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

23 Bala Kalyanasundaram and Georg Schintger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992.

24 Daniel M. Kane and Ryan Williams. Super-linear gate and super-quadratic wire lower bounds for depth-two and depth-three threshold circuits. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 633–643. ACM, 2016.

25 Donald E. Knuth, James H. Morris, Jr, and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM journal on computing*, 6(2):323–350, 1977.

26 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.

27 Eyal Kushilevitz and Dan Roth. On learning visual concepts and DNF formulae. *Machine Learning*, 24(1):65–85, 1996.

28 Troy Lee and Adi Shraibman. *Lower Bounds in Communication Complexity*, volume 3. Now Publishers, 2009. doi:10.1561/0400000040.

29 Robert A. Legenstein and Wolfgang Maass. Foundations for a circuit complexity theory of sensory processing. *Advances in neural information processing systems*, pages 259–265, 2001.

30 Robert A. Legenstein and Wolfgang Maass. Neural circuits for pattern recognition with small total wire length. *Theoretical Computer Science*, 287(1):239–249, 2002.

31 R. C. Lyndon and M. P. Schützenberger. The equation $a^M = b^N c^P$ in a free group. *Michigan Mathematical Journal*, 9:289–298, 1962.

32 James Martens, Arkadev Chattopadhya, Toni Pitassi, and Richard Zemel. On the representational efficiency of restricted Boltzmann machines. In *Advances in Neural Information Processing Systems*, pages 2877–2885, 2013.

33 Saburo Muroga. Threshold logic and its application. *Wily-Interscience*, 1971.

34 Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is Eighty*, 1:301–315, 1993.

35 Ian Parberry. *Circuit complexity and neural networks*. MIT press, 1994.

36 Ian Parberry and Georg Schnitger. Parallel computation with threshold functions. *Journal of Computer and System Sciences*, 36(3):278–302, 1988.

37 Benny Porat and Ely Porat. Exact and approximate pattern matching in the streaming model. In *Foundations of Computer Science, 2009. 50th Annual IEEE Symposium on*, pages 315–323. IEEE, 2009.

38 Alexander A. Razborov. On small depth threshold circuits. In *Scandinavian Workshop on Algorithm Theory*, pages 42–52. Springer, 1992.

39 Alexander A. Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992.

40 Ronald L. Rivest. On the worst-case behavior of string-searching algorithms. *SIAM Journal on Computing*, 6(4):669–674, 1977.

41 Dana Ron and Ronitt Rubinfeld. Exactly learning automata of small cover time. *Machine Learning*, 27(1):69–96, 1997.

42 Christian Rosenke. The exact complexity of projective image matching. *Journal of Computer and System Sciences*, 82(8):1360–1387, 2016.

43 Vwani P. Roychowdhury, Alon Orlitsky, and Kai-Yeung Siu. Lower bounds on threshold and related circuits via communication complexity. *IEEE Transactions on Information Theory*, 40(2):467–474, 1994.

44 Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

45 Haim Shvaytser. Learnable and nonlearnable visual concepts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):459–466, 1990.

46 Kai-Yeung Siu and Jehoshua Bruck. On the power of threshold circuits with small weights. *SIAM Journal on Discrete Mathematics*, 4(3):423–435, 1991.

47 Kai-Yeung Siu, Jehoshua Bruck, Thomas Kailath, and Thomas Hofmeister. Depth efficient neural networks for division and related problems. *IEEE Transactions on information theory*, 39(3):946–956, 1993.

48 Kei Uchizawa, Daiki Yashima, and Xiao Zhou. Threshold Circuits for Global Patterns in 2-Dimensional Maps. In *International Workshop on Algorithms and Computation*, pages 306–316. Springer, 2015.

49 Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

50 Thomas Watson. Communication Complexity of Statistical Distance. *ACM Transactions on Computation Theory*, 10(1):2:1–2:11, 2018. doi:10.1145/3170708.

51 Mihalis Yannakakis. Expressing combinatorial optimization problems by Linear Programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991. doi:10.1016/0022-0000(91)90024-Y.

A Learning – Extensions

Infinite alphabet

So far we have been considering the case of finite alphabet Σ. For an infinite Σ the VC dimension is essentially $\log n$ for every value of $k \geq 1$. Note that the upper bound of $\mathrm{VC}(\mathcal{H}_{k,\Sigma}) \leq \log n + 0.5 \log\log n + 2$ from Lemma 33 holds even for infinite alphabets Σ. Indeed, this upper bound counts the number of different patterns which have to occur in one string and compares it to the length of the string n. In the following lemma we give a lower bound of $\log n$ for all values of $k \geq 1$.

▶ **Lemma 38.** *Let Σ be an infinite alphabet, and $k \geq 1$. Then*

$$\mathrm{VC}(\mathcal{H}_{k,\Sigma}) = (1 + o(1)) \log n \,.$$

Learning multiple patterns

In this section we make a few simple observations regarding the VC dimension of classifiers defined by the occurrences of multiple patterns. The main observation is that learning a *constant* number of patterns does not change the asymptotics of the VC dimension so long as the number of patterns is upper bounded by the length of the pattern k. Let us consider two natural classes $\mathcal{H}^{\mathrm{and}}_{k,\Sigma}$ and $\mathcal{H}^{\mathrm{or}}_{k,\Sigma}$ of multi-pattern Boolean functions over Σ^n. Each function $h^{\mathrm{and}}_{\sigma} \in \mathcal{H}^{\mathrm{and}}_{k,\Sigma}$ is parameterized by $c > 0$ patterns $\sigma = (\sigma_1, \ldots, \sigma_c) \in \left(\Sigma^{\leq k}\right)^c$. Now, for an $s \in \Sigma^n$, $h^{\mathrm{and}}_{\sigma}(s) = 1$ if and only if s contains *each* $\sigma_i, 1 \leq i \leq c$ as a substring (for brevity we omit from notation the dependence of $\mathcal{H}^{\mathrm{and}}_{k,\Sigma}$ and $\mathcal{H}^{\mathrm{or}}_{k,\Sigma}$ on c). Similarly, a function $h^{\mathrm{or}}_{\sigma} \in \mathcal{H}^{\mathrm{or}}_{k,\Sigma}$ takes the value one: $h^{\mathrm{or}}_{\sigma}(s) = 1$ if and only if s contains *at least one* σ_i as a substring. We stress that we assume that the set of patterns $\sigma_i, i \in [c]$ are distinct.

An upper bound on the VC dimension of $\mathcal{H}^{\text{and}}_{k,\Sigma}$ and $\mathcal{H}^{\text{or}}_{k,\Sigma}$ follows at once from the following Lemma proved in [6] (Lemma 3.2.3).

▶ **Lemma 39.** *Let $\mathcal{H}_1, \ldots, \mathcal{H}_c$ be classes of functions of* VC *dimension at most $\forall i$: $\text{VC}(\mathcal{H}_i) \leq d$. Let*

$$\mathcal{H}^{and} = \{f_{h_1,\ldots,h_c}(x) = h_1(x) \wedge \ldots \wedge h_c(x) \colon h_1 \in \mathcal{H}_1, \ldots, h_c \in \mathcal{H}_c\} ,$$
$$\mathcal{H}^{or} = \{f_{h_1,\ldots,h_c}(x) = h_1(x) \vee \ldots \vee h_c(x) \colon h_1 \in \mathcal{H}_1, \ldots, h_c \in \mathcal{H}_c\} .$$

Then $\text{VC}(\mathcal{H}^{and}) = O(dc \log c)$ *and* $\text{VC}(\mathcal{H}^{or}) = O(dc \log c)$.

We now turn to the lower bound. Our result here is rather modest: We show that the lower bound on the VC dimension of a single pattern also holds for $\mathcal{H}^{\text{and}}_{k,\Sigma}$ and $\mathcal{H}^{\text{or}}_{k,\Sigma}$ provided that the number c of (distinct) patterns is not too large. Let us see that the lower bounds of Lemma 35 hold for $\mathcal{H}^{\text{and}}_{k,\Sigma}$ and $\mathcal{H}^{\text{or}}_{k,\Sigma}$. Indeed, for the class $\mathcal{H}^{\text{and}}_{k,\Sigma}$, we use the construction from Lemma 35, where for every pattern σ in that construction we consider a set of k patterns $\{\sigma^1, \ldots, \sigma^k\}$. We define $\sigma^i = \sigma_1 \ldots \sigma_i$ to be the prefix of length i of σ. For example, for the pattern 11010 we take the patterns $\{1, 11, 110, 1101, 11010\}$. We remark that we obtain k distinct subpatterns of σ. Since every string from the shattered set contains σ if and only if it contains every pattern from $\{\sigma^1, \ldots, \sigma^k\}$, all dichotomies are realized by the "last" pattern $\sigma^k = \sigma$. Since $c \leq k$, we take c longest patterns $\{\sigma^{k-c+1}, \ldots, \sigma^k\}$, and our construction gives a shattered set of size

$$\text{VC}(\mathcal{H}^{\text{and}}_{k,\Sigma}) \geq \min\left(\log|\Sigma|(k - O(\log k)), \log n + O(\log\log n)\right) .$$

For the class $\mathcal{H}^{\text{or}}_{k,\Sigma}$, we can take $T'_m \subseteq T_m$ with $|T'_m| = |T_m|/2$ and shatter a set of size $d-1$. Now for every $\sigma \in T'_m$ define a c-tuple of patterns by adding to σ $c-1$ patterns in $T_m \setminus T'_m$ (where $c \leq 2^{d-1} - 1$ because $c \leq k$). Since none of the strings in the shattered set contains a pattern from $T_m \setminus T'_m$, all dichotomies are realized by the "first" pattern σ_1. Again, our construction from Lemma 35 gives a shattered set of size $\min\left(\log|\Sigma|(k - O(\log k)), \log n + O(\log\log n)\right) - 1$.

To conclude, we have proved:

▶ **Theorem 40.** *Let $1 \leq c \leq k$ be a fixed constant. Then*

$$VC(\mathcal{H}^{and}_{k,\Sigma}), VC(\mathcal{H}^{or}_{k,\Sigma}) = \Theta\left(\min\left(\log|\Sigma|(k - O(\log k)), \log n + O(\log\log n)\right)\right) .$$

Patterns of length k

One can also consider learning patterns of length *exactly* k. We consider this case separately since it seems that getting tight bounds on VC-dimension in this case is a harder task. In particular, we are not able to get tight bounds for the regime $k = n^{1-o(1)}$ and leave this as an open question.

For a fixed *finite* alphabet Σ and an integer $k > 0$, the class of functions $\mathcal{E}_{k,\Sigma}$ over Σ^n is defined as follows. Every Boolean function $h_\sigma \in \mathcal{E}_{k,\Sigma}$ is parameterized by a pattern $\sigma \in \Sigma^k$ of length exactly k. Therefore, $|\mathcal{E}_{k,\Sigma}| = |\Sigma|^k$. For a string $s \in \Sigma^n$, $h_\sigma(s) = 1$ if and only if s contains σ as a substring. We use a simple double counting argument to prove:

▶ **Lemma 41.** $\text{VC}(\mathcal{E}_{k,\Sigma}) \leq \min(k \log|\Sigma|, \log(n-k+1) + 1)$.

Now we prove the following lower bound:

▶ **Lemma 42.** *Let Σ be a finite alphabet of size $|\Sigma| \geq 2$, then*

$$\mathrm{VC}(\mathcal{E}_{k,\Sigma}) \geq \min((k - \log k - 5)\log|\Sigma|, \log n - \log k) .$$

We remark that for the case of patterns of length *at most* k, Lemma 33 and Lemma 35 give essentially tight bounds for all regimes of the parameters. Here, in the case of patterns of length *exactly* k, we have a gap between lower and upper bounds for the regime $k = n^{1-o(1)}$.

2D patterns

Our bounds for learning one dimensional strings generalize to the 2D case. Here we have an $n \times n$ image over an alphabet Σ and am $m \times m$ pattern σ where $m \leq k \leq n$. An image is classified as 1 if and only if it contains σ.

▶ **Definition 43.** *For a fixed* finite *alphabet Σ and an integer $k > 0$, let us define the class of Boolean functions $\mathcal{G}_{k,\Sigma}$ over $\Sigma^{n\times n}$ as follows. Every function $g_\sigma \in \mathcal{G}_{k,\Sigma}$ is parameterized by a square 2D pattern $\sigma \in \Sigma^{m\times m}$ of dimension $m \leq k$. For a 2D image $s \in \Sigma^{n\times n}$ of dimension n, $g_\sigma(s) = 1$ if and only if s contains σ as a consecutive sub-matrix (sub-image).*

We give tight bounds (up to low order terms) on $\mathrm{VC}(\mathcal{G}_{k,\Sigma})$. Since the proofs are very similar to the 1D case, we only sketch the arguments here.

Since $|\mathcal{G}_{k,\Sigma}| = \sum_{1\leq i\leq k} |\Sigma|^{i^2} + 1 \leq \sum_{1\leq i\leq k} |\Sigma|^{ik} + 1 < 2|\Sigma|^{k^2}$, we have that $\mathrm{VC}(\mathcal{G}_{k,\Sigma}) \leq \lceil k^2 \log|\Sigma|\rceil$. Suppose that $\mathcal{G}_{k,\Sigma}$ shatters a set of d 2D images from $\Sigma^{n\times n}$. By considering a Sperner system over $\{1,\ldots,d-1\}$ of size $D = \binom{d-1}{\lfloor (d-1)/2\rfloor}$ and adding the element d to each subset, we get a family of $D = \binom{d-1}{\lfloor (d-1)/2\rfloor}$ patterns all lying in a single $n \times n$ image such that no pattern contains another one. We have that the bottom right corners of all these patterns are distinct, and thus $\frac{2^{d-1}}{\sqrt{2(d-1)}} \leq D \leq n^2$ implying that $d \leq 2\log n + 0.5\log\log n + 3$. Hence,

$$\mathrm{VC}(\mathcal{G}_{k,\Sigma}) \leq \min(\lceil k^2 \log|\Sigma|\rceil, 2\log n + 0.5\log\log n + 3).$$

For the lower bound, the main observation is that we can generalize Lemma 34 to the two dimensional case having a set R_m of $(m + 2\lceil\log m\rceil + 2) \times (m + 2\lceil\log m\rceil + 2)$ 2D patterns of cardinality $|\Sigma|^{m^2-1}$ such that for any four distinct patterns $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ from R_m, their concatenation (fitting the four patterns into a $2(m + 2\lceil\log m\rceil + 2) \times 2(m + 2\lceil\log m\rceil + 2)$ square image in each of the 4! possible ways) does not contain any $\alpha_5 \neq \alpha_i$ for $1 \leq i \leq 4$ from R_m. We achieve this by taking all $m \times m$ templates not containing the all 0 2D square template of size $(2\lceil\log m\rceil + 1) \times (2\lceil\log m\rceil + 1)$, padding them by an all zero strip of width $2\lceil\log m\rceil + 1$ on the right and bottom, and then adding a boundary of ones on those two sides. Similarly to Lemma 34, it can be verified that R_m satisfies the desired condition.

We now set

$$m = \left\lfloor \min\left(k - 2\log k - 4, \sqrt{\frac{2\log n}{\log|\Sigma|} - \frac{3\log\log n}{\log|\Sigma|}}\right)\right\rfloor .$$

Let R_m be a set of $|\Sigma|^{m^2-1}$ templates whose construction was described in the paragraph above and set $d = \lfloor (m^2-1)\log|\Sigma|\rfloor$. Since $|R_m| = |\Sigma|^{m^2-1} \geq 2^d$, we can choose 2^d distinct 2D patterns $q_0 \ldots q_{2^d-1}$ from R_m. The dimension of each pattern q_i is $m + 2\lceil\log m\rceil + 2$ which by the choice of m is at most k.

Define a set of $n \times n$ images $Y := \{y_0 \dots y_{d-1}\}$ where y_i is an image containing all the patterns q_j from R_m such that the binary expansion of j equals 1 in the ith location. This way, each image from Y must contain at most 2^{d-1} patterns, while we can fit $\left\lfloor \frac{n}{m+2\lceil \log m \rceil + 2} \right\rfloor^2$ patterns into an image of size $n \times n$. It can be verified that for the chosen values of m and d, $2^{d-1} \leq \left\lfloor \frac{n}{m+2\lceil \log m \rceil + 2} \right\rfloor^2$. Thus, we have that each y_i can be padded to an $n \times n$ image if necessary by assigning 1 to all unassigned positions. Finally, it follows in a similar fashion to the 1D case that the set of patterns $q_0 \dots q_{2^d-1}$ shatters Y. Hence R_m shatters Y. Since $|Y| = d$ the VC dimension of the set of all 2D patterns of dimensions at most k is at least d.

We conclude this discussion with the following Theorem:

▶ **Theorem 44.**

$$\mathrm{VC}(\mathcal{G}_{k,\Sigma}) = \min\left((k - O(\log k))^2 \log|\Sigma|, 2\log n - O(\log\log n)\right) .$$

Efficient Black-Box Identity Testing for Free Group Algebras

V. Arvind
Institute of Mathematical Sciences (HBNI), Chennai, India
arvind@imsc.res.in

Abhranil Chatterjee
Institute of Mathematical Sciences (HBNI), Chennai, India
abhranilc@imsc.res.in

Rajit Datta
Chennai Mathematical Institute, Chennai, India
rajit@cmi.ac.in

Partha Mukhopadhyay
Chennai Mathematical Institute, Chennai, India
partham@cmi.ac.in

Abstract

Hrubeš and Wigderson [12] initiated the study of noncommutative arithmetic circuits with division computing a noncommutative rational function in the *free skew field*, and raised the question of rational identity testing. For noncommutative *formulas* with inverses the problem can be solved in deterministic polynomial time in the *white-box* model [10, 13]. It can be solved in randomized polynomial time in the *black-box* model [8], where the running time is polynomial in the size of the formula. The complexity of identity testing of noncommutative rational functions, in general, remains open for noncommutative circuits with inverses.

We solve the problem for a natural special case. We consider expressions in the free group algebra $\mathbb{F}\langle X, X^{-1}\rangle$ [1] where $X = \{x_1, x_2, \ldots, x_n\}$. Our main results are the following.

1. Given a degree d expression f in $\mathbb{F}\langle X, X^{-1}\rangle$ as a black-box, we obtain a randomized $\mathrm{poly}(n, d)$ algorithm to check whether f is an identically zero expression or not. The technical contribution is an Amitsur-Levitzki type theorem [1] for $\mathbb{F}\langle X, X^{-1}\rangle$. This also yields a deterministic identity testing algorithm (and even an expression reconstruction algorithm) that is polynomial time in the sparsity of the input expression.
2. Given an expression f in $\mathbb{F}\langle X, X^{-1}\rangle$ of degree D and sparsity s, as black-box, we can check whether f is identically zero or not in randomized $\mathrm{poly}(n, \log s, \log D)$ time. This yields a randomized polynomial-time algorithm when D and s are exponential in n.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation

Keywords and phrases Rational identity testing, Free group algebra, Noncommutative computation, Randomized algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.57

Category RANDOM

Acknowledgements We thank the anonymous reviewers of RANDOM 2019 for their valuable comments.

[1] Here $\mathbb{F}\langle X, X^{-1}\rangle$ denotes $\mathbb{F}\langle x_1, \ldots, x_n, x_1^{-1}, \ldots, x_n^{-1}\rangle$.



© V. Arvind, Abhranil Chatterjee, Rajit Datta, and Partha Mukhopadhyay;
licensed under Creative Commons License CC-BY



1 Introduction

Noncommutative computation is an important sub-area of arithmetic circuit complexity. In the usual arithmetic circuit model for noncommutative computation, the arithmetic operations are addition and multiplication, where each circuit input is either a variable from $X = \{x_1, x_2, \ldots, x_n\}$ or a scalar from a prescribed field $\mathbb{F}$. Each multiplication gate in the circuit respect the order of its inputs since the variables x_i are noncommuting. Such circuits compute precisely noncommutative polynomials in the *free noncommutative ring* denoted by $\mathbb{F}\langle X\rangle$.

Analogous to commutative arithmetic computation, the central questions are to show circuit size lower bounds for explicit noncommutative polynomials and derandomization of polynomial identity testing (PIT) for noncommutative circuits (or subclasses of circuits). There is nontrivial progress on these problems unlike in the commutative case. Nisan [16] has shown that any algebraic branching program (ABP) computing the $n \times n$ noncommutative Determinant or Permanent polynomial requires exponential (in n) size. Raz and Shpilka [17] have shown a deterministic polynomial-time PIT for noncommutative ABPs in the white-box model. A quasi-polynomial time derandomization is also known for the black-box model [9]. However, for general circuits there are no better results (either lower bound or PIT) than known in the commutative setting.

The randomized polynomial-time PIT algorithm for noncommutative circuits computing a polynomial of polynomially bounded degree [6] follows from the Amitsur-Levitzki theorem [1] which states that a nonzero polynomial $p \in \mathbb{F}\langle X\rangle$ of degree $< 2k$ cannot be an identity for the ring $\mathbb{M}_k(\mathbb{F})$ of $k \times k$ matrices over $\mathbb{F}$. It is also known [2] that a nonzero noncommutative polynomial does not vanish on matrices of dimension logarithmic in the sparsity of the polynomial. This yields a randomized polynomial-time identity test for noncommutative circuits computing polynomials of exponential degree and exponential sparsity.

Hrubeš and Wigderson [12] initiated the study of noncommutative computation with inverses. In the commutative world, it suffices to consider additions and multiplications. By Strassen's result [20] (extended to finite fields [11]), divisions can be efficiently replaced by polynomially many additions and multiplications. However, divisions in noncommutative computation are more intricate [12]. In the same paper [12], the authors introduce *rational identity testing*: Given a noncommutative formula involving addition, multiplication and division gates, efficiently check if the resulting rational expression is identically zero in the free skew-field of noncommutative rational functions. They show that it is reducible to the following SINGULAR problem:

Given a matrix $A_{n\times n}$ where the entries are linear forms over noncommuting variables $\{x_1, x_2, \ldots, x_n\}$, is A invertible in the free skew-field?

In the white-box model the problem is in deterministic polynomial time [10, 13], and in randomized polynomial time in the black-box model [8]. Specifically, for rational formulas of size s, random matrix substitutions of dimension linear in s suffices to test if the rational expression is identically zero [8].

The complexity of identity testing for general rational expressions remains open. For example, given a noncommutative circuit involving addition, multiplication and division gates, no efficient algorithm (even randomized!) is known to check if the resulting rational expression is identically zero in the free skew-field of noncommutative rational functions. In order to precisely formulate the problem, we define classes of rational expressions based on Bergman's definition [5] of *inversion height* which we now recall and elaborate upon with some notation.

▶ **Definition 1** ([5]). *Let X be a set of free noncommuting variables. Polynomials in the free ring $\mathbb{F}\langle X\rangle$ are defined to be rational expressions of* height 0. *A rational expression of* height $i+1$ *is inductively defined to be a polynomial in rational expressions of height at most i, and inverses of such expressions.*

Let $\mathcal{E}_{d,0}$ denote all polynomials of degree at most d in the free ring $\mathbb{F}\langle X\rangle$. We inductively define rational expressions in $\mathcal{E}_{d,i+1}$ as follows: Let $f_1, f_2, \ldots, f_r$ and $g_1, g_2, \ldots, g_s$ be rational expressions in $\mathcal{E}_{d,i}$ in the variables $x_1, x_2, \ldots, x_n$. Let $f(y_1, y_2, \ldots, y_s, z_1, z_2, \ldots, z_r)$ be a degree-d polynomial in $\mathbb{F}\langle X\rangle$. Then $f(g_1, g_2, \ldots, g_s, f_1^{-1}, f_2^{-1}, \ldots, f_r^{-1})$ is a rational expression (of inversion height $i+1$) in $\mathcal{E}_{d,i+1}$.

Black-box identity testing for rational expressions is not well understood in general. In particular, no efficient randomized algorithm seems to be known even for identity testing of the class $\mathcal{E}_{d,1}$. One source of difficulty is the subtle behaviour of rational expressions when evaluated on matrix algebras. For example, a surprising result of Bergman [5, Proposition 5.1] shows that there are rational expressions that are nonzero over a dense subset of 2×2 matrices but evaluate to zero on dense subsets of 3×3 matrices.

▶ Remark 2. In this connection, we note that Hrubeš and Wigderson [12] have observed that testing if a *correct* rational expression Φ (see [12], Section 2) is not identically zero is equivalent to testing if the rational expression Φ^{-1} is *correct*. I.e. testing if a correct rational expression of *inversion height* i is identically zero or not can be reduced to testing if a rational expression of *inversion height* $i+1$ is correct or not. Furthermore, testing if a rational expression of *inversion height one* is correct can be done by applying (to each inversion operation in this expression) a theorem of Amitsur (see [18, 15]) which implies that a nonzero degree $2d-1$ noncommutative polynomial evaluated on $d \times d$ matrices will be invertible with high probability. However, this does not yield an efficient randomized identity testing algorithm for rational expressions of inversion height one. Because that requires testing correctness of expressions of inversion height two which is a question left open in their paper [12, Section 9].

Free Group Algebras

This motivates the study of black-box identity testing for rational expressions in the *free group algebra* $\mathbb{F}\langle X, X^{-1}\rangle$ which is a natural subclass of rational expressions of inversion height one, as we explain next.

We consider expressions in the free group algebra $\mathbb{F}\langle X, X^{-1}\rangle$, where $(X, X^{-1})^*$ denotes the free group generated by the n generators $X = \{x_1, x_2, \ldots, x_n\}$ and their inverses

$$X^{-1} = \{x_1^{-1}, x_2^{-1}, \ldots, x_n^{-1}\}.$$

Elements of the free group $(X, X^{-1})^*$ are words in X, X^{-1}. The only relations satisfied by the generators is $x_i x_i^{-1} = x_i^{-1} x_i = 1$ for all i. Thus, the elements in the free group $(X, X^{-1})^*$ are the *reduced words* which are words to which the above relations are not applicable.

The elements of the *free group algebra* $\mathbb{F}\langle X, X^{-1}\rangle$ are $\mathbb{F}$-linear combinations of the form

$$f = \sum_w \alpha_w w, \quad \alpha_w \in \mathbb{F},$$

where each $w \in (X, X^{-1})^*$ is a reduced word. The *degree* of the expression f is defined as the maximum length of a word w such that $\alpha_w \neq 0$. The expression f is said to have *sparsity* s if there are s many reduced words w such that $\alpha_w \neq 0$ in f. We also use the notation $[w]f$ to denote the coefficient α_w of the reduced word w in the expression f.

The free noncommutative ring $\mathbb{F}\langle X\rangle$ is a subalgebra of $\mathbb{F}\langle X, X^{-1}\rangle$. Clearly, the elements of $\mathbb{F}\langle X, X^{-1}\rangle$ are a special case of rational expressions of *inversion height one*. I.e., we note that:

▶ **Proposition 3.** $\mathbb{F}\langle X, X^{-1}\rangle \subset \cup_{d>0}\mathcal{E}_{d,1}$.

Note that the rational expressions in $\mathbb{F}\langle X, X^{-1}\rangle$ allows inverses only of the variables x_i, whereas the *free skew field* $\mathbb{F}\langle\langle X\rangle\rangle$ contains all possible rational expressions (with inverses at any nested level).

Our results

Our main goal is to obtain black-box identity tests for rational expressions in the free group algebra $\mathbb{F}\langle X, X^{-1}\rangle$.

Our first result is an Amitsur-Levitzki type theorem [1] for $\mathbb{F}\langle X, X^{-1}\rangle$. Let A be an associative algebra with identity over $\mathbb{F}$. An expression $f \in \mathbb{F}\langle X, X^{-1}\rangle$ is an *identity* for A if

$$f(a_1, \ldots, a_n) = 0,$$

for all $a_i \in A$ such that a_i^{-1} is defined for each $i \in [n]$.

▶ **Theorem 4.** *Let $\mathbb{F}$ be any field of characteristic zero and $f \in \mathbb{F}\langle X, X^{-1}\rangle$ be a nonzero expression of degree d. Then f is not an identity for the matrix algebra $\mathbb{M}_{2d}(\mathbb{F})$.*

The following corollary is immediate.

▶ **Corollary 5** (Black-box identity testing in free group algebras)**.** *There is a black-box randomized* $\mathrm{poly}(n, d)$ *identity test for degree d expressions in $\mathbb{F}\langle X, X^{-1}\rangle$.*

If the black-box contains a sparse expression, we show efficient deterministic algorithms for identity testing and interpolation algorithm.

▶ **Theorem 6** (Reconstruction for sparse expressions)**.** *Let $\mathbb{F}$ be any field of characteristic zero and f is an expression in $\mathbb{F}\langle X, X^{-1}\rangle$ of degree d and sparsity s given as black-box. Then we can reconstruct f in deterministic* $\mathrm{poly}(n, d, s)$ *time with matrix-valued queries to the black-box.*

Nonzero polynomials in $\mathbb{F}\langle X\rangle$ of sparsity s cannot vanish on $O(\log s)$ dimensional matrix algebras [2]. We obtain a similar result for $\mathbb{F}\langle X, X^{-1}\rangle$: nonzero expressions in $\mathbb{F}\langle X, X^{-1}\rangle$ of degree D and sparsity s do not vanish on $O(\log s)$ dimensional matrices. It yields a randomized polynomial-time identity test if the black-box contains an expression f of exponential degree and exponential sparsity.

▶ **Theorem 7.** *Let $\mathbb{F}$ be any field of characteristic zero. Then, a degree-D expression $f \in \mathbb{F}\langle X, X^{-1}\rangle$ of sparsity s is not an identity for the matrix algebra $\mathbb{M}_k(\mathbb{F})$ for $k \geq c \log s$ for some small constant c.*

▶ **Corollary 8.** *Given a degree-D expression $f \in \mathbb{F}\langle X, X^{-1}\rangle$ of sparsity s as black box, we can check whether f is identically zero or not in randomized* $\mathrm{poly}(n, \log D, \log s)$ *time.*

▶ Remark 9. We have stated our results for fields of characteristic zero for simplicity. With suitable modifications, the results easily extend to fields of positive characteristic as discussed in Section 4.

Each of our proofs typically involves the construction of a nondeterministic substitution automaton $\mathcal{A}$. Consider any expression $f \in \mathbb{F}\langle X, X^{-1}\rangle$. For $\mathcal{A}$, let M_i denote its transition matrix for variable $x_i \in X$. The automaton $\mathcal{A}$ has the property that $f \not\equiv 0$ iff a specified entry of the matrix $f(M_1, M_2, \ldots, M_n)$ is nonzero. This entry will actually be a commutative polynomial (or a ratio of two commutative polynomials). Automata constructions for noncommutative PIT have been used before [4, 3, 2]. In this work, an important difference is that we have to deal with $\mathbb{F}$-linear combinations of words in $\{X, X^{-1}\}$. Thus, if M_i is the transition matrix for x_i then M_i^{-1} will be substituted for x_i^{-1}. Hence, in the construction we have to ensure M_i is invertible. Furthermore, when the automaton reads x_i^{-1} its state transition will be governed by M_i^{-1}. In order to ensure that the final output matrix is nonzero, the transition matrices for the x_i have to be chosen carefully, taking into account the above aspects.

Organization

The paper is organized as follows. In Section 2, we prove Theorem 4, Corollary 5, and Theorem 6. In Section 3, we prove Theorem 7 and Corollary 8. Finally, in Section 4, we discuss suitable modifications to extend our results for finite fields.

2 An Amitsur-Levitzki Type Theorem

The main idea in our proof is to efficiently encode expressions in $\mathbb{F}\langle X, X^{-1}\rangle$ as polynomials in a suitable commutative ring preserving the identity. Let $\mathbb{F}[Y, Z]$ denote the commutative ring $\mathbb{F}[y_{ij}, z_{ij}]_{i\in[n],j\in[d]}$ for $n, d \in \mathbb{N}$, where $Y = \{y_{ij} \mid i \in [n], j \in [d]\}$ and $Z = \{z_{ij} \mid i \in [n], j \in [d]\}$.

▶ **Definition 10.** *Define a map $\varphi : \mathbb{F}\langle X, X^{-1}\rangle \to \mathbb{F}[Y, Z]$ such that φ is identity on $\mathbb{F}$, and for each reduced word $w = x_{i_1}^{b_1} x_{i_2}^{b_2} \cdots x_{i_d}^{b_d}$,*

$$\varphi(x_{i_1}^{b_1} x_{i_2}^{b_2} \cdots x_{i_d}^{b_d}) = \prod_{j=1}^{d} (\mathbb{1}_{[b_j=1]} \cdot y_{i_j j} + \mathbb{1}_{[b_j=-1]} \cdot z_{i_j j}),$$

where $\mathbb{1}_{[b_j=b]} = 1$ if $b_j = b$ and $\mathbb{1}_{[b_j=b]} = 0$ otherwise.

By linearity the map φ is defined on all expressions in $\mathbb{F}\langle X, X^{-1}\rangle$. We observe the following properties of φ.

1. The map φ is injective on the reduced words $(X, X^{-1})^*$. I.e., it maps each reduced word $w \in (X, X^{-1})^*$ to a unique monomial over the commuting variables $Y \cup Z$.
2. Consequently, φ is identity preserving. I.e., an expression f in $\mathbb{F}\langle X, X^{-1}\rangle$ is identically zero if and only if its image $\varphi(f)$ is the zero polynomial in $\mathbb{F}[Y, Z]$.
3. φ preserves the sparsity of the expression. I.e., f in $\mathbb{F}\langle X, X^{-1}\rangle$ is s-sparse iff $\varphi(f)$ in $\mathbb{F}[Y, Z]$ is s-sparse.
4. Given the image $\varphi(f) \in \mathbb{F}[Y, Z]$ in its sparse description (i.e., as a linear combination of monomials), we can efficiently recover the sparse description of $f \in \mathbb{F}\langle X, X^{-1}\rangle$.

Given polynomials $f, f' \in \mathbb{F}[Y, Z]$, we say f and f' are *weakly equivalent*, if for each monomial m, $[m]f = 0$ if and only if $[m]f' = 0$, where $[m]f$ denotes the coefficient of monomial m in f.

Given a black-box expression f in $\mathbb{F}\langle X, X^{-1}\rangle$, we show how to evaluate it on suitable matrices and obtain a polynomial in $\mathbb{F}[Y, Z]$ that is *weakly equivalent to* $\varphi(f)$ as a specific entry of the resulting matrix. The matrix substitutions are based on automata constructions.

Similar ideas have been used earlier to design PIT algorithms for noncommutative polynomials [4]. However, since we are dealing with rational expressions, some difficulties arise. The matrix substitutions for the variables $x_1, \ldots, x_n$ are obtained as the corresponding transition matrices M_i of the automaton. The matrix substitution for x_i^{-1} will be M_i^{-1}. Therefore, we must ensure that the transition matrices M_i are invertible and sufficiently structured to be useful for the identity testing.

We first illustrate our construction for an example degree-2 expression $f = x_1x_2^{-1} + x_2x_1^{-1}$, where $X = \{x_1, x_2\}$.

The basic "building block" for the transition matrix M_i is the 2×2 block matrix

$$\begin{bmatrix} 0 & y_{ij} \\ \frac{1}{z_{ij}} & 0 \end{bmatrix},$$

whose inverse is

$$\begin{bmatrix} 0 & z_{ij} \\ \frac{1}{y_{ij}} & 0 \end{bmatrix}.$$

When the 2×2 block is the j^{th} diagonal block in M_i, the corresponding automaton will go from state $2j-1$ to state $2j$ replacing x_i by y_{ij} (or if x_i^{-1} occurs, it will replace it by z_{ij}).

We will keep the transition matrix M_i for x_i a block diagonal matrix with such 2×2 invertible blocks as the principal minors along the diagonal. In order to ensure this we introduce two new variables $W = \{w_1, w_2\}$ and substitute x_i by the word $w_ix_iw_i$ in the expression. This will ensure that we do not have two consecutive x_i in the resulting reduced words. In fact, between two X variables (or their inverses) we will have inserted exactly two W variables (or their inverses). Now, we define M_i for the above example as

$$M_i = \begin{bmatrix} 0 & y_{i1} & 0 & 0 \\ \frac{1}{z_{i1}} & 0 & 0 & 0 \\ 0 & 0 & 0 & y_{i2} \\ 0 & 0 & \frac{1}{z_{i2}} & 0 \end{bmatrix}, \qquad M_i^{-1} = \begin{bmatrix} 0 & z_{i1} & 0 & 0 \\ \frac{1}{y_{i1}} & 0 & 0 & 0 \\ 0 & 0 & 0 & z_{i2} \\ 0 & 0 & \frac{1}{y_{i2}} & 0 \end{bmatrix}.$$

The corresponding transitions of the automaton is shown in Figure 1.

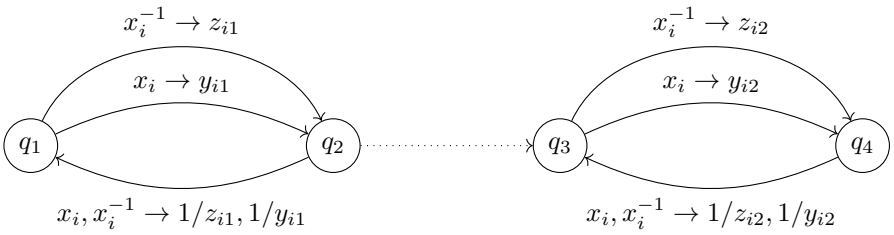


Figure 1 The transition diagram of the automaton for x variables.

We now describe the transition matrices N_i for w_i. The matrix N_i is also a 4×4 block diagonal matrix. There are three blocks along the diagonal. The first and third are 1×1 blocks of the identity. The second one is a 2×2 block for w_i-transitions from state q_2 to state q_3. It ensures that for any subword $w_1^{b_1}w_2^{b_2}$, $b_i \in \{1, -1\}$, in the resulting product matrix $N_1^{b_1}N_2^{b_2}$ the $(1,2)^{th}$ entry of the 2×2 block is nonzero. The corresponding transitions of the automaton is depicted in Figure 2.

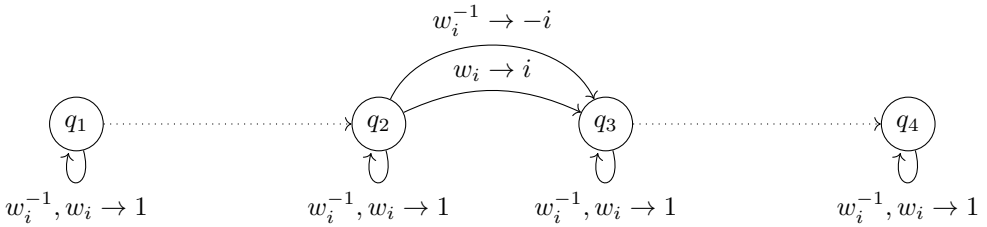


Figure 2 The transition diagram of the automaton for w variables.

$$N_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & i & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad N_i^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -i & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad N_i^{b_1} N_j^{b_2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & b_1 i + b_2 j & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Hence, evaluating $f(N_1 M_1 N_1, N_2 M_2 N_2)$ we obtain (a polynomial weakly equivalent to) $\varphi(f)$ at the $(1,4)^{th}$ entry. The complete automaton is depicted in figure 3.

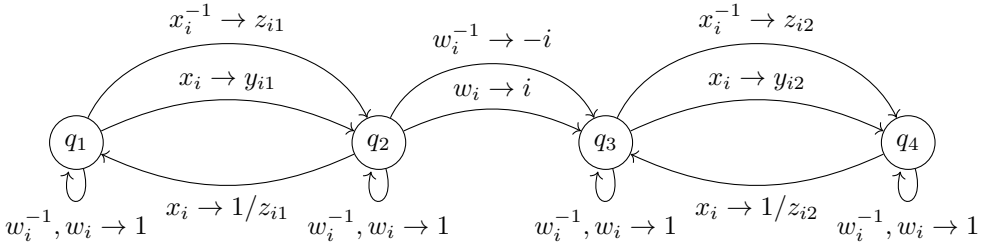


Figure 3 The transition diagram of the automaton.

We now explain the general construction. For $f \in \mathbb{F}\langle X, X^{-1}\rangle$ let $H_\ell(f)$ denote the degree-ℓ homogeneous part of f. We will denote by $\varphi(\widehat{H_\ell(f)})$ an arbitrary polynomial in $\mathbb{F}[Y, Z]$ weakly equivalent to $\varphi(H_\ell(f))$.

▶ **Lemma 11.** *Let $f \in \mathbb{F}\langle X, X^{-1}\rangle$ be a nonzero expression of degree d. There is an n-tuple of $2d \times 2d$ matrices $(M_1, M_2, \ldots, M_n)$ whose entries are either scalars, or variables $u \in Y \cup Z$, or their inverses $1/u$, such that*

$$(f(M_1, \ldots, M_n))_{1,2d} = \varphi(\widehat{H_d(f)}).$$

Furthermore, for each degree-d reduced word of $m = x_{i_1}^{b_1} x_{i_2}^{b_2} \cdots x_{i_d}^{b_d}$ in $\mathbb{F}\langle X, X^{-1}\rangle$,

$$[\varphi(m)]\varphi(\widehat{H_d(f)}) = [m]f \cdot \prod_{j=1}^{d-1} (b_j \cdot i_j + b_{j+1} \cdot i_{j+1}). \tag{1}$$

Proof. Let e_{ij}, for $i, j \in [k]$, be the $(i,j)^{th}$ elementary matrix in $\mathbb{M}_k(\mathbb{F})$: its $(i,j)^{th}$ entry is 1 and other entries are 0.

We now define the transition matrices of the NFA for variables $\{w_i : 1 \le i \le n\}$ and $\{x_i : 1 \le i \le n\}$. For each $i \in [n]$, define 2×2 matrix $N'_i = e_{11} + e_{22} + i \cdot e_{12}$. Now N_i is a $2d \times 2d$ matrix defined as the block diagonal matrix,

$$N'_i = \begin{bmatrix} 1 & i \\ 0 & 1 \end{bmatrix}, \qquad N_i = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & N'_i & 0 & \dots & 0 & 0 \\ 0 & 0 & N'_i & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & N'_i & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

$$N'^{-1}_i = \begin{bmatrix} 1 & -i \\ 0 & 1 \end{bmatrix}, \qquad N^{-1}_i = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & N'^{-1}_i & 0 & \dots & 0 & 0 \\ 0 & 0 & N'^{-1}_i & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & N'^{-1}_i & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

Each $M_i, 1 \le i \le n$ is the $2d \times 2d$ block diagonal matrix where each 2×2 block $M'_{ij}, 1 \le j \le d$ is a 2×2 matrix defined as $M'_{i,j} = y_{ij} \cdot e_{12} + \frac{1}{z_{ij}} \cdot e_{21}$. Their inverses have a similar structure.

$$M'_{i,p} = \begin{bmatrix} 0 & y_{ip} \\ \frac{1}{z_{ip}} & 0 \end{bmatrix}, \qquad M_i = \begin{bmatrix} M'_{i,1} & 0 & 0 & \dots & 0 \\ 0 & M'_{i,2} & 0 & \dots & 0 \\ 0 & 0 & M'_{i,3} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & M'_{i,d} \end{bmatrix}.$$

$$M'^{-1}_{i,p} = \begin{bmatrix} 0 & z_{ip} \\ \frac{1}{y_{ip}} & 0 \end{bmatrix}, \qquad M^{-1}_i = \begin{bmatrix} M'^{-1}_{i,1} & 0 & 0 & \dots & 0 \\ 0 & M'^{-1}_{i,2} & 0 & \dots & 0 \\ 0 & 0 & M'^{-1}_{i,3} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & M'^{-1}_{i,d} \end{bmatrix}.$$

The corresponding NFA is depicted in Figure 4. We substitute each x_{i_j} by the $2d \times 2d$ matrix $N_{i_j} M_{i_j} N_{i_j}$. Each $x^{-1}_{i_j}$ is substituted by its inverse matrix $N^{-1}_{i_j} M^{-1}_{i_j} N^{-1}_{i_j}$.

Correctness:

Consider a degree-d reduced word $m = x^{b_1}_{i_1} x^{b_2}_{i_2} \cdots x^{b_d}_{i_d}$.

Following the automaton construction of Figure 4, $x^{b_i}_i$ occurring at position j is substituted by $(\mathbb{1}_{[b_i=1]} y_{ij} + \mathbb{1}_{[b_i=-1]} z_{ij})$. Moreover, for each position $j \in [d-1]$, the adjacent pair $x^{b_j}_{i_j} x^{b_{j+1}}_{i_{j+1}}$ produces a scalar factor $(b_j \cdot i_j + b_{j+1} \cdot i_{j+1})$ due to the product $N^{b_j}_{i_j} N^{b_{j+1}}_{i_{j+1}}$. Consequently, it follows that

$$(m(M_1, \dots, M_n))_{1,2d} = \prod_{j=1}^{d-1} (b_j \cdot i_j + b_{j+1} \cdot i_{j+1}) \prod_{j=1}^{d} ([b_j = 1] y_{i_j j} + [b_j = -1] z_{i_j j}).$$

As φ is a linear map, the lemma follows. ◀

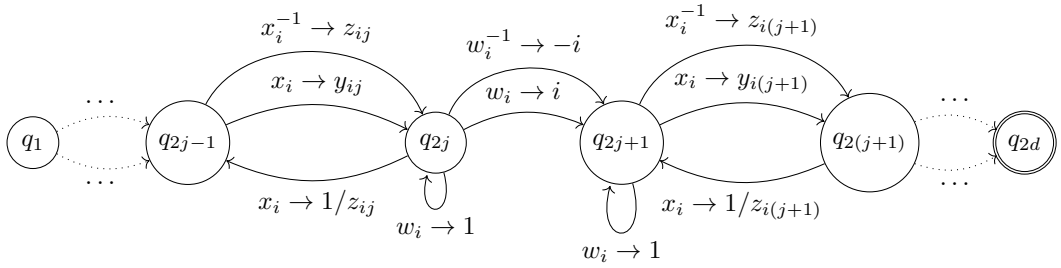


Figure 4 The transition diagram of the automaton.

2.1 Black-box identity testing for circuits in free group algebras

Proof of Theorem 4. The proof follows easily from Lemma 11. Lemma 11 says that if $f \in \mathbb{F}\langle X, X^{-1}\rangle$ is nonzero of degree d then the $(1, 2d)$ entry of the matrix $p(N_1M_1N_1, \ldots, N_nM_nN_n)$ is a nonzero polynomial in $\mathbb{F}[Y, Z]$. Hence f can not be an identity for $M_{2d}(\mathbb{F})$. ◀

Proof of Corollary 5. The identity testing algorithm follows from Theorem 4. We can randomly substitute for the variables and apply the Schwartz-Zippel-Demillo-Lipton Theorem [19, 21, 7]. This completes the proof of the Corollary 5. ◀

2.2 Reconstruction of sparse expressions

If the black-box contains an s-sparse expression in $\mathbb{F}\langle X, X^{-1}\rangle$, we give a $\text{poly}(s, n, d)$ deterministic interpolation algorithm (which also gives a deterministic identity testing for such expressions). We use a result of Klivans-Spielman [14, Theorem11] that constructs a test set in deterministic polynomial time for sparse commutative polynomials, which is used for the interpolation algorithm.

Proof of Theorem 6. Let the black-box expression f be s-sparse of degree d. By Lemma 11, a polynomial $\varphi(\widehat{H_d(f)})$ in $\mathbb{F}[Y, Z]$ is obtained at the $(1, 2d)^{th}$ entry of the matrix $f(M_1, \ldots, M_n)$, where $M_i \in \mathbb{M}_{2d}(\mathbb{F}[Y, Z])$ is as defined in Lemma 11. By Definition 10, $\varphi(f) \in \mathbb{F}[Y, Z]$ is s-sparse and has $2nd$ variables. Let $\mathcal{H}_{2nd,d,s}$ be the corresponding test set from [14] to interpolate a polynomial of degree d and s-sparse over $2nd$ variables. Querying the black-box on $M_1(\vec{h}), M_2(\vec{h}), \ldots, M_n(\vec{h})$ for each $\vec{h} \in \mathcal{H}_{2nd,d,s}$ we can interpolate the commutative polynomial $\varphi(\widehat{H_d(f)})$ and obtain an expression for $\varphi(\widehat{H_d(f)}) = \sum_{t=1}^{s} c_{m_t} m_t$ as a sum of monomials.

We will now adjust the extra scalar factors for each monomial in $\varphi(\widehat{H_d(f)})$ to obtain $\varphi(H_d(f))$. We can adjust this for each monomial as Lemma 11 shows that the extra scalar factor for the word $m = x_{i_1}^{b_1} x_{i_2}^{b_2} \cdots x_{i_\ell}^{b_\ell}$ is just $\alpha_{\varphi(m)} = \prod_{j=1}^{\ell-1}(b_j \cdot i_j + b_{j+1} \cdot i_{j+1})$. So we construct $\varphi(H_d(f)) = \sum_{t=1}^{s} \frac{c_{m_t}}{\alpha_{m_t}} m_t$ by removing the factors α_{m_t} for each monomial m_t. We now *invert* the map φ (using the 4^{th} property of Definition 10) on every monomial m_t to obtain $H_d(f)$ as a sum of degree d reduced words. This yields the expression for highest degree homogeneous component of f. We can repeat the above procedure on $f - H_d(f)$ and reconstruct the remaining homogeneous components of f. ◀

3 Black-box Identity Testing for Expressions of Exponential Degree and Exponential Sparsity

It is known [2] that a nonzero noncommutative polynomial of sparsity s cannot be an *identity* for $O(\log s)$ dimensional matrix algebras. In this section, we show a similar result for free group algebras. In particular, we prove that the dimension of the matrix algebra for which a nonzero free group algebra expression f does not vanish is logarithmic in the sparsity of f. It yields a randomized $\text{poly}(\log D, \log s, n)$ time identity testing algorithm when the black-box contains an expression of degree D and sparsity s.

We first recall the notion of *isolating index set* from [2].

▶ **Definition 12.** *Let $\mathcal{M} \subseteq \{X, X^{-1}\}^D$ be a subset of reduced words of degree D. An index set $I \subseteq [D]$ is an* isolating index set *for $\mathcal{M}$ if there is a word $m \in \mathcal{M}$ such that for each $m' \in \mathcal{M} \setminus \{m\}$ there is an index $i \in I$ for which $m[i] \neq m'[i]$. I.e. no other word in $\mathcal{M}$ agrees with m on all positions in the index set I. We say m is an* isolated word*.*

In the following lemma we show that $\mathcal{M}$ has an isolating index set of size $\log|\mathcal{M}|$. The proof is identical to [2]. Nevertheless, we give the simple details for completeness as we deal with both variables and their inverses.

▶ **Lemma 13** ([2])**.** *Let $\mathcal{M} \subseteq \{X, X^{-1}\}^D$ be reduced degree-D words. Then $\mathcal{M}$ has an isolating index set of size k which is bounded by $\log|\mathcal{M}|$.*

Proof. The words $m \in \mathcal{M}$ are indexed, where $m[i]$ denotes the variable (or the inverse of a variable) in the i^{th} position of m. Let $i_1 \leq D$ be the first index such that not all words agree on the i_1^{th} position. Let

$$S_j^+ = \{m : m[i_1] = x_j\}$$
$$S_j^- = \{m : m[i_1] = x_j^{-1}\}.$$

For some j, $|S_j^+|$ or $|S_j^-|$ is of size at most $|\mathcal{M}|/2$. Let S_j^b denote that subset, $b \in \{+, -\}$. We replace $\mathcal{M}$ by S_j^b and repeat the same argument for at most $\log|\mathcal{M}|$ steps. Clearly, by this process, we identify a set of indices $I = \{i_1, \ldots, i'_k\}$, $k' \leq \log|\mathcal{M}|$ such that the set shrinks to a singleton set $\{m\}$. Clearly, I is an isolating index set as witnessed by the *isolating word* m. ◀

Proof of Theorem 7

Proof. Let $k = 4(k'+1)$ where k' is the size of the isolating set I. As in Section 2, we substitute each x_i by $w_i x_i w_i$, where $w_i, i \in [n]$ are n new variables. The transition matrices for w_i and x_i are denoted by N_i and M_i respectively.

For $1 \leq i \leq n$, we define $k \times k$ matrix N_i as a block diagonal matrix of $k'+1$ many copies of a 4×4 matrix N'_i where $N'_i = I_4 + i(e_{12} + e_{34} + e_{32} + e_{14})$.

$$N'_i = \begin{bmatrix} 1 & i & 0 & i \\ 0 & 1 & 0 & 0 \\ 0 & i & 1 & i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad N_i = \begin{bmatrix} N'_i & 0 & 0 & \ldots & 0 \\ 0 & N'_i & 0 & \ldots & 0 \\ 0 & 0 & N'_i & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & N'_i \end{bmatrix},$$

$$N_i'^{-1} = \begin{bmatrix} 1 & -i & 0 & -i \\ 0 & 1 & 0 & 0 \\ 0 & -i & 1 & -i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad N_i^{-1} = \begin{bmatrix} N_i'^{-1} & 0 & 0 & \dots & 0 \\ 0 & N_i'^{-1} & 0 & \dots & 0 \\ 0 & 0 & N_i'^{-1} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & N_i'^{-1} \end{bmatrix}.$$

Notice that

$$N_i'^{b_1} N_j'^{b_2} = \begin{bmatrix} 1 & (b_1 i + b_2 j) & 0 & (b_1 i + b_2 j) \\ 0 & 1 & 0 & 0 \\ 0 & (b_1 i + b_2 j) & 1 & (b_1 i + b_2 j) \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

We now define the $k \times k$ transition matrix M_i as a block diagonal matrix,

$$M'_{i,j} = \begin{bmatrix} 0 & y_{ij} \\ \frac{1}{z_{ij}} & 0 \end{bmatrix}, \qquad M'_{\xi_i} = \begin{bmatrix} 0 & \xi_i \\ \frac{1}{\xi_i} & 0 \end{bmatrix},$$

$$M_i = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & M_{\xi_1} & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & M'_{i,1} & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & M_{\xi_2} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & M_{\xi_{k'+1}} & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

These matrices can be seen as the transitions of a suitable NFA. We sketch the construction of this NFA.

Let $I = \{i_1, \dots, i_{k'}\}$ be an isolating set such that $i_1 < \dots < i_{k'}$. Intuitively, the NFA does one of two operations on each symbol (a variable or its inverse) of the input expression: a *Skip* or an *Encode*. In a *Skip* stage, the NFA deals with positions that are not part of the (guessed) isolating index set. In this stage, the NFA substitutes the w_i variables by suitable scalars (coming from the N'_i matrices) and x_i variables by block variables $\{\xi_1, \dots \xi_{k'+1}\}$. The NFA nondeterministically decides whether the *Skip* stage is over and it enters the *Encode* stage for a guessed index of the isolating set. It substitutes x_i and x_i^{-1} variables by y_{ij} and z_{ij} respectively. Fig. 5 summarizes the action of the NFA.

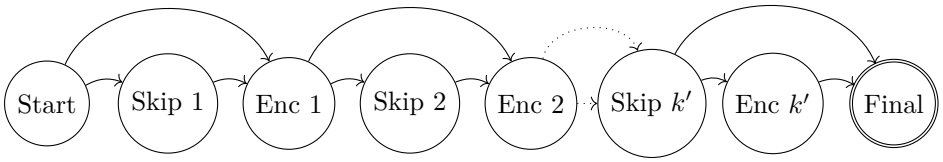


Figure 5 The transition diagram of the automaton.

Define $\hat{f}$ in $\mathbb{F}(Y, Z, \overline{\xi})$ to be rational function we obtain at the $(1,k)^{th}$[2] entry by evaluating the expression $f(N_1M_1N_1, \ldots, N_nM_nN_n)$. Notice that, the isolating word m of degree D will be of following form $m = W_1 x_{i_1}^{b_{i_1}} W_2 x_{i_2}^{b_{i_2}} \cdots W'_k x_{i'_k}^{b_{i'_k}} W_{k'+1}$ where each subword $W_j = x_{j_1}^{b_1} x_{j_2}^{b_2} \cdots x_{j_{\ell_j}}^{b_{\ell_j}}$ is of length $\ell_j \geq 0$, where some of the W_j could be the empty word as well.

We refer to an NFA transition $q_i \to q_j$ as a *forward edge* if $i < j$ and a *backward edge* if $i > j$. We classify the backward edges in three categories based on the substitution on the edge-label. We say, a backward edge is of *type A* if a variable is substituted by a scalar value; a backward edge is of *type B* if a variable is substituted by $\frac{1}{\xi_j}$ for some j; a backward edge is of *type C* if a variable is substituted by $\frac{1}{y_{ij}}$ or $\frac{1}{z_{ij}}$ for some i, j.

Consider a walk of the NFA on an input word m that reaches state k using only *type A* backward edges. In that case, m is substituted by $\alpha \cdot \hat{m}$ where $\hat{m}$ is a monomial over $\{Y, Z, \xi\}$ of same degree,

$$\hat{m} = \prod_{j=1}^{k'+1} \xi_j^{\ell_j} \cdot \prod_{j=1}^{k'} ([b_{i_j} = 1] y_{i_j j} + [b_{i_j} = -1] z_{i_j j}).$$

and α is some nonzero constant obtained as a product of $[m]f$ with the scalars obtained as substitutions from the edges involving the w_i variables in the *Skip* stages. Indeed, as we can see from the entries of product matrices $N_i'^{b_1} \cdot N_j'^{b_2}$, where $b_1, b_2 \in \{-1, 1\}$, the scalar α is a product of $[m]f$ with terms of the form $b_1 i + b_2 j$, for $i \neq j$, each of which is nonzero for any reduced word.

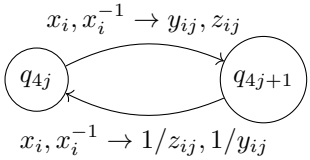


Figure 6 The transition diagram of the automaton at *Encode* stage.

▷ Claim 14.

$[\hat{m}]\hat{f} \neq 0$ iff $[m]f \neq 0$.

Proof. It suffices to show that for any word $m' \neq m$, where m' has degree $\leq D$, no walks of the NFA accepting m' generate $\hat{m}$ after substitution. For a computation path J, the monomial m_J in $\hat{f}$ has two parts, let us call it $skip_J$ and $encode_J$ where $skip_J$ is a monomial over $\{\xi_1, \ldots, \xi_{k'+1}\}$ and $encode_J$ is a monomial over $\{y_{ij}, z_{ij}\}_{i\in[n], j\in[k']}$. If the computation path J (which is different from the computation path described above for $\hat{m}$) uses only *type A* backward edges, then necessarily $m_J \neq \hat{m}$ from the definition of *isolating index set*. This argument is analogous to the argument given in [2].

Now consider a walk J which involves backward edges of other types. Let us first consider those walks that take backward edges only of *type A* and *type B*. Such a walk still produces a monomial over $\{y_{ij}, z_{ij}\}_{i\in[n], j\in[k']}$ and $\{\xi_i\}_{1\leq i\leq k'+1}$ because division only by ξ_i variables

[2] Recall that $k = 4(k'+1)$ where k' is the size of an isolating set.

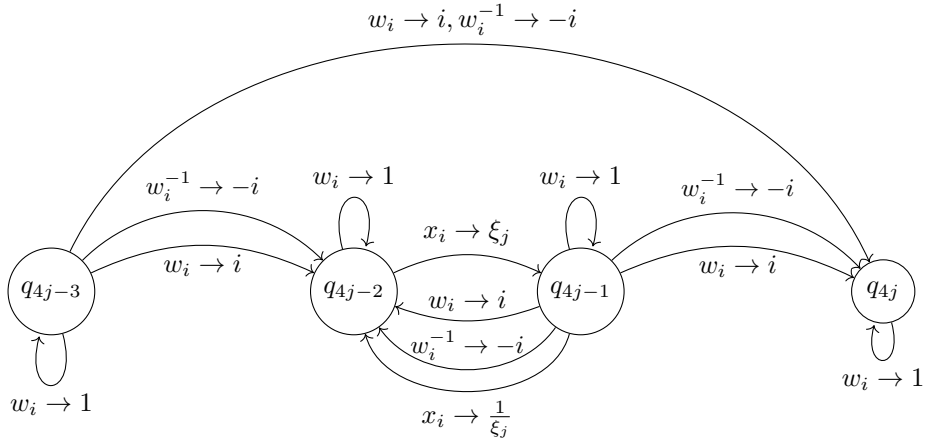


Figure 7 The transition diagram of the automaton at *Skip* stage.

occur in the resulting expression. Since $\hat{m}$ is of highest degree, the total degree of these monomials is strictly lesser than degree of $\hat{m}$. For those walks that take at least one backward edge of *type C*, a rational expression in $\{y_{ij}, z_{ij}\}_{i\in[n],j\in[k']}$ and $\{\xi_i\}_{1\le i\le k'+1}$ is produced (as there is division by y_{ij} or z_{ij} variables). As the sum of the degree of the numerator and degree of the numerator is bounded by the total degree, the degree of the numerator is smaller than degree of $\hat{m}$.

Thus the $(1,k)^{th}$ entry of the output matrix is of the form $\sum_{i=1}^{N_1} c_i m_i + \sum_{j=1}^{N_2} r_j$ where $\{m_1,\ldots,m_{N_1}\}$ are monomials arising from different walks (w.l.o.g. assume that $m_1 = \hat{m}$) and $\{r_1,\ldots,r_{N_2}\}$ are the rational expressions from the other walks (due to the backward edges of *type C*). Note that, denominator in each r_j is a monomial over Y, Z of degree at most D. Let $L = \prod_{i=1}^{n}\prod_{j=1}^{k'} y_{i,j}^D \cdot z_{i,j}^D$. Now, we have,

$$\sum_{i=1}^{N_1} c_i m_i + \sum_{j=1}^{N_2} r_j = \frac{1}{L}\cdot\left(\sum_{i=1}^{N_1} c_i m_i L + \sum_{j=1}^{N_2} p_j\right).$$

Since $\hat{m}L \neq m_i L$ for any $i \in \{2,\ldots,N_1\}$ and degree of each $p_j <$ degree of $\hat{m}L$ for any $j \in \{1,\ldots,N_2\}$, the numerator of the final expression is a nonzero polynomial in $\mathbb{F}[Y,Z,\overline{\xi}]$. ◁

The above proof shows that the matrix $f(N_1M_1N_1,\ldots,N_nM_nN_n)$ is nonzero with rational entries in $\mathbb{F}[Y,Z,\overline{\xi}]$. Each entry is a linear combination of terms of the form m_1/m_2, where m_1 and m_2 are monomials in $Y \cup Z \cup \{\xi_1,\ldots,\xi_{k'+1}\}$ of degree bounded by D. Note that, the matrix dimension is $k = c\log s$ for some constant c. This completes the proof of Theorem 7. ◀

To get an identity testing algorithm, we can do random substitutions.The matrix dimension is $\log s$ and the overall running time of the algorithm is $\mathrm{poly}(n, \log s, \log D)$. This also proves Corollary 8. ◀

▶ Remark 15. For algorithmic purposes, we note that Theorem 4 is sometimes preferable to Theorem 7. For instance, the encoding used in Theorem 7 does not preserve the sparsity of the polynomial as required in the sparse reconstruction result (see Theorem 6).

4 Adaptation for Fields of Positive Characteristic

Let $\mathbb{F}$ be any finite field of characteristic p. We will ensure that for each word m in the free group algebra, the scalar α_m (see Equation 1) produced by the automaton described in Section 2 is not zero in $\mathbb{F}$. Recall that, reading $w_i^{b_i} w_j^{b_j}$ for two consecutive positions, the automaton produces a scalar $(b_i \cdot i + b_j \cdot j)$ where $b_i, b_j \in \{-1, +1\}$. Moreover, this is the only way the automaton produces a scalar and for each m, α_m is a product of such terms. Hence, it suffices to ensure that for each pair $i, j \in [n]$, $(b_i \cdot i + b_j \cdot j) \neq 0$. Similarly, it ensures that the scalar produced by the automaton described in Section 3 is nonzero.

We note that, if p is more than $2n$ then each term $(b_i \cdot i + b_j \cdot j) \neq 0 \pmod{p}$ where $b_i, b_j \in \{-1, +1\}$ and $i, j \in [n]$. This results in a dependence on the characteristic of the base field for the analogous statements of Theorems 4, 7 for finite field. Additionally, for Theorem 4, the $(1, 2d)^{th}$ entry of the output matrix is a polynomial of degree d, and for Theorem 7, the degrees of the numerator polynomials in the rational expression of the output matrix is bounded by some scalar multiple of $nD \log s$. This lower bounds the size of the fields in the application. We summarize the above discussion in the following.

▶ **Observation 16.** *We can obtain results analogous to Theorem 4 and Theorem 7 for finite fields of characteristic more than $2n$ and sizes at least $d + 1$ and $\Omega(nD \log s)$ respectively.*

However, the algorithms presented in Theorem 6 and Corollaries 5, 8 can be modified to work for finite fields of any characteristic. To this end, we first notice the following simple fact.

▶ **Proposition 17.** *Let $\mathbb{F}$ be a finite field of characteristic $p \leq 2n$. In We can find elements $\alpha_1, \alpha_2, \ldots, \alpha_n$ from a suitable (deterministically constructed) small extension field $\mathbb{F}'$ of $\mathbb{F}$ in deterministic* poly(n) *time, such that for any $b_i \in \{-1, 1\}, 1 \leq i \leq n$ we have*

For each $1 \leq i < j \leq n$, $b_i \alpha_i + b_j \alpha_j \neq 0$.

Let $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}'$ as given by the above proposition. We modify the matrix ${N'}_i$ in the proof of Theorem 6 and Corollary 5 as

$$N'_i = \begin{bmatrix} 1 & \alpha_i \\ 0 & 1 \end{bmatrix},$$

and in Corollary 8 we modify ${N'}_i$ as

$$N'_i = \begin{bmatrix} 1 & \alpha_i & 0 & \alpha_i \\ 0 & 1 & 0 & 0 \\ 0 & \alpha_i & 1 & \alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

For each pair $i, j \in [n]$, $(b_i \cdot \alpha_i + b_j \cdot \alpha_j) \neq 0$ by Proposition 17. Thus, for each word m, the scalar α_m produced by the automata are nonzero in the extension field $\mathbb{F}'$ as well. Furthermore, the test set of [14] works for all fields. Hence Theorem 6 holds for all finite fields too. To obtain Corollaries 5 and 8, we will do random substitutions from a suitable small degree extension field and use Schwartz-Zippel-Demillo-Lipton Theorem [19, 21, 7]. In summary, our algorithms in the paper can be adapted to work for all fields.

Proof of Proposition 17. Define polynomial $g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ as

$$g(x_1, x_2, \ldots, x_n) = \prod_{1 \leq i < j \leq n} (x_i + x_j) \cdot (x_i - x_j).$$

We substitute y^i for $x_i, 1 \leq i \leq n$. Then $g(y, y^2, \ldots, y^n) = G(y) \in \mathbb{F}[y]$ is a univariate polynomial of degree at most $2n^3$. Using standard techniques, in deterministic polynomial time we can construct an extension field $\mathbb{F}'$ of $\mathbb{F}$ such that $|\mathbb{F}'|$ is of poly$(n) \geq 2n^3 + 1$ size. We can find an element $\alpha \in \mathbb{F}'$ such that $G(\alpha) \neq 0$ and set $\alpha_i = \alpha^i, 1 \leq i \leq n$. ◀

References

1 A. S. Amitsur and J. Levitzki. Minimal Identities for Algebras. *Proceedings of the American Mathematical Society*, 1(4):449–463, 1950. URL: http://www.jstor.org/stable/2032312.

2 Vikraman Arvind, Pushkar S. Joglekar, Partha Mukhopadhyay, and S. Raja. Randomized polynomial time identity testing for noncommutative circuits. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 831–841, 2017. doi:10.1145/3055399.3055442.

3 Vikraman Arvind and Partha Mukhopadhyay. The ideal membership problem and polynomial identity testing. *Inf. Comput.*, 208(4):351–363, 2010. doi:10.1016/j.ic.2009.06.003.

4 Vikraman Arvind, Partha Mukhopadhyay, and Srikanth Srinivasan. New Results on Noncommutative and Commutative Polynomial Identity Testing. *Computational Complexity*, 19(4):521–558, 2010. doi:10.1007/s00037-010-0299-8.

5 George M Bergman. Rational relations and rational identities in division rings. *Journal of Algebra*, 43(1):252–266, 1976. doi:10.1016/0021-8693(76)90159-9.

6 Andrej Bogdanov and Hoeteck Wee. More on Noncommutative Polynomial Identity Testing. In *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*, pages 92–99, 2005. doi:10.1109/CCC.2005.13.

7 Richard A. Demillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 7(4):193–195, 1978. doi:10.1016/0020-0190(78)90067-4.

8 Harm Derksen and Visu Makam. Polynomial degree bounds for matrix semi-invariants. *Advances in Mathematics*, 310:44–63, 2017. doi:10.1016/j.aim.2017.01.018.

9 Michael Forbes and Amir Shpilka. Quasipolynomial-time Identity Testing of Non-Commutative and Read-Once Oblivious Algebraic Branching Programs. *Foundations of Computer Science, 1975., 16th Annual Symposium on*, September 2012. doi:10.1109/FOCS.2013.34.

10 Ankit Garg, Leonid Gurvits, Rafael Mendes de Oliveira, and Avi Wigderson. A Deterministic Polynomial Time Algorithm for Non-commutative Rational Identity Testing. *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 109–117, 2016.

11 Pavel Hrubes and Amir Yehudayoff. Arithmetic Complexity in Ring Extensions. *Theory of Computing*, 7:119–129, 2011.

12 Pavel Hrubeš and Avi Wigderson. Non-commutative arithmetic circuits with division. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 49–66, 2014. doi:10.1145/2554797.2554805.

13 Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Constructive non-commutative rank computation is in deterministic polynomial time. *computational complexity*, 27(4):561–593, December 2018. doi:10.1007/s00037-018-0165-7.

14 Adam R. Klivans and Daniel Spielman. Randomness Efficient Identity Testing of Multivariate Polynomials. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 216–223, New York, NY, USA, 2001. ACM. doi:10.1145/380752.380801.

15 Tsiu-Kwen Lee and Yiqiang Zhou. Right ideals generated by an idempotent of finite rank. *Linear Algebra and its Applications*, 431:2118–2126, November 2009. doi:10.1016/j.laa.2009.07.005.

16 Noam Nisan. Lower Bounds for Non-Commutative Computation (Extended Abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418, 1991. doi:10.1145/103418.103462.

17 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005. doi:10.1007/s00037-005-0188-8.

18 Louis Halle Rowen. *Polynomial identities in ring theory*. Pure and Applied Mathematics. Academic Press, 1980.

19 Jacob T. Schwartz. Fast Probabilistic algorithm for verification of polynomial identities. *J. ACM.*, 27(4):701–717, 1980.

20 Volker Strassen. Vermeidung von Divisionen. *Journal für die reine und angewandte Mathematik*, 264:184–202, 1973. URL: http://eudml.org/doc/151394.

21 R. Zippel. Probabilistic algorithms for sparse polynomials. In *Proc. of the Int. Sym. on Symbolic and Algebraic Computation*, pages 216–226, 1979.

The Maximum Label Propagation Algorithm on Sparse Random Graphs

Charlotte Knierim
ETH Zurich, Switzerland
cknierim@inf.ethz.ch

Johannes Lengler
ETH Zurich, Switzerland
johannes.lengler@inf.ethz.ch

Pascal Pfister
ETH Zurich, Switzerland
ppfister@inf.ethz.ch

Ulysse Schaller
ETH Zurich, Switzerland
ulysses@student.ethz.ch

Angelika Steger
ETH Zurich, Switzerland
angelika.steger@inf.ethz.ch

Abstract

In the Maximum Label Propagation Algorithm (`Max-LPA`), each vertex draws a distinct random label. In each subsequent round, each vertex updates its label to the label that is most frequent among its neighbours (including its own label), breaking ties towards the larger label. It is known that this algorithm can detect communities in random graphs with planted communities if the graphs are very dense, by converging to a different consensus for each community. In [17] it was also conjectured that the same result still holds for sparse graphs if the degrees are at least $C \log n$. We disprove this conjecture by showing that even for degrees n^{ε}, for some $\varepsilon > 0$, the algorithm converges without reaching consensus. In fact, we show that the algorithm does not even reach almost consensus, but converges prematurely resulting in orders of magnitude more communities.

2012 ACM Subject Classification Mathematics of computing → Random graphs; Theory of computation → Distributed algorithms

Keywords and phrases random graphs, distributed algorithms, label propagation algorithms, consensus, community detection

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.58

Category RANDOM

1 Introduction

In the last years, opinion exchange dynamics on graphs and networks has received much attention, see [19] for an excellent survey. Apart from the desire to improve our understanding of social processes, opinion exchange dynamics have also found applications in the fields of distributed computing and network analysis. For example, opinion exchange dynamics like the 3-majority protocol or the 2-choice dynamics have been proposed as simple distributed solutions to the basic problems of consensus forming, majority detection, and plurality consensus in distributed networks [2, 3, 5, 7, 10, 12, 13].



© Charlotte Knierim, Johannes Lengler, Pascal Pfister, Ulysse Schaller, and Angelika Steger;
licensed under Creative Commons License CC-BY



Label propagation algorithms (LPA) are a certain kind of opinion exchange dynamics which have been used for community detection in networks [4, 14, 15, 21, 23]. Despite their great practical importance (see the surveys [4, 15, 23]), and although obtaining theoretical bounds on success and speed of LPAs was proposed as an important research question [1, 8, 18], rigorous theoretical analyses of LPAs have only appeared recently. The first such study by Kothapalli, Pemmaraju and Sardeshmukh [17] investigated an algorithm called `Max-LPA`. In this algorithm, each vertex starts with a random label in the interval $[0, 1]$. In each round, every vertex switches its label to the majority label in its neighbourhood (including its own label), breaking ties towards larger labels. In [17] `Max-LPA` was studied on an Erdős-Rényi model with planted communities.[1] In this model, the vertex set is partitioned as $V = V_1 \dot\cup \ldots \dot\cup V_k$, where all sets V_k have a certain minimal size. Then every edge inside of one of the sets V_i is inserted independently with some probability p_i, and every edge between different partite sets is inserted independently with probability $p' \ll p := \min_i\{p_i\}$. The study [17] considered dense cases, e.g. for $|V_i| = \Omega(n)$ they set $p = \Omega(n^{-1/4+\varepsilon})$ and $p' = O(p^2)$. For this case they show that `Max-LPA` successfully recovers the communities, i.e., it converges quickly to a state where for each i the labels within the set V_i are all identical, and any two distinct sets V_i have different labels. The authors of [17] conjectured that their conditions on p are very far from tight, and that it should suffice to require $p \geq C \log n/n$ (i.e., expected degrees of $C \log n$ instead of $\Omega(n^{3/4+\varepsilon})$), if there is a sufficient gap between p and p'.

The conjecture from [17] has been considered in at least two subsequent papers [6, 9], both of whom have tried to obtain results for LPA algorithms in sparser cases, see Section 1.1 below. However, the conjecture remained unresolved, and the behaviour of `Max-LPA` and other LPAs generally remained poorly understood in the sparse case. In this paper we show that the conjecture from [17] is actually false in a strong sense. Even in the extreme case of just one rather dense community, i.e. $p' = 0$, and $p = n^{-1+\varepsilon}$, `Max-LPA` fails to label the communities consistently. In fact, with high probability the algorithm gets stuck with all label classes having size $o(n)$. For this negative result, note that the case of just one community corresponds to running the `Max-LPA` process on a classical Erdős-Rényi graphs $G_{n,p}$, in which each edge is inserted independently with probability p. We thus phrase our main result for Erdős-Rényi graphs $G_{n,p}$ with expected degree $d := np \leq n^{\varepsilon}$.

▶ **Theorem 1.** *There exists constants $C > 0$ and $\varepsilon > 0$ such that for any $C \log n/n \leq p \leq n^{-1+\varepsilon}$, the `Max-LPA` process on an Erdős-Rényi graph $G_{n,p}$ terminates with $\Omega(d^3)$ different label classes each of size at most $O\left(n/d^3\right)$, where $d := np$ denotes the expected degree of a vertex in $G_{n,p}$.*

Note that our result does not immediately extend to smaller values of p, as the property "`Max-LPA` finds consensus on G" is not monotone.

A rigorous analysis of LPAs is challenging due to the high influence of dependencies. In [9] this was stated as: "The absence of substantial theoretical progress in the analysis of LPAs is largely due to the lack of techniques for handling the interplay between the non-linearity of the local update rules and the topology of the graph." In order to prove our result we need to combine local analysis of the dynamics with global properties of the graph, in particular by inventing a discharging technique for showing that no set of size $O(n/d^3)$ can propagate much further without the help of (suitably defined) unstable vertices. We hope that this technique will also turn out to be useful for the analysis of other LPAs.

[1] also called *clustered Erdős-Rényi graph* or *stochastic block model*

1.1 Other related work

There is a huge body of experimental work on LPAs for community detection, and we refer the reader to surveys [4, 15, 23]. Basic properties of `Max-LPA` were analysed in [20], and in particular it was shown that the algorithm always converges to a stable configuration or to a limiting cycle of length 2. An experimental comparison of `Max-LPA` with other tie-breaking techniques was performed in [8], with the conclusion that `Max-LPA` typically converges faster than other tie-breaking rules, but that repeated runs on the same graph produces slightly less consistent outcomes than other LPAs.

As outlined above, there are only very few rigorous mathematical results on LPAs. In an attempt to improve on results from [17], Cruciani, Natale, and Scornavacca [9] studied the 2-choices dynamics as a "sampling" approximation of an LPA. In this dynamics, a vertex updates its label to the majority label among its own and the labels of two randomly drawn neighbours, breaking ties towards its own label. For this variant of an LPA, [9] studied clustered regular graphs with two clusters, i.e., graphs with fixed degree d within each of the clusters, and fixed degree $d' \ll d$ between the clusters. While both, the algorithm and the setting, differ slightly from [17], the authors compare their results with [17] and improve the (average) degree d from $d = n^{3/4+\varepsilon}$ to $d = n^{1/2}$. The proof idea relies on expansion properties of the graph, and $n^{1/2}$ is a natural barrier for this approach.

In [6], Clementi et al. analysed the Erdős-Rényi model with two planted communities, for an LPA which uses a different tie-breaking rule than `Max-LPA`: it breaks ties randomly. For this variant they claim that a repeated application of this LPA with random tie breaking can successfully recover the communities with high probability, even for densities as sparse as $p = \Theta(1/n)$. However, their result has a catch: the analysis is only done for a *dynamic* version of the random graph model, in which all edges are re-drawn in every round. This simplifies the analysis considerably, since it removes dependencies between different rounds. In particular, the state of the algorithm at any time can be completely described by the *number* of vertices of each label in each of the partite sets, whereas in the static case the structural information is essential. As we show in this paper, the behaviour of the `Max-LPA` in the sparse case reveals that it fails precisely because of the structural properties of the label classes. In other words, the behaviour in the static and the dynamic case are completely different, and an analysis of the dynamic case in general will not explain the behaviour of the static case.

1.2 Some intuition on the `Max-LPA` process

In this section we provide some intuition for Theorem 1. Recall that we assume that, by definition of the `Max-LPA` process, each $v \in V$ chooses it original label uniformly and independently from $[0, 1]$. With high probability all vertices will therefore have different labels. Throughout our paper we will thus assume without loss of generality that in the beginning of the process the labels of all vertices are pairwise different.

Consider a random graph $G_{n,p}$ and recall that we denote by $d = pn$ its average degree. Then almost all vertices will have a neighbour which holds one of the n/d highest labels (here and in the sequel of this overview we ignore polylog(d) factors). After the first round we thus expect that (almost) all but the largest n/d labels are extinct, i.e., are not present any more at any vertex.

Let us thus have a closer look at how such a high label evolves in the first few rounds of the `Max-LPA` process. Hence, consider a vertex v with a label ℓ that belongs to the n/d largest labels. We expect that v pushes ℓ to some of its neighbours. Depending on the size

of ℓ, the number of neighbours which receive ℓ in round 1 can range from very few to almost all neighbours (but this is not our concern at the moment). In order to understand what can happen in subsequent rounds, we need to distinguish two cases: (i) v receives in round 1 a new label $\ell' > \ell$ or (ii) v keeps its initial label ℓ in round 1. In case (i), whenever the label ℓ got pushed onto enough neighbours of v in round 1, vertex v will get back its initial label ℓ in round 2. In this case quite a number of different scenarios can happen in further rounds, as for example v can receive back its initial label not only in round 2, but also in other subsequent rounds if the label ℓ was pushed down far enough into the r-neighbourhood of u.

In case (ii), on the contrary, v is already in a quite stable situation, as v and many of its neighbours have the same label. More precisely, in order for v to change its label in any of the subsequent rounds, (almost) all neighbours of v which receive label ℓ in round 1 need to lose ℓ again in one of the subsequent rounds. For any such neighbour $u \in N(v)$ this can either happen if, at some point in time, say t, the initial label of u is pushed back onto u or the neighbourhood of u will contain a different label $\ell' \neq \ell$ on two or more of its vertices. Note that, for most neighbours of v, the first case is unlikely, as the vast majority of neighbours of v initially held one of the $n - n/d$ labels which are extinct after the first round. The latter case, on the other hand, can only happen if u is in a cycle of length at most $2(t+1)$ (here we use that we assumed that in the beginning all vertices have pairwise different labels). Note that for each constant $t \in \mathbb{N}$ there exists a constant $\varepsilon = \varepsilon(t) > 0$ so that the random graph $G_{n,p}$ with $\omega(1/n) \leq p \leq n^{-1+\varepsilon}$ has the property that all but a negligible number of vertices are *not* contained in a cycle of length at most t. Hence, almost all vertices $v \in V$ which keep their label in round 1, together with their neighbours $u \in N(v)$ which receive the label of v in round 1 and initially held one of the $n - n/d$ smallest labels, are in a quite stable configuration already after the first round.

After the first round only a constant fraction of vertices are in such a stable configuration. However, by analysing the subsequent rounds of the process in a similar fashion we can show that after the first few rounds, all but n/d^4 vertices are in such a stable configuration.

Once we have shown that all but n/d^4 vertices are in stable configurations, we still have to argue that the n/d^4 non-stable vertices will not be able to break up these stable configurations. To handle this case we use a discharging argument to show that if a label class grows to size n/d^3, then this label class induces a subgraph with density at least $3/2$. As a random graph $G_{n,p}$ with $\log(n)/n \leq p \leq n^{\varepsilon}$ whp does not contain such a set, we deduce that no label class gets that large. We note that this idea is similar to techniques used to study k-bootstrap percolation on Erdős-Rényi graphs [11], in which an initial set of active vertices successively activates every vertex that has at least k active neighbours. Our case can be viewed as a (hypothetical) 3/2-bootstrap percolation.

1.3 Notation and terminology

Our graph-theoretic notation is standard and follows that of [22]. In particular, for a graph G, we denote by V and E the set of vertices and edges, respectively. Moreover, $e(G) := |E|$ is the number of edges of G. For any subset $S \subseteq V$ we let $G[S]$ denote the subgraph of G that is induced by the vertices of S. We denote by $E(S)$ the set of edges of $G[S]$ and define $e(S) := |E(S)|$. For any two disjoint subsets $S, T \subset V$ we denote by $E(S,T)$ the set of edges with one endpoint in S and the other in T and define $e(S,T) := |E(S,T)|$. For a vertex $v \in V$, we denote by $N(v)$ the neighbourhood of v, which excludes v, and by $d(v) := |N(v)|$ its degree. For any positive integer $r \geq 2$, the r-neighbourhood of a vertex v, denoted by $N^r(v)$, is the set of vertices that can be reached from v by a path of length at most r (i.e. the r-neighbourhood of a vertex v includes v).

We consider the classical random graph model from Erdős and Rényi. For a positive integer n and $0 \leq p := p(n) \leq 1$. We denote by $G_{n,p}$ the probability space over graphs on n vertices where every possible edge is present with probability p independently of all other edges. We write $d := np$ for the expected degree of a graph $G \sim G_{n,p}$ and $\Delta(G)$ for its maximum degree. We say that an event $\mathcal{E} = \mathcal{E}(n)$ happens *with high probability*, or *whp*, if $\Pr[\mathcal{E}(n)] \to 1$ for $n \to \infty$. We use the notation $\tilde{O}(.)$ to hide polylog(d) factors.

2 Some properties of random graphs

In this section, we provide some results on random graphs which are important in our analysis. First, we state a standard result from random graph theory on degree concentration (the proof follows e.g. from [16, Corollary 2.3] together with a union bound).

▶ **Lemma 2.** *For every $\varepsilon > 0$ there exists a positive constant C such that for $G \sim G_{n,p}$ with $p \geq \frac{C \log(n)}{n}$ with probability at least $1 - 2ne^{-\frac{\varepsilon^2 d}{3}}$, it holds for every vertex $v \in V$ that*

$$(1-\varepsilon)d \leq d(v) \leq (1+\varepsilon)d,$$

where $d = pn$.

The next lemma gives a lower bound on the number of vertices that do not have a neighbour in a large subset.

▶ **Lemma 3.** *Let $M > 0$, $\varepsilon > 0$, $\log(n)/n \leq p \leq n^{-1+\varepsilon}$ and $G = (V, E) \sim G_{n,p}$. Let $S \subset V$ be a subset of vertices of size $|S| = \Omega\left(\frac{n(\log(d))^2}{d}\right)$, where $d = pn$. Then whp all but at most nd^{-M} vertices $v \in V \setminus S$ have at least one neighbour in S.*

Proof. As each vertex $v \in V \setminus S$ has $|S|$ opportunities to have an edge into S we have that

$$\Pr\left[e(v, S) = 0\right] = (1-p)^{|S|} \leq e^{-\Omega((\log(d))^2)} = d^{-\Omega(\log(d))}.$$

Thus, letting X be the random variable which counts the number of vertices in $V \setminus S$ with no neighbour on S and writing X as a sum of indicator random variables, we can conclude that

$$\mathbb{E}[X] \leq nd^{-\Omega(\log(d))}.$$

Setting $t = nd^{-M} - \mathbb{E}[X] = \omega(\log(n))$, the proof now follows by Chernoff bounds. ◀

In the following we argue that there are not many vertices which are in short cycles.

▶ **Lemma 4.** *Let k be a positive integer and let $G \sim G_{n,p}$ with $p = \omega(1/n)$. Then whp the number of cycles of length at most k is less than d^{k+1}, where $d = pn$. Therefore, at most kd^{k+1} vertices are contained in such cycles.*

Proof. For $1 \leq i \leq k$ let X_i be a random variable counting the number of cycles of length i in G. Then the expectation of X_i is given by

$$\mathbb{E}[X_i] = \binom{n}{i} \frac{(i-1)!}{2} p^i \leq d^i \leq d^k,$$

where the last inequality follows as $d \geq 1$. Thus, letting $X := \sum_{i=1}^{k} X_i$ be the random variable counting the number of cycle of length at most k, we can conclude by linearity of expectation that

$$\mathbb{E}[X] \leq kd^k.$$

Using Markov's inequality we get

$$\Pr\left[X \geq d^{k+1}\right] \leq \frac{k}{d} = o(1)$$

which finishes the proof, as each cycle contains at most k vertices. ◀

3 Notation and terminology for the Max-LPA process

Let us assume that the Max-LPA process runs on a random graph $G \sim G_{n,p}$ with $C\log(n)/n \leq p \leq n^{-1+\varepsilon}$, for some suitable constants $C > 0$ and $\varepsilon > 0$. We first introduce some notation. We define $\mathcal{L}$ to be the set of labels used in the Max-LPA process and for any $t \geq 0$ we denote by $\ell_t(v)$ the label of a vertex $v \in V$ after the t-th round of the Max-LPA process. Recall that we assume without loss of generality that all labels $\ell_0(v)$ are distinct. For any label $\ell \in \mathcal{L}$ we denote by v_ℓ the (by our assumption unique) vertex from which label ℓ originated from, i.e. $\ell_0(v_\ell) = \ell$. Moreover, we call a label ℓ *extinct* in a round $t > 0$ if there exists no vertex $v \in V$ with $\ell_t(v) = \ell$.

During the Max-LPA process, we say that a vertex v *propagates* its label onto $u \in N(v)$ in the t-th round of the process if $\ell_{t-1}(v) = \ell_t(u)$ and $\ell_{t-1}(v) \neq \ell_{t-1}(u)$. If $\ell_i(u) \neq \ell_{t-1}(v)$ for all $i \leq t-1$ (i.e. if u never held the label $\ell_{t-1}(v)$ so far), we say that v *forward-propagates* the label $\ell_{t-1}(v)$ onto u in round t. Otherwise, we say that v *back-propagates* the label $\ell_{t-1}(v)$ onto u in round t. In abuse of notation, we will also call a label ℓ to be *forward-propagating* and *backward-propagating* from or onto a vertex v.

As we saw in the introduction, in the absence of (short) cycles, back-propagation is essentially the only way that a vertex which has a neighbour holding the same label can change its current label. Thus, in order to create stable configurations, we need to understand such back-propagation of labels properly. The following definition will help us do so.

▶ **Definition 5.** *Let $v \in V$, $t \geq 0$ and assume that ℓ is a label which v holds for the first time in round t, i.e. $\ell_t(v) = \ell$ and $\ell_i(v) \neq \ell$ for all $i < t$. Then we define the* ℓ-propagation set of v *to be the vertex v together with all vertices w for which there exists a path $v = v_0, \ldots, v_k = w$ such that for all $0 \leq i \leq k$ the vertex v_i receives label ℓ in round $t + i$ from v_{i-1} and did not hold it in any round $j < t + i$.*

Recall from Section 1.2 that two vertices u and v that are connected by an edge and that hold the same label in some round t can only change their label if they are in a short cycle or if a label is back-propagated. In the following definition we only capture the latter (as we will argue in Section 4 that we do not have to consider vertices that are in short cycles).

▶ **Definition 6.** *Let $t \geq 0$ and assume that the Max-LPA process has run for t rounds. For any $v \in V$ we denote by $L_v^{(t)} := \{\ell \in \mathcal{L} : \exists i \leq t$ such that $\ell_i(u) = \ell\}$ the set of labels v held in the first t rounds of the process. We then call an edge $\{u, v\} \in E$* stable *after round t if*

- **(i)** *$\ell_t(v) = \ell_t(u)$,*
- **(ii)** *for all forward-propagating labels $\ell \in L_v^{(t)} \setminus \{\ell_t(v)\}$ of v we have that no vertex in the ℓ-propagation set of v holds the label ℓ after round t, and*
- **(iii)** *for all forward-propagating labels $\ell \in L_u^{(t)} \setminus \{\ell_t(u)\}$ of u we have that no vertex in the ℓ-propagation set of u holds the label ℓ after round t.*

A vertex $v \in V$ is then called stable after round t *if it belongs to a stable edge after the t-th round. All other vertices are called* unstable after round t.

Moreover, we call an unstable vertex vulnerable in round $t+1$ *if the labels of all vertices in $N(v) \cup \{v\}$ are pairwise different after round t.*

Note that being stable a priori does not mean that a vertex will never change its label again. However, we can show:

▶ **Lemma 7.** *Let t_0 and t be positive integers with $t < t_0$ and let $u, v \in V$ be adjacent vertices. If $\{u, v\}$ is a stable edge at time t and neither u nor v belong to a cycle of length $2t_0$ or less then both u and v will keep their label (and thus remain stable vertices) until round t_0.*

Proof. Let $t \leq i < t_0$ and let us assume that $\{u, v\}$ is a stable edge in round i. We claim that then the same is true in round $i+1$ as well. Indeed, as u and v are in a stable edge and not contained in cycles of length less than $2t_0$, we have that any label $\ell \in L_u^{(t)} \setminus \ell_t(u)$ appears at most once in the neighbourhood of u namely on the unique vertex which pushed the label ℓ onto u. As the same is true for v as well, no label ℓ can be back-propagated onto u or v.

Hence, the only way for u or v to change their label is if they see a label ℓ in their neighbourhood at least twice. As these two neighbours cannot be connected (else u or v would be in a triangle), this can only happen if there exist two different paths from u or v to the vertex v_ℓ from which the label ℓ originated from. Hence, the appearance of the label ℓ in two neighbours of u (or v) in round $t \leq i < t_0$ implies the existence of a cycle of length at most $2i + 2 \leq 2t_0$ which contains u (or v). As this is a contradiction, we have that u and v will not change their label in the i-th round. Thus $\ell_{i+1}(u) = \ell_{i+1}(v)$. As furthermore, again since u and v are not contained in cycles of length less than $2t_0$, no label $\ell \in L_u^{(t)} \setminus \ell_t(u)$ (or $\ell \in L_v^{(t)} \setminus \ell_t(v)$) can reappear in the ℓ-propagation set of u (or v) through a path from v_ℓ not through u (or v), the edge $\{u, v\}$ remains stable as desired. The statement of the lemma now follows by a simple induction. ◀

4 The first two rounds of the process

In this section we carefully analyse the first two rounds of the process and show that after two rounds there are at most $\tilde{O}(n/d)$ unstable vertices left. As explained in the introduction, we are mainly interested in the behaviour of the highest labels, hence we define the following.

▶ **Definition 8.** *For a label $\ell \in \mathcal{L}$, let the rank of ℓ be $\mathrm{rk}(\ell) := |\{\ell' \in \mathcal{L} \mid \ell' \leq \ell\}|$. In particular, the smallest label has rank 1, and the largest label has rank n. Then we define*

$$L_X = \left\{ \ell \in \mathcal{L} \colon \mathrm{rk}(\ell) \geq n \left(1 - \frac{(\log(d))^2}{d^3}\right) \right\},$$

$$L_Y = \left\{ \ell \in \mathcal{L} \colon n \left(1 - \frac{(\log(d))^2}{d^2} - \frac{(\log(d))^2}{d^3}\right) \leq \mathrm{rk}(\ell) < n \left(1 - \frac{(\log(d))^2}{d^3}\right) \right\},$$

$$L_Z = \left\{ \ell \in \mathcal{L} \colon n \left(1 - \frac{(\log(d))^2}{d}\right) \leq \mathrm{rk}(\ell) < n \left(1 - \frac{(\log(d))^2}{d^2} - \frac{(\log(d))^2}{d^3}\right) \right\}.$$

Additionally, we denote by $X^{(t)}$, $Y^{(t)}$ and $Z^{(t)}$ the sets of vertices holding labels in L_X, L_Y and L_Z after round $t \geq 0$, respectively.

We are particularly interested in vertices that propagate their label to at least one neighbour in the first round but also lose their label in this round. Those are vertices which initially engage in back-propagation. As this can cause a series of troubles, we call those vertices and their labels "bad". More precisely, we define the following:

▶ **Definition 9.** *A label ℓ is called* good *if $\ell_1(v_\ell) = \ell$ and v_ℓ forward-propagates label ℓ in the first round to at least one of its neighbours. Otherwise, we call ℓ* bad.

As not only vertices which initially hold a bad label can cause troubles, but also all vertices which (potentially) hold such a bad label in later rounds, we also consider the 2-neighbourhood of such vertices.

▶ **Definition 10.** *We denote by*

$$X^{(2)}_{\text{bad}} := \bigcup_{\substack{\ell \in L_X \\ \ell \text{ is bad}}} N^2(v_\ell)$$

the set of all vertices that are in the 2-neighbourhood of a vertex $v \in X^{(0)}$ initially holding a bad label. Similarly, we define

$$Y^{(2)}_{\text{bad}} := \bigcup_{\substack{\ell \in L_Y \\ \ell \text{ is bad}}} N^2(v_\ell).$$

Additionally, for any bad label $\ell \in L_X$ or $\ell \in L_Y$ we call the 2-neighbourhood $N^2(v_\ell)$ an X_{bad}-set *and a* Y_{bad}-set*, respectively.*

In the later rounds the following situation can also arise. Whenever we have a vulnerable vertex, it can propagate its label and get a new label in the same round. This again results in vertices which engage in back-propagation. The following rather general definition will later allow us to capture all such situations. The definition may remain a bit mysterious at first glance, but we will show later why it makes sense (see Section 5).

▶ **Definition 11.** *We define two sets set of vertices, called* A-nodes *and* B-nodes*, as follows:*

$$A := \bigcup_{v \in X^{(0)}} \bigcup_{w \in N(v) \cap Z^{(0)}} (N(w) \setminus \{v\})$$
$$B := \bigcup_{v \in Y^{(0)}} \bigcup_{w \in N(v) \cap Z^{(0)}} (N(w) \setminus \{v\})$$

With the above definitions at hand, we can state our first main lemma, which summarises the behaviour of the algorithm in the first two rounds.

▶ **Lemma 12.** *For every $M > 0$ there exist $C > 0$ and $\varepsilon > 0$ such that the following holds. Let $\frac{C \log n}{n} \leq p \leq n^{-1+\varepsilon}$ and assume that we run the* `Max-LPA` *process on a graph $G \sim G_{n,p}$. Then with high probability there exists a set $D \subseteq V$ of size at most nd^{-M} such that the following statements hold:*

(ACYC)	*All cycles in $G[V \setminus D]$ have length larger than* 200*.*
(NBZ)	*Every vertex in $V \setminus D$ has at least one neighbour in $Z^{(0)}$. Moreover, all vertices in $V \setminus D$ which forward-propagate in round 1 hold a label from L_X, L_Y or L_Z.*
(NBY)	*Every vertex in $V \setminus D$ has at least one neighbour in $Y^{(1)}$. Moreover, all vertices in $V \setminus D$ which forward-propagate in round 2 hold a label from L_X or L_Y.*
(NBX)	*Every vertex in $V \setminus D$ has at least one neighbour in $X^{(2)}$. Moreover, all vertices in $V \setminus D$ which forward-propagate in round 3 hold a label from L_X.*
(UNST-TYPES2)	*After the second round, every unstable vertex in $V \setminus N(D)$ is in $Z^{(0)}$, A, B, $X^{(2)}_{\text{bad}}$ or $Y^{(2)}_{\text{bad}}$.*

(UNST2X) *There are $\tilde{O}(n/d^3)$ vertices in $X^{(2)}_{\text{bad}}$.*

(UNST2Y) *There are $\tilde{O}(n/d)$ vertices in $Y^{(2)}_{\text{bad}}$.*

(UNST2Z) *There are $\tilde{O}(n/d)$ vertices in $Z^{(0)}$.*

(UNST2A) *There are $\tilde{O}(n/d^2)$ vertices in A.*

(UNST2B) *There are $\tilde{O}(n/d)$ vertices in B.*

(UNST2) *There are $\tilde{O}(n/d)$ unstable vertices after the second round.*

As the proof of Lemma 12 is rather long, we only give a short overview. As a first step one can argue that the labels from L_Y and L_X propagate their labels to a linear fraction of their 1-Neighbourhood and 2-Neighbourhood, respectively. I.e. one can show that both $Y^{(1)}$ and $X^{(2)}$ are of size $\Theta(n(\log(d))^2/d)$. Thus, by Lemma 3 and Lemma 4, we can define D as the 5-Neighbourhood of all vertices which do not have a neighbour in $Z^{(0)}$ or $Y^{(1)}$ or $X^{(2)}$ and which are contained in cycles of length 200 or less, and the first four items of Lemma 12 then follow easily.

The proof of (UNST-TYPES2) is a rather involved case analysis. In a nutshell, one can show that all vertices $v \in V \setminus N(D)$ which are not contained in $Z^{(0)}$, A, B, $X^{(2)}_{\text{bad}}$ or $Y^{(2)}_{\text{bad}}$ are connected through a stable path of length at most two to a vertex v_ℓ for some good label ℓ.

To show items (UNST2X) to (UNST2B) one can calculate the number of bad labels in L_X and L_Y using standard probabilistic tools (as e.g. Chernoff bounds). The sizes of these five sets then simply follow from their definition and Lemma 2. Last, note that (UNST2) is just a summary of the other statements of Lemma 12.

5 The next rounds of the Max-LPA process

The main goal of this section is to show the following proposition.

▶ **Proposition 13.** *After at most* 100 *rounds, whp the number of unstable vertices is $\tilde{O}(n/d^4)$.*

In the remainder of the paper, we will mostly neglect vertices which are outside the 101-neighbourhood of D. Since $|N^{101}(D)| = o(n/d^4)$ (if we choose the constant M in Lemma 12 large enough) the above proposition also follows if we only show that a $1-\tilde{O}(n/d^4)$ fraction of vertices in $V \setminus N^{101}(D)$ is stable after round 100. The main advantage of neglecting these vertices is that all vertices we consider in this section, and their complete 100-neighbourhood, are not contained in cycles of length less than 200. Thus, all structures we analyse (such as X_{bad}-sets, Y_{bad}-sets and ℓ-propagation sets, label classes) are actually trees (and in the following, we hence also refer to them as e.g. ℓ-propagation trees instead of ℓ-propagation sets). To ease notation, we henceforth denote for any set $S \subseteq V$ by $\hat{S} := S \setminus N^{101}(D)$ the set S without the 100-neighbourhood of D.

As the proof of the above proposition is quite involved, we only provide a detailed overview.

By Lemma 12, we know that after the second round a $1 - \tilde{O}(1/d)$ fraction of the vertices in $\hat{V}$ is already stable. In order to argue that the number of unstable vertices drops even further over the next few rounds, we cover all unstable vertices in $\hat{V}$ by the five classes $\hat{X}_{\text{bad}}$, $\hat{Y}_{\text{bad}}$, $\hat{Z}^{(0)}$, $\hat{A}$ and $\hat{B}$, depending on the mechanism that keep them unstable. Note that the definitions of $\hat{X}_{\text{bad}}$, $\hat{Y}_{\text{bad}}$, $\hat{Z}^{(0)}$, $\hat{A}$ and $\hat{B}$ are a bit over-pessimistic: vertices may occur in

several classes, or several times in the same class, and all classes may also contain some stable vertices. However, our definitions allow us to show that the vertices in each class behave as if they were randomly distributed. In particular, for a given vertex, say, of type $\hat{X}_{\text{bad}}$, it is easy to count the number of neighbours in other $\hat{X}_{\text{bad}}$-structures, in $\hat{Y}_{\text{bad}}$-structures, and so on.

The weighted meta-graph in Figure 1 summarizes the situation after round 2. Each vertex in the meta-graph corresponds to one (or two) of the five classes of unstable vertices (we depict $\hat{Z}^{(0)}$ and $\hat{B}$ as one vertex in all our figures because they evolve identically). The weights on the edges in the meta-graph are guided by the following idea. Assume that a vertex is, say, in an $\hat{X}_{\text{bad}}$-structure, and assume pessimistically that its label can take over arbitrary parts of this structure. Then the weights of the arrows going out from $\hat{X}_{\text{bad}}$ are (an upper bound for) the expected number of structures of other types that the label sees (and thus, can potentially take over). If we pessimistically assume that it also takes over these new structures, then the weights of the outgoing edges in the meta-graph also gives a bound on the expected number of structures that the label can see form there, and so on. Thus if for each walk in the meta-graph we multiply the weights along a walk, and then sum these products over all walks in the meta-graph, then we obtain an a bound for the expected number of structures that the label can take over.

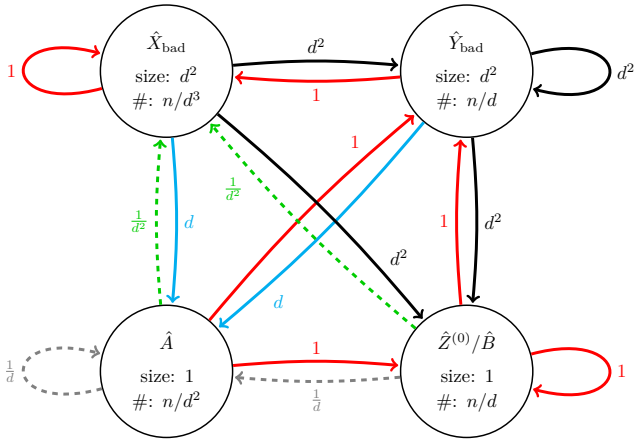


Figure 1 Meta-graph after round 2. The first value in a node S indicates the size of a structure S, the second value indicates number of vertices (counted with multiplicity) in structures of type S. The weight x of a meta-edge from node S to node T indicates that from a random S-structure S_0 there are in expectation at most x edges to T-structures (not counting edges within S_0 if $S = T$). The same bound is also valid if S_0 is not random among all S-structures, but rather a random neighbour of a T'-structure, for any type T' that appears in the meta-graph. All values are upper bounds and suppress any polylog(d) factors.

An important intermediate goal is to show that the graph $G' = (V', E')$ induced by unstable vertices from $\hat{V}$ is scattered, i.e. that most vertices are in small components. To this end, we would like to have that for any walk in the meta-graph, the weights of the edges multiply up to something of size $o(1)$. Then, if we start with a random vertex in V', all paths in G' containing this vertex are short, since its unstable structure is only connected to few other meta-vertices. Hence, the vertex has small probability to be in a large component. Unfortunately, after round 2 the meta-graph is still much too dense for our purpose.

Analysing the `Max-LPA` process further, we show that in the third round the number of unstable $\hat{Z}^{(0)}$ and $\hat{B}$ structures decreases by a factor of $\tilde{O}(1/d)$. For the $\hat{Y}_{\text{bad}}$ structures, something even more dramatic happens: most of them scatter internally by round 5, in the

sense that if we choose a random vertex in a random $\hat{Y}_{\text{bad}}$ structure, then the expected size of the component of this vertex within the $\hat{Y}_{\text{bad}}$ structure shrinks to $\tilde{O}(1)$. The analysis after round 2 becomes more tricky, since the structures lose their independence. For example, although the number of $\hat{Z}^{(0)}/\hat{B}$-structures decreases, the weight of the arrow $\hat{A} \to \hat{Z}^{(0)}/\hat{B}$ does not decrease because neighbours of $\hat{A}$-vertices are biased towards remaining unstable. However, we can prove that some weights decrease as one would expect for random edges, namely for the edges between $\hat{Y}_{\text{bad}}$ and $\hat{Z}^{(0)}/\hat{B}$, and the edges that go out of $\hat{Y}_{\text{bad}}$. The meta-graph after round 5 can be found in Figure 2.

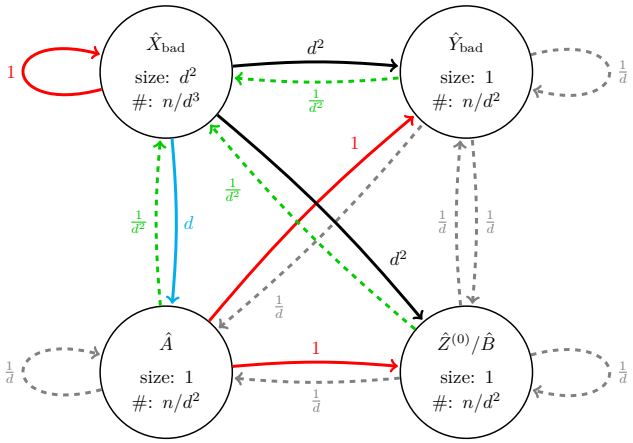


Figure 2 Meta-graph after round 5. The first value in a node S indicates, given a random vertex v in a random structure S_0 of type S, the size of the connected component of v induced by unstable vertices in S_0. The weight of the edge from S to T indicates how many edges into T-structures there are in expectation from this connected component. Again, all bounds remain valid if we choose v as a random vertex as seen from structures of some type T'. The second value in a node S indicates the number of unstable vertices in structures of type S. All values are upper bounds and suppress any polylog(d) factors. Compared to round 2, only a $\tilde{O}(1/d)$-fraction of the nodes in $\hat{Y}_{\text{bad}}$ and in $\hat{Z}^{(0)}/\hat{B}$ remain unstable. This decreases the weights from Y_{bad} and Z/B into Y_{bad} and Z/B by a factor of $1/d$, but not the weights from $\hat{X}_{\text{bad}}$ or $\hat{A}$ into these sets, because the target nodes of these edges are biased towards remaining unstable. Moreover, the remaining $\hat{Y}_{\text{bad}}$-structures are scattered into connected components of expected size $\tilde{O}(1)$, which decreases the weights of all outgoing edges from $\hat{Y}_{\text{bad}}$ by $1/d^2$. There are no cycles in which the weights multiply to strictly more than one, and every cycle with a product of one is a combination of the cycles $\hat{X}_{\text{bad}} \to \hat{X}_{\text{bad}}$, $\hat{X}_{\text{bad}} \leftrightarrow \hat{Y}_{\text{bad}}$ and $\hat{X}_{\text{bad}} \leftrightarrow \hat{Z}^{(0)}/\hat{B}$.

After round 5, there are still some cycles for which the weights do not multiply to $o(1)$, in particular the loop at $\hat{X}_{\text{bad}}$ and the cycles $\hat{X}_{\text{bad}} \leftrightarrow \hat{Z}^{(0)}/\hat{B}$ and $\hat{X}_{\text{bad}} \leftrightarrow \hat{Y}_{\text{bad}}$. To deal with those, we use the fact that a typical leaf v of an $\hat{X}_{\text{bad}}$-structure sees neighbours in $\hat{Y}_{\text{bad}}$ and in $\hat{Z}^{(0)}/\hat{B}$ after round 2, but these do not see further unstable neighbours in round 3 and 4 with probability $1 - \tilde{O}(1/d)$. In this case, we show that the leaf v stabilizes by round 20, and we may decrease the weights from $\hat{X}_{\text{bad}}$ to $\hat{Y}_{\text{bad}}$ and to $\hat{Z}^{(0)}/\hat{B}$ accordingly (Figure 3). In the remaining meta-graph, all walks accumulate decreasing factors except for the loop for $\hat{X}_{\text{bad}}$. However, since there are only $\tilde{O}(n/d^4)$ vertices in $\hat{X}_{\text{bad}}$ structures, we can show that even the union of the connected components in G' of all these structures has size $\tilde{O}(n/d^4)$. For all vertices outside of this union, a long path in G' from such a vertex induces a long walk in the meta-graph, which is unlikely. Thus almost all vertices are either among the $\tilde{O}(n/d^4)$ vertices of $\hat{X}_{\text{bad}}$ components, or are in components of diameter less than K, for a suitable constant K. For the latter one, we show that they stabilise after K further rounds if the vertices are not contained in any cycles of length at most $2K$.

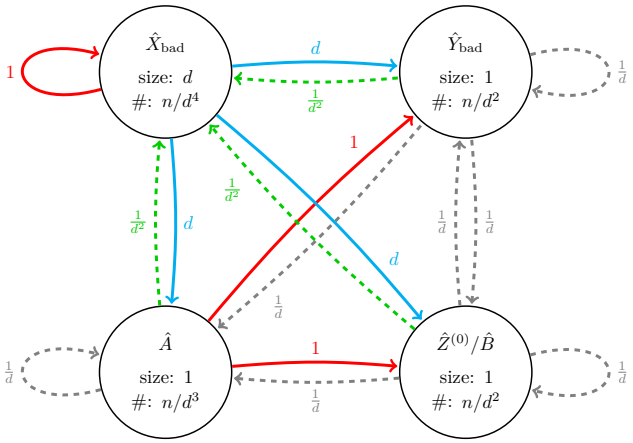


Figure 3 Meta-graph after round 20. Values have the same meaning as in Figure 2. Compared to Figure 2, the X_{bad}-structures have decreased by a $1/d$ factor in component size and in the number of unstable vertices, and the number of outgoing edges from $\hat{X}_{\text{bad}}$ to $\hat{Y}_{\text{bad}}$ and to $\hat{Z}^{(0)}/\hat{B}$ has decreased by a factor of $1/d$. The only remaining cycle in which the labels multiply to one is the loop at $\hat{X}_{\text{bad}}$.

6 Finishing the proof

For the last part of the proof, we fix a label, and show that this label cannot take over the complete graph. We only give the proof under the following simplifying assumption, and defer the full proof to the journal version. More precisely, we will assume that the label is $\ell_{\max}$, the maximum label, and after round 100 we change the labels of all unstable vertices and all vertices whose label appears in a cycle of length at most 200 to $\ell_{\max}$. This gives the label class of $\ell_{\max}$ a considerable boost after round 100, but it also simplifies the setting. In particular, since every other vertex v is stable and not in a cycles of length at most 200, the definition of stable implies that all neighbours of v have either the label of v, the label $\ell_{\max}$, or other mutually distinct labels. Moreover, after round 100 they have at least one neighbour of the same label, so they can only change their label to $\ell_{\max}$. This remains true inductively, since if a vertex v loses its neighbours of the same label, then those neighbours change their label to $\ell_{\max}$, and thus v also changes its label to $\ell_{\max}$. Thus the only possible change in the remaining graph is that vertices change their label to $\ell_{\max}$.

Let us first estimate the number of vertices that have or receive label $\ell_{\max}$ after round 100. There are at most $\tilde{O}(n/d^4)$ unstable vertices by Lemma 13. Moreover, at this point no label class has swallowed more than its 100-neighbourhood, which has size $O(d^{100}) = \tilde{O}(n/d^4)$ whp (Lemma 2). Consider some $\delta > 0$. By choosing $\varepsilon = \varepsilon(\delta)$ sufficiently small, if follows from Lemma 4 that the number of vertices that are contained in cycles of length at most 200 is $O(n^\delta)$. Since each label class has at most size $O(d^{100})$, the number of vertices with labels that appear in such cycles is $O(d^{100}n^\delta) = \tilde{O}(n/d^4)$, if we choose δ sufficiently small. Thus after round 100 the label class of $\ell_{\max}$ has size $\tilde{O}(n/d^4)$. In the following, we will show that with high probability, the structure of $G_{n,p}$ is such that no set of this size can take over all the stable vertices in the graph.

First we argue that in order to take over a certain set of stable vertices S from one of the stable trees, there needs to be a certain number of edges going from S to the vertices holding label $\ell_{\max}$. Let $T \subseteq V$ be the set of vertices that initially (i.e., after the relabelings in round 100) have label $\ell_{\max}$, and denote for each $\ell \neq \ell_{\max}$ by V_ℓ the set of vertices with

label ℓ at this time. Now fix some later point in time t. Let $T' \supseteq T$ be the set of vertices with label $\ell_{\max}$ at round t, and for each $\ell \neq \ell_{\max}$, let $S_\ell = V_\ell \cap T'$ be the set of vertices with label ℓ that have been taken over by $\ell_{\max}$ by round t. Then we claim that

$$e(T' \setminus T, T) + e(T' \setminus T) \geq |T' \setminus T| + \sum_{\ell \in \mathcal{L}, \ell \neq \ell_{\max}} (e(S_\ell, V_\ell \setminus S_\ell) + e(S_\ell)). \tag{1}$$

To prove (1), let us assume that $v_1, v_2, \dots, v_k$ is the order in which the vertices of $T' \setminus T$ acquire the label $\ell_{\max}$, where we break ties arbitrarily. For an index $i \leq k$, let ℓ_i be the label of v_i and let $T_i := T \cup \{v_1, \dots, v_{i-1}\}$. Note that

$$e(v_i, T') = e(v_i, T_i) + e(v_i, T' \setminus T_i).$$

Moreover, when v_i changes its label then all vertices in $V_{\ell_i} \setminus T_i$ still have label ℓ_i. Hence, v_i can only change its label if $e(v_i, T_i) \geq e(v_i, V_{\ell_i} \setminus T_i) + 1$, where the "+1" comes from the fact that the vertex v considers its own label as well when taking the majority. Hence,

$$\begin{aligned} e(v_i, T') &= e(v_i, T_i) + e(v_i, T' \setminus T_i) \\ &\geq e(v_i, V_{\ell_i} \setminus T_i) + 1 + e(v_i, T' \setminus T_i) \\ &= 1 + e(v_i, V_{\ell_i}) - e(v_i, \{v_1, \dots, v_{i-1}\} \cap V_{\ell_i}) + e(v_i, T' \setminus T_i). \end{aligned}$$

Now we sum both sides over all $1 \leq i \leq k$. Note that summing over $e(v_i, T')$ yields $e(T' \setminus T, T) + 2e(T' \setminus T)$ since edges in $T' \setminus T$ are counted twice. Likewise, summing over $e(v_i, V_{\ell_i})$ yields $\sum_{\ell \neq \ell_{\max}} (e(S_\ell, V_\ell \setminus S_\ell) + 2e(S_\ell))$, and summing over $e(v_i, \{v_1, \dots, v_{i-1}\} \cap V_{\ell_i})$ yields $\sum_{\ell \neq \ell_{\max}} e(S_\ell)$. Finally, summing over $e(v_i, T' \setminus T_i)$ yields $e(T' \setminus T)$. Thus, summing and canceling $e(T' \setminus T)$ yields

$$e(T' \setminus T, T) + e(T' \setminus T) \geq k + \sum_{\ell \in \mathcal{L}, \ell \neq \ell_{\max}} (e(S_\ell, V_\ell \setminus S_\ell) + e(S_\ell)),$$

which implies (1) as $k = |T' \setminus T|$.

Note that the term $e(T' \setminus T, T) + e(T' \setminus T)$ on the left hand side counts the number of edges by which the label class of $\ell_{\max}$ increases when it grows from T to T', while the term $e(S_\ell, V_\ell \setminus S_\ell) + e(S_\ell)$ counts the number of edges within V_ℓ which have at least one endpoint in S_ℓ. So basically (1) says that in order to recruit k vertices, the label class is "charged" at least $k + \sum_{\ell \neq \ell_{\max}} e(S_\ell, V_\ell \setminus S_\ell) + e(S_\ell)$ edges. It is easy to check that the minimal ratio of charged edges per recruited vertex is attained if $S_\ell = V_\ell$ is of size 2 for all $\ell \neq \ell_{\max}$, in which case the ratio is $3/2$ (three edges for two vertices).

Hence, in order to take over a set S of size k we need at least $3k/2$ edges in $S \cup V_{\ell_{\max}}$. However, a sparse $G_{n,p}$ does not have sets of this density of order $\Theta(n/d^3)$, as the following lemma shows.

▶ **Lemma 14** (Lemma 4.2 in [11]). *Consider $G_{n,p}$ with $1 \ll d = np = o(n)$. Let $\beta = \frac{3}{2} - \frac{1}{2\log(d)}$, and set $s = \frac{n}{3d^3}$. Then with high probability no set of s vertices spans at least βs edges.*

We may thus conclude the proof by contradiction as follows. Assume that $\ell_{\max}$ would take over the graph. After round 100, it has size $s_0 = \tilde{O}(n/d^4)$, so at some later point the label class must have size $s = n/(2d^3)$. At this point, the number of edges in the label class is at least $\frac{3}{2}(s - s_0) = (1 - \tilde{O}(1/d))\frac{3}{2}s \geq \beta s$, where β is as in Lemma 14. This is a contradiction to Lemma 14. Hence, the assumption must be wrong and $\ell_{\max}$ cannot take over the graph. In fact, the proof shows that the label class of $\ell_{\max}$ cannot grow to any size larger than $O(n/d^3)$.

7 Conclusion

We have shown that `Max-LPA` does not reach consensus on $G_{n,p}$ if $p = O(n^{-1+\varepsilon})$. Consequently it fails to identify communities in planted network models. This disproves a conjecture by Kothapalli, Pemmarajum, and Sardeshmukh. Our result is obtained by combining a careful local analysis of the process with suitable global properties of the network.

For the `Max-LPA` process, it is natural to assume that there is some threshold α such that for any small $\delta > 0$ we have that for $p = \Omega(n^{-\alpha+\delta})$ the `Max-LPA` process reaches consensus on $G_{n,p}$ with high probability, while for $p = O(n^{-\alpha-\delta})$ it does not reach consensus with high probability. Assuming such an α exists, it follows from our result that $\alpha \leq 1 - \varepsilon$; from [17] we know that $\alpha \geq 1/4$.

We conducted some experiments, which seem to suggest $\alpha = 4/5$. Our experimental data was obtained by doing a binary search where in every step we ran the algorithm on 32 independent $G_{n,p}$. If the majority of the runs converged with a unique label (modulo isolated vertices) then we decreased the value of p for the following run, otherwise we increased it. We stopped when the change in the probability was small enough. To visualize the data we plot it on a log-log-scale, with basis 2 for the log. In this setting the exponent becomes a linear factor. We computed a linear regression of the log-log-data (i.e., the line which minimizes the sum of the square-distances of the log-log data points), and obtained the line $\log_2(d) = 0.19964 \log_2(n) + 0.4652$. Since the leading constant is very close to 0.2, this suggests that the correct threshold might be $d = \Theta(n^{1/5})$ and thus $p = \Theta(n^{-4/5})$.

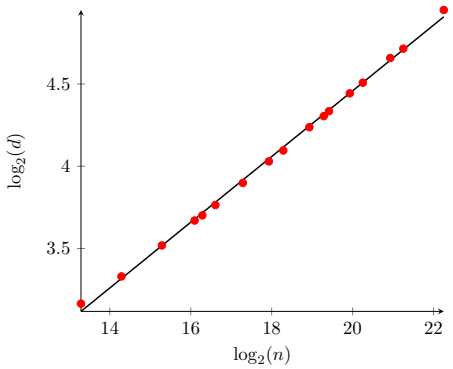


Figure 4 Our experimental data with a linear regression. We plot an experimental evaluation of the threshold d such that the `Max-LPA` converges in 50% of the cases with a unique label, on a log-log scale. The experimental data is extremely well described by a line with slope ≈ 0.2, suggesting that the threshold satisfies $d = \Theta(n^{1/5})$.

References

1 Michael J Barber and John W Clark. Detecting network communities by propagating labels under constraints. *Physical Review E*, 80(2):026129, 2009.

2 Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, Riccardo Silvestri, and Luca Trevisan. Simple dynamics for plurality consensus. *Distributed Computing*, 30(4):293–306, 2017.

3 Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Luca Trevisan. Stabilizing consensus with many opinions. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 620–635. SIAM, 2016.

4 Punam Bedi and Chhavi Sharma. Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 6(3):115–135, 2016.

5 Petra Berenbrink, Andrea Clementi, Robert Elsässer, Peter Kling, Frederik Mallmann-Trenn, and Emanuele Natale. Ignore or comply? on breaking symmetry in consensus. *arXiv preprint*, 2017. arXiv:1702.04921.

6 Andrea Clementi, Miriam Di Ianni, Giorgio Gambosi, Emanuele Natale, and Riccardo Silvestri. Distributed Community Detection in Dynamic Graphs. *Theor. Comput. Sci.*, 584(C):19–41, June 2015. doi:10.1016/j.tcs.2014.11.026.

7 Colin Cooper, Tomasz Radzik, Nicolás Rivera, and Takeharu Shiraga. Fast plurality consensus in regular expanders. *arXiv preprint*, 2016. arXiv:1605.08403.

8 Gennaro Cordasco and Luisa Gargano. Label propagation algorithm: a semi-synchronous approach. *International Journal of Social Network Mining*, 1(1):3–26, 2012.

9 Emilio Cruciani, Emanuele Natale, and Giacomo Scornavacca. On the Metastability of Quadratic Majority Dynamics on Clustered Graphs and its Biological Implications. *CoRR*, abs/1805.01406, 2018. arXiv:1805.01406.

10 Robert Elsässer, Tom Friedetzky, Dominik Kaaser, Frederik Mallmann-Trenn, and Horst Trinker. Efficient k-party voting with two choices. *ArXiv e-prints*, 2016.

11 Uriel Feige, Michael Krivelevich, and Daniel Reichman. Contagious sets in random graphs. *The Annals of Applied Probability*, 27(5):2675–2697, 2017.

12 Mohsen Ghaffari and Johannes Lengler. Nearly-tight analysis for 2-choice and 3-majority consensus dynamics. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, pages 305–313. ACM, 2018.

13 Mohsen Ghaffari and Merav Parter. A polylogarithmic gossip algorithm for plurality consensus. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, pages 117–126. ACM, 2016.

14 Steve Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12(10):103018, 2010.

15 Steve Harenberg, Gonzalo Bello, L Gjeltema, Stephen Ranshous, Jitendra Harlalka, Ramona Seay, Kanchana Padmanabhan, and Nagiza Samatova. Community detection in large-scale networks: a survey and empirical evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(6):426–439, 2014.

16 S. Janson, T. Łuczak, and A. Rucinski. *Random graphs*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience, New York, 2000.

17 Kishore Kothapalli, Sriram V. Pemmaraju, and Vivek Sardeshmukh. On the Analysis of a Label Propagation Algorithm for Community Detection. In *Distributed Computing and Networking*, pages 255–269, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

18 Ian XY Leung, Pan Hui, Pietro Lio, and Jon Crowcroft. Towards real-time community detection in large networks. *Physical Review E*, 79(6):066107, 2009.

19 Elchanan Mossel and Omer Tamuz. Opinion exchange dynamics. *Probability Surveys*, 14:155–204, 2017.

20 Svatopluk Poljak and Miroslav Sůra. On periodical behaviour in societies with symmetric influences. *Combinatorica*, 3(1):119–121, 1983.

21 Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.

22 Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

23 Zhao Yang, René Algesheimer, and Claudio J Tessone. A comparative analysis of community detection algorithms on artificial networks. *Scientific Reports*, 6:30750, 2016.

Samplers and Extractors for Unbounded Functions

Rohit Agrawal
John A. Paulson School of Engineering and Applied Sciences,
Harvard University, Cambridge, MA 02138, USA
https://rohitagr.com
rohitagr@seas.harvard.edu

Abstract

Błasiok (SODA'18) recently introduced the notion of a subgaussian sampler, defined as an averaging sampler for approximating the mean of functions $f : \{0,1\}^m \to \mathbb{R}$ such that $f(U_m)$ has subgaussian tails, and asked for explicit constructions. In this work, we give the first explicit constructions of subgaussian samplers (and in fact averaging samplers for the broader class of subexponential functions) that match the best known constructions of averaging samplers for $[0,1]$-bounded functions in the regime of parameters where the approximation error ε and failure probability δ are subconstant. Our constructions are established via an extension of the standard notion of randomness extractor (Nisan and Zuckerman, JCSS'96) where the error is measured by an arbitrary divergence rather than total variation distance, and a generalization of Zuckerman's equivalence (Random Struct. Alg.'97) between extractors and samplers. We believe that the framework we develop, and specifically the notion of an extractor for the Kullback–Leibler (KL) divergence, are of independent interest. In particular, KL-extractors are stronger than both standard extractors and subgaussian samplers, but we show that they exist with essentially the same parameters (constructively and non-constructively) as standard extractors.

2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors; Theory of computation → Pseudorandomness and derandomization; Mathematics of computing → Information theory

Keywords and phrases averaging samplers, subgaussian samplers, randomness extractors, Kullback–Leibler divergence

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.59

Category RANDOM

Related Version The full version of this paper is available at https://arxiv.org/abs/1904.08391 [1].

Funding *Rohit Agrawal*: Supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program.

Acknowledgements The author would like to thank Jarosław Błasiok for suggesting the problem of constructing subgaussian samplers and for helpful discussions and feedback, Salil Vadhan for many helpful discussions and his detailed feedback on this writeup, and the anonymous reviewers for their helpful comments and feedback.

1 Introduction

1.1 Averaging samplers

Averaging (or oblivious) samplers, introduced by Bellare and Rompel [6], are one of the main objects of study in pseudorandomness. Used to approximate the mean of a $[0,1]$-valued function with minimal randomness and queries, an averaging sampler takes a short random seed and produces a small set of correlated points such that any given $[0,1]$-valued function will (with high probability) take approximately the same mean on these points as on the entire space. Formally,


© Rohit Agrawal;
licensed under Creative Commons License CC-BY



▶ **Definition 1.1** ([6]). *A function* $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^D$ *is a* (δ, ε) averaging sampler *if for all* $f : \{0,1\}^m \to [0,1]$*, it holds that*

$$\Pr_{x \sim U_n}\left[\left|\frac{1}{D}\sum_{i=1}^{D} f(\mathrm{Samp}(x)_i) - \mathbb{E}[f(U_m)]\right| > \varepsilon\right] \leq \delta,$$

where U_n *is the uniform distribution on* $\{0,1\}^n$*. The number* n *is the* randomness complexity *of the sampler, and* D *is the* sample complexity*. A sampler is* explicit *if* $\mathrm{Samp}(x)_i$ *can be computed in time* $\mathrm{poly}(n, m, \log D)$*.*

Traditionally, averaging samplers have been used in the context of randomness-efficient error reduction for algorithms and protocols, where the function f is the indicator of a set ($\{0,1\}$-valued), or more generally the acceptance probability of an algorithm or protocol ($[0,1]$-valued). There has been significant effort in the literature to establish optimal explicit and non-explicit constructions of samplers, which we summarize in Table 1. We recommend the survey of Goldreich [17] for more details, especially regarding non-averaging samplers[1].

Table 1 Best known constructions of averaging samplers for $[0,1]$-valued functions.

Key Idea	Randomness complexity n	Sample complexity D	Best regime
Pairwise-independent Expander Neighbors [19]	$m + O(\log(1/\delta) + \log(1/\varepsilon))$	$O\left(\frac{1}{\delta\varepsilon^2}\right)$	$\delta = \Omega(1)$
Ramanujan Expander Neighbors[a] [22, 19]	m	$O\left(\frac{1}{\delta\varepsilon^2}\right)$	$\delta = \Omega(1)$
Extractors [40, 19, 30, 20]	$m + (1+\alpha)\cdot\log(1/\delta)$ any constant $\alpha > 0$	$\mathrm{poly}(\log(1/\delta), 1/\varepsilon)$	$\varepsilon, \delta = o(1)$
Expander Walk Chernoff [16]	$m + O(\log(1/\delta)/\varepsilon^2)$	$O\left(\frac{\log(1/\delta)}{\varepsilon^2}\right)$	$\varepsilon = \Omega(1)$
Pairwise Independence [12]	$O(m)$	$O\left(\frac{1}{\delta\varepsilon^2}\right)$	None, but simple
Non-Explicit [40]	$m + \log(1/\delta) - \log\log(1/\delta) + O(1)$	$O\left(\frac{\log(1/\delta)}{\varepsilon^2}\right)$	All
Lower Bound [11, 40, 27]	$m + \log(1/\delta) + \log(1/\varepsilon) - \log(D) - O(1)$	$\Omega\left(\frac{\log(1/\delta)}{\varepsilon^2}\right)$	N/A

a) Requires explicit constructions of Ramanujan graphs.

However, averaging samplers can also have uses beyond bounded functions: Błasiok [9], motivated by an application in streaming algorithms, introduced the notion of a *subgaussian sampler*, which he defined as an averaging sampler for functions $f : \{0,1\}^m \to \mathbb{R}$ such that $f(U_m)$ is a subgaussian random variable. Since subgaussian random variables have strong tail bounds, subgaussian functions from $\{0,1\}^m$ have a range contained in an interval of length $O(\sqrt{m})$, and thus one can construct a subgaussian sampler from a $[0,1]$-sampler by simply scaling the error ε by a factor of $O(\sqrt{m})$. Unfortunately, looking at Table 1 one sees that this

[1] A non-averaging sampler is an algorithm Samp which makes oracle queries to f and outputs an estimate of its average which is good with high probability, but need not simply output the average of f's values on the queried points.

induces a multiplicative dependence on m in the sample complexity, and for the expander walk sampler induces a dependence of $m \log(1/\delta)$ in the randomness complexity. This loss can be avoided for some samplers, such as the sampler of Chor and Goldreich [12] based on pairwise independence (as its analysis requires only bounded variance) and (as we will show) the Ramanujan Expander Neighbor sampler of [22, 19], but Błasiok showed [8] that the expander-walk sampler does not in general act as a subgaussian sampler without reducing the error to $o(1)$. We remark briefly that the median-of-averages sampler of Bellare, Goldreich, and Goldwasser [5] still works and is optimal up to constant factors in the subgaussian setting (since the underlying pairwise independent sampler works), but it is not an averaging sampler[1], and matching its parameters with an averaging sampler remains open in general even for $[0,1]$-valued functions.

One of the contributions of this work is to give explicit averaging samplers for subgaussian functions (in fact even for *subexponential* functions that satisfy weaker tail bounds) matching the extractor-based samplers for $[0,1]$-valued functions in Table 1 (up to the hidden polynomial in the sample complexity). This achieves the best parameters currently known in the regime of parameters where ε and δ are both subconstant, and in particular has no dependence on m in the sample complexity. We also show non-constructively that subexponentially samplers exist with essentially the same parameters as $[0,1]$-valued samplers.

▶ **Theorem 1.2** (Informal version of Theorem 6.1). *For every integer $m \in \mathbb{N}$ and $1 > \delta, \varepsilon > 0$, there is an explicit subgaussian (in fact subexponential) sampler with randomness complexity $n = m + O(\log(1/\delta))$ and sample complexity $D = \mathrm{poly}(\log(1/\delta), 1/\varepsilon)$.*

In the full version of this work [1], we show also that for every $m \in \mathbb{N}$, $1 > \delta, \varepsilon > 0$, and $\alpha > 0$, there is a function $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^D$ that is:

- *an explicit subexponential sampler with randomness complexity $n = m + (1+\alpha) \cdot \log(1/\delta)$ and sample complexity $D = \mathrm{poly}(\log(1/\delta), 1/\varepsilon)$.*
- *a non-constructive subexponential sampler with randomness complexity $n = m + \log(1/\delta) - \log\log(1/\delta) + O(1)$ and sample complexity $D = O(\log(1/\delta)/\varepsilon^2)$.*

1.2 Randomness extractors

To prove Theorem 1.2, we develop a corresponding theory of generalized *randomness extractors* which we believe is of independent interest. For bounded functions, Zuckerman [40] showed that averaging samplers are essentially equivalent to randomness extractors, and in fact several of the best-known constructions of such samplers arose as extractor constructions. Formally, a randomness extractor is defined as follows:

▶ **Definition 1.3** (Nisan and Zuckerman [26]). *A function $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is said to be a (k, ε)* extractor *if for every distribution X over $\{0,1\}^m$ satisfying $2^{-k} \geq \max_{x \in \{0,1\}^n} \Pr[X = x]$, the distributions $\mathrm{Ext}(X, U_d)$ and U_m are ε-close in total variation distance. Equivalently, for all $f : \{0,1\}^m \to [0,1]$ it holds that $\mathbb{E}[f(\mathrm{Ext}(X, U_d))] - \mathbb{E}[f(U_m)] \leq \varepsilon$. The number d is called the* seed length*, and m the* output length*.*

The formulation of Definition 1.3 in terms of $[0,1]$-valued functions implies that extractors produce an output distribution that is indistinguishable from uniform by all bounded functions f. It is therefore natural to consider a variant of this definition for a different set $\mathcal{F}$ of test functions $f : \{0,1\}^m \to \mathbb{R}$ which need not be bounded.

▶ **Definition 1.4** (Special case of Definition 3.1 using Definition 2.5). *A function $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is said to be a (k, ε) extractor for a set of real-valued functions $\mathcal{F}$ from $\{0,1\}^m$ if for every distribution X over $\{0,1\}^m$ satisfying $\max_{x \in \{0,1\}^n} \Pr[X = x] \leq 2^{-k}$ and every $f \in \mathcal{F}$, it holds that $\mathbb{E}[f(\mathrm{Ext}(X, U_d))] - \mathbb{E}[f(U_m)] \leq \varepsilon$.*

We show that much of the theory of extractors and samplers carries over to this more general setting. In particular, we generalize the connection of Zuckerman [40] to show that extractors for a class of functions of $\mathcal{F}$ are also samplers for that class, along with the converse (though as for total variation distance, there is some loss of parameters in this direction). Thus, to construct a subgaussian sampler it suffices (and is preferable) to construct a corresponding extractor for subgaussian test functions, which is how we prove Theorem 1.2.

Unfortunately, the distance induced by subgaussian test functions is not particularly pleasant to work with: for example the point masses on 0 and 1 in $\{0,1\}$ are $O(1)$ apart, but embedding them in the larger universe $\{0,1\}^m$ leads to distributions which are $\Theta(\sqrt{m})$ apart. We solve this problem by constructing extractors for a stronger notion, the *Kullback–Leibler (KL) divergence*, equivalently, extractors whose output is required to have very high Shannon entropy.

▶ **Definition 1.5** (Special case of Definition 3.1 using KL divergence)**.** *A function* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is said to be a* (k,ε) *KL-extractor if for every distribution* X *over* $\{0,1\}^m$ *satisfying* $\max_{x\in\{0,1\}^n} \Pr[X = x] \le 2^{-k}$ *it holds that* $\mathrm{KL}(\mathrm{Ext}(X, U_d) \parallel U_m) \le \varepsilon$*, or equivalently* $\mathrm{H}(\mathrm{Ext}(X, U_d)) \ge m - \varepsilon$*.*

A strong form of Pinsker's inequality (e.g. [10, Lemma 4.18]) implies that a (k,ε^2) KL-extractor is also a (k,ε) extractor for subgaussian test functions. The KL divergence has the advantage that is nonincreasing under the application of functions (the famous *data-processing inequality*), and although it does not satisfy a traditional triangle inequality, it does satisfy a similar inequality when one of the segments satisfies stronger ℓ_2 bounds. These properties allow us to show in the full version of this paper that the zig-zag product for extractors of Reingold, Wigderson, and Vadhan [30] also works for KL-extractors, and therefore to construct KL-extractors with seed length depending on n and k only through the *entropy deficiency* $n - k$ of X rather than n itself, which in the sampler perspective corresponds to a sampler with sample complexity depending on the failure probability δ rather than the universe size 2^m. Hence, we prove Theorem 1.2 by constructing corresponding KL-extractors.

▶ **Theorem 1.6** (Informal version of Theorem 6.5)**.** *For all integers* m *and* $1 > \delta, \varepsilon > 0$ *there is an explicit* (k,ε) *KL-extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with* $n = m + O(\log(1/\delta))$*,* $k = n - \log(1/\delta)$*, and* $d = O(\log\log(1/\delta) + \log(1/\varepsilon))$*.*

In the full version, we show that n *can be as small as* $m + (1+\alpha)\cdot\log(1/\delta)$ *for any constant* $\alpha > 0$*.*

Though the above theorem is most interesting in the high min-entropy regime where $n - k = o(n)$, we also show the existence of KL-extractors matching most of the existing constructions of total variation extractors. In particular, we note that extractors for ℓ_2 are immediately KL-extractors without loss of parameters, and also that any extractor can be made a KL-extractor by taking slightly smaller error, so that the extractors of Guruswami, Umans, and Vadhan [20] can be taken to be KL-extractors with essentially the same parameters.

Furthermore, in addition to our explicit constructions, we also show non-constructively that KL-extractors (and hence subgaussian extractors) exist with very good parameters:

▶ **Theorem 1.7** (Formal statement and proof in full version [1])**.** *For any integers* $k < n \in \mathbb{N}$ *and* $1 > \varepsilon > 0$ *there is a* (k,ε) *KL-extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with* $d = \log(n-k) + \log(1/\varepsilon) + O(1)$ *and* $m = k + d - \log(1/\varepsilon) - O(1)$*.*

One key thing to note about the nonconstructive KL extractors of the above theorem is that they incur an entropy loss of only $1 \cdot \log(1/\varepsilon)$, whereas total variation extractors necessarily incur entropy loss $2 \cdot \log(1/\varepsilon)$ by the lower bound of Radhakrishnan and Ta-Shma [27]. In particular, by Pinsker's inequality, (k, ε^2) KL-extractors with the above parameters are also optimal (k, ε) standard (total variation) extractors [27], so that one does not lose anything by constructing a KL-extractor rather than a total variation extractor. We also remark that the above theorem gives subgaussian samplers with better parameters than a naive argument that a random function should directly be a subgaussian sampler, as it avoids the need to take a union bound over $O(M^M) = O(2^{M \log M})$ test functions (for $M = 2^m$) which results in additional additive $\log \log$ factors in the randomness complexity.

In the total variation setting, there are only a couple of methods known to explicitly achieve optimal entropy loss $2 \cdot \log(1/\varepsilon)$, the easiest of which is to use an extractor which natively has this sort of loss, of which only three are known: An extractor from random walks over Ramanujan Graphs due to Goldreich and Wigderson [19], the Leftover Hash Lemma due to Impagliazzo, Levin, and Luby [21] (see also [23, 7]), and the extractor based on almost-universal hashing of Srinivasan and Zuckerman [33]. Unfortunately, all of these are ℓ_2 extractors and so must have seed length linear in $\min(n-k, m)$ (cf. [35, Problem 6.4]), rather than logarithmic in $n-k$ as known non-constructively. The other alternative is to use the generic reduction of Raz, Reingold, and Vadhan [28] which turns any extractor Ext with entropy loss Δ into one with entropy loss $2 \cdot \log(1/\varepsilon) + O(1)$ by paying an additive $O(\Delta + \log(n/\varepsilon))$ in seed length. We show in the full version of this paper that all of these ℓ_2 extractors and the [28] transformation also work to give KL-extractors with entropy loss $1 \cdot \log(1/\varepsilon) + O(1)$, so that applications which require minimal entropy loss can also use explicit constructions of KL-extractors.

1.3 Future directions

Broadly speaking, we hope that the perspective of KL-extractors will bring new tools (perhaps from information theory) to the construction of extractors and samplers. For example, since KL-extractors can have seed length with dependence on ε of only $1 \cdot \log(1/\varepsilon)$, trying to explicitly construct a KL-extractor with seed length $1 \cdot \log(1/\varepsilon) + o(\min(n-k, k))$ may also shed light on how to achieve optimal dependence on ε in the total variation setting.

In the regime of constant $\varepsilon = \Omega(1)$, we do not have explicit constructions of subgaussian samplers matching the expander-walk sampler of Gillman [16] for $[0,1]$-valued functions, which achieves randomness complexity $m + O(\log(1/\delta))$ and sample complexity $O(\log(1/\delta))$, as asked for by Błasiok [9]. From the extractor point-of-view, it would suffice (by the reduction of [19, 30] that we analyze for KL-extractors) to construct explicit *linear degree* KL-extractors with parameters matching the linear degree extractor of Zuckerman [41], i.e. with seed length $d = \log(n) + O(1)$ and $m = \Omega(k)$ for $\varepsilon = \Omega(1)$. A potentially easier problem, since the Zuckerman linear degree extractor is itself based on the expander-walk sampler, could be to instead match the parameters of the near-linear degree extractors of Ta-Shma, Zuckerman, and Safra [34] based on Reed–Muller codes, thereby achieving sample complexity $O(\log(1/\delta) \cdot \operatorname{poly} \log \log(1/\delta))$.

Finally, we hope that KL-extractors can also find uses beyond being subgaussian samplers and total variation extractors: for example it seems likely that there are applications (perhaps in coding or cryptography, cf. [4]) where it is more important to have high Shannon entropy in the output than small total variation distance to uniform, in which case one may be able to use (k, ε) KL-extractors with entropy loss only $1 \cdot \log(1/\varepsilon)$ directly, rather than a total variation extractor or (k, ε^2) KL-extractor with entropy loss $2 \cdot \log(1/\varepsilon)$.

2 Preliminaries

2.1 (Weak) statistical divergences and metrics

Our results in general will require very few assumptions on notions of "distance" between probability distributions, so we will give a general definition and indicate in our theorems when we need which assumptions.

▶ **Definition 2.1.** *A* weak statistical divergence *(or simply* weak divergence*) on a finite set $\mathcal{X}$ is a function* D *from pairs of probability distributions over X to $\mathbb{R} \cup \{\pm\infty\}$. We write* $\mathrm{D}(P \parallel Q)$ *for the value of* D *on distributions P and Q. Furthermore*

1. *If* $\mathrm{D}(P \parallel Q) \geq 0$ *with equality iff $P = Q$, then* D *is* positive-definite*, and we simply call* D *a* divergence*.*
2. *If* $\mathrm{D}(P \parallel Q) = \mathrm{D}(Q \parallel P)$*, then* D *is* symmetric*.*
3. *If* $\mathrm{D}(P \parallel R) \leq \mathrm{D}(P \parallel Q) + \mathrm{D}(Q \parallel R)$*, then* D *satisfies the* triangle inequality*.*
4. *If* $\mathrm{D}(\lambda P_1 + (1-\lambda)P_2 \parallel \lambda Q_1 + (1-\lambda)Q_2) \leq \lambda\,\mathrm{D}(P_1 \parallel Q_1) + (1-\lambda)\,\mathrm{D}(P_2 \parallel Q_2)$ *for all* $\lambda \in [0,1]$*, then* D *is* jointly convex*. If this holds only when $Q_1 = Q_2$ then* D *is* convex in its first argument*.*
5. *If* D *is defined on all finite sets $\mathcal{Y}$ and for all functions $f : \mathcal{X} \to \mathcal{Y}$ the divergence is nonincreasing under f, that is* $\mathrm{D}(f(P) \parallel f(Q)) \leq \mathrm{D}(P \parallel Q)$*, then* D *satisfies the* data-processing inequality*.*

If D *is positive-definite, symmetric, and satisfies the triangle inequality, then it is called a* metric*.*

▶ **Example 2.2.** The *ℓ_p distance* for $p > 0$ between probability distributions over $\mathcal{X}$ is

$$d_{\ell_p}(P,Q) \stackrel{\text{def}}{=} \left(\sum_{x\in\mathcal{X}} \left|P_x - Q_x\right|^p\right)^{1/p}$$

and is positive-definite and symmetric. Furthermore, for $p \geq 1$ it satisfies the triangle inequality (and so is a metric), and is jointly convex. The ℓ_p distance is nonincreasing in p.

▶ **Example 2.3.** The *total variation distance* is

$$d_{TV}(P,Q) \stackrel{\text{def}}{=} \frac{1}{2} d_{\ell_1}(P,Q) = \sup_{S\subseteq\mathcal{X}} \Big|\Pr[P \in S] - \Pr[Q \in S]\Big| = \sup_{f\in[0,1]^{\mathcal{X}}} \Big(\mathbb{E}[f(P)] - \mathbb{E}[f(Q)]\Big)$$

and is a jointly convex metric that satisfies the data-processing inequality.

▶ **Example 2.4** (Rényi Divergences [31])**.** For two probability distributions P and Q over a finite set $\mathcal{X}$, the *Rényi α-divergence* or *Rényi divergence of order α* is defined for real $0 < \alpha \neq 1$ by

$$\mathrm{D}_\alpha(P \parallel Q) \stackrel{\text{def}}{=} \frac{1}{\alpha - 1} \log\left(\sum_{x\in\mathcal{X}} \frac{P_x^\alpha}{Q_x^{\alpha-1}}\right)$$

where the logarithm is in base 2 (as are all logarithms in this paper unless noted otherwise). The Rényi divergence is continuous in α and so is defined by taking limits for $\alpha \in \{0, 1, \infty\}$, giving for $\alpha = 0$ the divergence $\mathrm{D}_0(P \parallel Q) \stackrel{\text{def}}{=} \log(1/\Pr_{x\sim Q}[P_x \neq 0])$, for $\alpha = 1$ the *Kullback–Leibler (or KL) divergence*

$$\mathrm{KL}(P \parallel Q) \stackrel{\text{def}}{=} \mathrm{D}_1(P \parallel Q) = \sum_{x\in X} P_x \log \frac{P_x}{Q_x},$$

and for $\alpha = \infty$ the *max-divergence* $\mathrm{D}_\infty(P \parallel Q) \stackrel{\text{def}}{=} \max_{x \in X} \log \frac{P_x}{Q_x}$. The Rényi divergence is nondecreasing in α. Furthermore, when $\alpha \leq 1$ the Rényi divergence is jointly convex, and for all α the Rényi divergence satisfies the data-processing inequality [37].

When $Q = U_\mathcal{X}$ is the uniform distribution over the set $\mathcal{X}$, then for all α, $\mathrm{D}_\alpha(P \parallel U_\mathcal{X}) = \log|\mathcal{X}| - \mathrm{H}_\alpha(P)$ where $0 \leq \mathrm{H}_\alpha(P) \leq \log|\mathcal{X}|$ is called the *Rényi α-entropy of P*. For $\alpha = 0$, $\mathrm{H}_0(P) = \log|\mathrm{Supp}(P)|$ is the *max-entropy of P*, for $\alpha = 1$, $\mathrm{H}_1(P) = \sum_{x \in \mathcal{X}} P_x \log(1/P_x)$ is the *Shannon entropy of P*, and for $\alpha = \infty$, $\mathrm{H}_\infty(P) = \min_{x \in \mathcal{X}} \log(1/P_x)$ is the *min-entropy of P*.

For $\alpha = 2$, the Rényi 2-entropy can be expressed in terms of the ℓ_2-distance to uniform:

$$\log|\mathcal{X}| - H_2(P) = \mathrm{D}_2(P \parallel U_\mathcal{X}) = \log\bigl(1 + |\mathcal{X}| \cdot d_{\ell_2}(P, U_\mathcal{X})^2\bigr)$$

2.2 Statistical weak divergences from test functions

Zuckerman's connection [40] between samplers for bounded functions and extractors for total variation distance is based on the following standard characterization of total variation distance as the maximum distinguishing advantage achieved by bounded functions,

$$d_{TV}(P, Q) = \sup_{f \in [0,1]^\mathcal{X}} \mathbb{E}[f(P)] - \mathbb{E}[f(Q)].$$

By considering an arbitrary class of functions in the supremum, we get the following weak divergence:

▶ **Definition 2.5.** *Given a finite $\mathcal{X}$ and a set of real-valued functions $\mathcal{F} \subseteq \mathbb{R}^\mathcal{X}$, the* $\mathcal{F}$-distance on $\mathcal{X}$ *between probability measures on $\mathcal{X}$ is denoted by $\mathrm{D}^\mathcal{F}$ and is defined as*

$$\mathrm{D}^\mathcal{F}(P \parallel Q) \stackrel{\text{def}}{=} \sup_{f \in \mathcal{F}} \Bigl(\mathbb{E}[f(P)] - \mathbb{E}[f(Q)]\Bigr) = \sup_{f \in \mathcal{F}} \mathrm{D}^{\{f\}}(P \parallel Q),$$

where we use a superscript to avoid confusion with the Csiszár-Morimoto-Ali-Silvey f-divergences [13, 24, 2].

We call the set of functions $\mathcal{F}$ symmetric *if for all $f \in \mathcal{F}$ there is $c \in \mathbb{R}$ and $g \in \mathcal{F}$ such that $g = c - f$, and* distinguishing *if for all $P \neq Q$ there exists $f \in \mathcal{F}$ with $\mathrm{D}^{\{f\}}(P \parallel Q) > 0$.*

▶ **Example 2.6.** If $\mathcal{F} = \{0,1\}^\mathcal{X}$ or $\mathcal{F} = [0,1]^\mathcal{X}$, then $\mathrm{D}^\mathcal{F}$ is exactly the total variation distance.

▶ Remark 2.7. An equivalent definition of $\mathcal{F}$ being symmetric is that for all $f \in \mathcal{F}$ there exists $g \in \mathcal{F}$ with $\mathrm{D}^{\{g\}}(P \parallel Q) = -D^{\{f\}}(P \parallel Q) = \mathrm{D}^{\{f\}}(Q \parallel P)$ for all distributions P and Q. Hence, one might also consider a weaker notion of symmetry that reverses quantifiers, where $\mathcal{F}$ is "weakly-symmetric" if for all $f \in \mathcal{F}$ and distributions P and Q there exists $g \in \mathcal{F}$ such that $\mathrm{D}^{\{g\}}(P \parallel Q) = -\mathrm{D}^{\{f\}}(P \parallel Q) = \mathrm{D}^{\{f\}}(Q \parallel P)$. However, such a class $\mathcal{F}$ gives exactly the same weak divergence $\mathrm{D}^\mathcal{F}$ as its "symmetrization" $\overline{\mathcal{F}} = \mathcal{F} \cup \{-f \mid f \in \mathcal{F}\}$, so we do not need to introduce this more complex notion.

▶ Remark 2.8. By identifying distributions with their probability mass function, one can realize $\mathbb{E}[f(P)] - \mathbb{E}[f(Q)]$ as an inner product $\langle P - Q, f\rangle$. Definition 2.5 can thus be written as $\mathrm{D}^\mathcal{F}(P \parallel Q) = \sup_{f \in \mathcal{F}} \langle P - Q, f\rangle$, which is essentially the notion of indistinguishability considered in several prior works, (see e.g. the survey of Reingold, Trevisan, Tulsiani, and Vadhan [29]), but without requiring all f to be bounded.

▶ Remark 2.9. For simplicity, all our probabilistic distributions are given only for random variables and distributions over finite sets as this is all we need for our application. A more general version of Definition 2.5 has been studied by e.g. Zolotarev [39] and Müller [25] and is commonly used in developments of Stein's method in probability.

We now note some basic properties of $\mathrm{D}^{\mathcal{F}}$.

▶ **Lemma 2.10.** *Let $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$ be a set of real-valued functions over a finite set $\mathcal{X}$. Then $\mathrm{D}^{\mathcal{F}}$ satisfies the triangle inequality and is jointly convex, and*

1. *if $\mathcal{F}$ is symmetric then $\mathrm{D}^{\mathcal{F}}$ is symmetric and*

$$\mathrm{D}^{\mathcal{F}}(P \parallel Q) = \sup_{f \in \mathcal{F}} \Big| \mathbb{E}[f(P)] - \mathbb{E}[f(Q)] \Big| \geq 0,$$

2. *if $\mathcal{F}$ is distinguishing then $\mathrm{D}^{\mathcal{F}}$ is positive-definite,*

so that if $\mathcal{F}$ is both symmetric and distinguishing then $\mathrm{D}^{\mathcal{F}}$ is a jointly convex metric on probability distributions over $\mathcal{X}$, in which case we also use the notation $d_{\mathcal{F}}(P, Q) \stackrel{\text{def}}{=} \mathrm{D}^{\mathcal{F}}(P \parallel Q)$.

Furthermore, the notion of dual norm has an appealing interpretation in this framework via Remark 2.8, generalizing the fact that total variation distance corresponds to $[0, 1]$-valued test functions (or equivalently that ℓ_1 distance corresponds to to $[-1, 1]$-valued functions).

▶ **Proposition 2.11.** *Let $1 \leq p, q \leq \infty$ be Hölder conjugates (meaning $1/p + 1/q = 1$), and let*

$$\mathcal{M}_q \stackrel{\text{def}}{=} \left\{ f : \{0,1\}^m \to \mathbb{R} \;\middle|\; \|f(U_m)\|_q \stackrel{\text{def}}{=} \mathbb{E}[|f(U_m)|^q]^{1/q} \leq 1 \right\}$$

be the set of real-valued functions from $\{0,1\}^m$ with bounded q-th moments. Then $d_{\ell_p} = 2^{-m/q} \cdot d_{\mathcal{M}_q}$, in the sense that for all probability distributions A and B over $\{0,1\}^m$ it holds that $d_{\ell_p}(A, B) = 2^{-m/q} \cdot d_{\mathcal{M}_q}(A, B)$. In particular, taking $p = 1$ and $q = \infty$ recovers the result for ℓ_1 (equivalently total variation) distance.

Proof Sketch. As mentioned this is just the standard fact that the ℓ_p and ℓ_q norms are dual, but for completeness we include a proof in Appendix A. ◀

3 Extractors for weak divergences and connections to samplers

3.1 Definitions

We now use this machinery to extend the notion of an extractor due to Nisan and Zuckerman [26] and the average-case variant of Dodis, Ostrovsky, Reyzin, and Smith [14].

▶ **Definition 3.1** (Extends Definition 1.4)**.** *Let* D *be a weak divergence on the set $\{0,1\}^m$, and* Ext $: \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$*. Then if for all distributions X over $\{0,1\}^n$ with $\mathrm{H}_\infty(X) \geq k$ it holds that*

1. $\mathrm{D}(\mathrm{Ext}(X, U_d) \parallel U_m) \leq \varepsilon$*, then* Ext *is said to be a (k, ε)* extractor for D*, or a (k, ε)* D-extractor*.*
2. $\mathbb{E}_{s \sim U_d}[\mathrm{D}(\mathrm{Ext}(X, s) \parallel U_m)] \leq \varepsilon$*, then* Ext *is said to be a (k, ε)* strong extractor for D*, or a (k, ε)* strong D-extractor*.*

Furthermore, if for all joint distributions (Z, X) where X is distributed over $\{0,1\}^n$ with $\tilde{\mathrm{H}}_\infty(X|Z) \stackrel{\text{def}}{=} \log\big(1/\mathbb{E}_{z \sim Z}\big[2^{-\mathrm{H}_\infty(X|_{Z=z})}\big]\big) \geq k$, it holds that

3. $\mathbb{E}_{z \sim Z}[\mathrm{D}(\mathrm{Ext}(X|_{Z=z}, U_d) \parallel U_m) \leq \varepsilon]$*, then* Ext *is said to be a (k, ε)* average-case extractor for D*, or a (k, ε)* average-case D-extractor*.*
4. $\mathbb{E}_{z \sim Z, s \sim U_d}[\mathrm{D}(\mathrm{Ext}(X|_{Z=z}, s) \parallel U_m)] \leq \varepsilon$*, then* Ext *is said to be a (k, ε)* average-case strong extractor for D*, or a (k, ε)* average-case strong D-extractor*.*

▶ Remark 3.2. By taking D to be the total variation distance we recover the standard definitions of extractor and strong extractor due to [26] and the definition of average-case extractor due to [14].

However, our definitions are phrased slightly differently for strong and average-case extractors as an expectation rather than a joint distance, that is, for strong average-case extractors we require a bound on the expectation $\mathbb{E}_{z\sim Z, s\sim U_d}[\mathrm{D}(\mathrm{Ext}(X|_{Z=z}, s) \parallel U_m)]$ rather than a bound on $\mathrm{D}(Z, U_d, \mathrm{Ext}(X, U_d) \parallel Z, U_d, U_m)$. In our setting, the weak divergence D need not be defined over the larger joint universe, but it is defined for all random variables over $\{0,1\}^m$. In the case of d_{TV} and KL divergence, both definitions are equivalent (for KL divergence, this is an instance of the *chain rule*).

In the full version of this work [1] we include more discussion about this definition, and also generalize a result of Vadhan [35, Problem 6.8] showing that all $\mathrm{D}^{\mathcal{F}}$-extractors are average-case with only a constant factor loss in the error parameter.

We also give the natural definition of averaging samplers for arbitrary classes of functions $\mathcal{F}$ extending Definition 1.1, along with the strong variant of Zuckerman [40].

▶ **Definition 3.3.** *Given a class of functions* $\mathcal{F} : \{0,1\}^m \to \mathbb{R}$*, a function* $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^D$ *is said to be a* (δ, ε) strong averaging sampler for $\mathcal{F}$ *or a* (δ, ε) strong averaging $\mathcal{F}$-sampler *if for all* $f \in \mathcal{F}$*, it holds that*

$$\Pr_{x\sim U_n}\left[\mathop{\mathbb{E}}_{i\sim U_{[D]}}\left[f_i(\mathrm{Samp}(x)_i) - \mathbb{E}[f_i(U_m)]\right] > \varepsilon\right] \leq \delta$$

where $[D] = \{1, \ldots, D\}$*. If this holds only when* $f_1 = \cdots = f_D$*, then it is called a* (non-strong) (δ, ε) averaging sampler for $\mathcal{F}$ *or* (δ, ε) averaging $\mathcal{F}$-sampler*. We say that* Samp *is a* (δ, ε) strong absolute averaging sampler for $\mathcal{F}$ *if it also holds that*

$$\Pr_{x\sim U_n}\left[\left|\mathop{\mathbb{E}}_{i\sim U_{[D]}}\left[f_i(\mathrm{Samp}(x)_i) - \mathbb{E}[f_i(U_m)]\right]\right| > \varepsilon\right] \leq \delta.$$

with the analogous definition for non-strong samplers.

▶ Remark 3.4. We separated a single-sided version of the error bound in Definition 3.3 as in [35], as it makes the connection between extractors and samplers cleaner and allows us to be specific about what assumptions are needed. Note that if $\mathcal{F}$ is symmetric then every (δ, ε) (strong) sampler for $\mathcal{F}$ is a $(2\delta, \varepsilon)$ (strong) absolute sampler for $\mathcal{F}$, recovering the standard notion up to a factor of 2 in δ.

3.2 Equivalence of extractors and samplers

We now show that Zuckerman's connection [40] does indeed generalize to this broader setting as promised.

▶ **Theorem 3.5.** *Let* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be an* $(n - \log(1/\delta), \varepsilon)$*-extractor (respectively strong extractor) for the weak divergence* $\mathrm{D}^{\mathcal{F}}$ *defined by a class of test functions* $\mathcal{F} : \{0,1\}^m \to \mathbb{R}$ *as in Definition 2.5. Then the function* $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^D$ *for* $D = 2^d$ *defined by* $\mathrm{Samp}(x)_i = \mathrm{Ext}(x, i)$ *is a* (δ, ε)*-sampler (respectively strong sampler) for* $\mathcal{F}$*.*

Proof sketch. The proof is given in Appendix A and is similar to that of Zuckerman [40]. The key idea is that for any function $f \in \mathcal{F}$, the set of seeds B_f which are bad for Samp with respect to f must be small, as otherwise $\mathbb{E}\left[f(\mathrm{Ext}(U_{B_f}, U_d))\right] - \mathbb{E}[f(U_m)] > \varepsilon$ contradicting the extractor property, where U_{B_f} is uniform over the set B_f. ◀

▶ Remark 3.6. Hölder's inequality implies that an extractor for ℓ_p with error $\varepsilon \cdot 2^{-m(p-1)/p}$ is also an ℓ_1 extractor and thus $[-1,1]$-averaging sampler with error ε. Proposition 2.11 and Theorem 3.5 show that they are in fact samplers for the much larger class of functions $\mathcal{M}_{p/(p-1)}$ with bounded $p/(p-1)$ moments (rather than just ∞ moments), also with error ε.

Furthermore, if all the functions in $\mathcal{F}$ have bounded deviation from their mean (for example, subgaussian functions from $f : \{0,1\}^m \to \mathbb{R}$ have such a bound of $O(\sqrt{m})$ by the tail bounds from Lemma 4.3), then we also have a partial converse that recovers the standard converse in the case of total variation distance.

▶ **Theorem 3.7.** *Let $\mathcal{F}$ be a class of functions $\mathcal{F} \subset \{0,1\}^m \to \mathbb{R}$ with finite* maximum deviation from the mean*, meaning* $\max \mathrm{dev}(\mathcal{F}) \stackrel{\text{def}}{=} \sup_{f\in\mathcal{F}} \max_{x\in\{0,1\}^n} \big(f(x) - \mathbb{E}[f(U_m)]\big) < \infty$*. Then given a (δ,ε) $\mathcal{F}$-sampler (respectively (δ,ε) strong $\mathcal{F}$-sampler)* $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^D$*, the function* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *for $d = \log D$ defined by* $\mathrm{Ext}(x,i) = \mathrm{Samp}(x)_i$ *is a $\big(k, \varepsilon + \delta \cdot 2^{n-k} \cdot \max \mathrm{dev}(\mathcal{F})\big)$ $\mathrm{D}^{\mathcal{F}}$-extractor (respectively strong $\mathrm{D}^{\mathcal{F}}$-extractor) for every $0 \le k \le n$.*

In particular, Ext *is an $\big(n - \log(1/\delta) + \log(1/\eta), \varepsilon + \eta \cdot \max \mathrm{dev}(\mathcal{F})\big)$ average-case $\mathrm{D}^{\mathcal{F}}$-extractor (respectively strong average-case $\mathrm{D}^{\mathcal{F}}$-extractor) for every $\delta \le \eta \le 1$.*

Proof sketch. The proof is given in Appendix A and is again similar to that of Zuckerman [40]. The key idea is that for any function $f \in \mathcal{F}$, since most $x \in \{0,1\}^n$ are good for Samp, for any source X of sufficient min-entropy, the probability over x from X that $\mathbb{E}[f(\mathrm{Ext}(x,U_d))] - \mathbb{E}[f(U_m)] > \varepsilon$ must be at most η, and in this failure case we can fall back on the trivial bound of $\max \mathrm{dev}(\mathcal{F})$. ◀

4 Subgaussian distance and connections to other notions

Now that we've introduced the general machinery we need, we can go back to our motivation of subgaussian samplers. We will need some standard facts about subgaussian and subexponential random variables, we recommend the book of Vershynin [38] for an introduction.

▶ **Definition 4.1.** *A real-valued mean-zero random variable Z is said to be* subgaussian with parameter σ *if for every $t \in \mathbb{R}$ the moment generating function of Z is bounded as $\ln \mathbb{E}\big[e^{tZ}\big] \le \frac{t^2\sigma^2}{2}$. If this is only holds for $|t| \le b$ then Z is said to be (σ,b)-*subgamma*, and if Z is $(\sigma, 1/\sigma)$-subgamma then Z is said to be* subexponential with parameter σ.

▶ Remark 4.2. There are many definitions of subgaussian (and especially subexponential) random variables in the literature, but they are all equivalent up to constant factors in σ and only affect constants already hidden in big-O's.

▶ **Lemma 4.3.** *Let Z be a real-valued random variable. Then*

1. *(Hoeffding's lemma) If Z is bounded in the interval $[0,1]$, then $Z - \mathbb{E}[Z]$ is subgaussian with parameter $1/2$.*
2. *If Z is mean-zero, then Z is subgaussian (respectively subexponential) with parameter σ if and only if cZ is subgaussian (respectively subexponential) with parameter $|c|\sigma$ for every $c \neq 0$.*

Furthermore, if Z is mean-zero and subgaussian with parameter σ, then

1. *For all $t > 0$, $\max\big(\Pr[Z > t], \Pr[Z < -t]\big) \le e^{-t^2/2\sigma^2}$.*
2. *$\|Z\|_p \stackrel{\text{def}}{=} \mathbb{E}[|Z|^p]^{1/p} \le 2\sigma\sqrt{p}$ for all $p \ge 1$.*
3. *Z is subexponential with parameter σ.*

We are now in a position to formally define the *subgaussian distance*.

▶ **Definition 4.4.** *For every finite set $\mathcal{X}$, we define the set $\mathcal{G}_\mathcal{X}$ of* subgaussian test functions on $\mathcal{X}$ *(respectively the set $\mathcal{E}_\mathcal{X}$ of* subexponential test functions on $\mathcal{X}$*) to be the set of functions $f : \mathcal{X} \to \mathbb{R}$ such that the random variable $f(U_\mathcal{X})$ is mean-zero and subgaussian (respectively subexponential) with parameter $1/2$. Then $\mathcal{G}_\mathcal{X}$ and $\mathcal{E}_\mathcal{X}$ are symmetric and distinguishing, so by Lemma 2.10 the respective distances induced by $\mathcal{G}_\mathcal{X}$ and $\mathcal{E}_\mathcal{X}$ are jointly convex metrics called the* subgaussian distance *and* subexponential distance *respectively and are denoted as $d_\mathcal{G}(P, Q)$ and $d_\mathcal{E}(P, Q)$.*

▶ Remark 4.5. We choose subgaussian parameter $1/2$ in Definition 4.4 as by Hoeffding's lemma, all functions $f : \{0,1\}^m \to [0,1]$ have that $f(U_m) - \mathbb{E}[f(U_m)]$ is subgaussian with parameter $1/2$, so this choice preserves the same "scale" as total variation distance. However, the choice of parameter is essentially irrelevant by linearity, as different choices of parameter simply scale the metric $d_\mathcal{G}$.

Note that absolute averaging samplers for $\mathcal{G}_{\{0,1\}^m}$ from Definition 3.3 are exactly subgaussian samplers as defined in the introduction. Thus, by Remark 3.4 and Theorem 3.5, to construct subgaussian samplers it is enough to construct extractors for the subgaussian distance $d_\mathcal{G}$.

4.1 Composition

Unfortunately, the subgaussian distance has a major disadvantage compared to total variation distance that complicates extractor construction: it does not satisfy the data-processing inequality, that is, there are probability distributions P and Q over a set A and a function $f : A \to B$ such that

$$d_\mathcal{G}(f(P), f(Q)) \not\leq d_\mathcal{G}(P, Q).$$

This happens because subgaussian distance is defined by functions which are required to be subgaussian only with respect to the *uniform distribution*. A simple explicit counterexample comes from taking $f : \{0,1\}^1 \to \{0,1\}^m$ defined by $x \mapsto (x, 0^{m-1})$ and taking P to be the point mass on 0 and Q the point mass on 1. Their subgaussian distance in $\{0,1\}^1$ is obviously $O(1)$, but the subgaussian distance of $f(P)$ and $f(Q)$ in $\{0,1\}^m$ is $\Theta(\sqrt{m})$.

The reason this matters because a standard operation (cf. Nisan and Zuckerman [26]; Goldreich and Wigderson [19]; Reingold, Vadhan, and Wigderson [30]) in the construction of samplers and extractors for bounded functions is to do the following: given extractors

$$\mathrm{Ext}_{out} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m \qquad \mathrm{Ext}_{in} : \{0,1\}^{n'} \times \{0,1\}^{d'} \to \{0,1\}^d,$$

define $\mathrm{Ext} : \{0,1\}^{n+n'} \times \{0,1\}^{d'} \to \{0,1\}^m$ by

$$\mathrm{Ext}\big((x,y),s\big) = \mathrm{Ext}_{out}\big(x, \mathrm{Ext}_{in}(y,s)\big).$$

The reason this works for total variation distance is exactly the data-processing inequality: if Y has enough min-entropy given X, then $\mathrm{Ext}_{in}(Y, U_{d'})$ will be close in total variation distance to U_d, and by the data-processing inequality for total variation distance this closeness is not lost under the application of Ext_{out}. The assumption that Y has min-entropy given X means that (X, Y) is a so-called *block-source*, and is implied by (X, Y) having enough min-entropy as a joint distribution. From the sampler perspective, this construction uses the inner sampler Ext_{in} to subsample the outer sampler. On the other hand, for subgaussian distance, the

distribution $\mathrm{Ext}_{in}(Y, U_{d'})$ can be ε-close to uniform but still have some element with excess probability mass $\Omega(\varepsilon/\sqrt{d})$, and this element (seed) when mapped by Ext_{out} can retain[2] this excess mass in $\{0,1\}^m$, which results in subgaussian distance $\Theta(\varepsilon\sqrt{m/d}) \gg \varepsilon$. Similarly, from the sampler perspective, even when the outer sampler Ext_{out} is a good subgaussian sampler for $\{0,1\}^m$, there is no reason that a good subgaussian sampler Ext_{in} for $\{0,1\}^d$ the seeds of Ext_{out} will preserve the larger sampler property when $m \gg d$.

Thus, since this composition operation is needed to construct high-min entropy extractors with the desired seed length even for total variation distance, to construct such extractors for subgaussian distance we need to bypass this barrier. The natural approach is to construct extractors for a better-behaved weak divergence that bounds the subgaussian distance.

4.2 Connections to other weak divergences

Therefore, to aid in extractor construction, we show how $d_{\mathcal{G}}$ relates to other statistical weak divergences (though for space reasons, we defer all proofs to Appendix A).

Most basically, the subgaussian distance over $\{0,1\}^m$ differs from total variation distance up to a factor of $O(\sqrt{m})$.

▶ **Lemma 4.6.** *Let P and Q be distributions on $\{0,1\}^m$. Then*

$$d_{TV}(P,Q) \le d_{\mathcal{G}}(P,Q) \le \sqrt{2\ln 2 \cdot m} \cdot d_{TV}(P,Q)$$

While this allows constructing subgaussian extractors and samplers from total variation extractors, as discussed in the introduction the fact that the upper bound depends on m leads to suboptimal bounds. By starting with a stronger measure of error, we pay a much smaller penalty.

▶ **Lemma 4.7.** *Let P and Q be distributions on $\{0,1\}^m$. Then for every $\alpha > 0$*

$$2d_{TV}(P,Q) = d_{\ell_1}(P,Q) \le 2^{m\alpha/(1+\alpha)} \cdot d_{\ell_{1+\alpha}}(P,Q)$$
$$d_{\mathcal{G}}(P,Q) \le 2^{m\alpha/(1+\alpha)} \sqrt{1 + \frac{1}{\alpha}} \cdot d_{\ell_{1+\alpha}}(P,Q)$$

In particular, that there is only an additional $\sqrt{1+1/\alpha}$ factor when moving to subgaussian distance compared to total variation, which in particular does not depend on m and is constant for constant α. We give the proof in Appendix A.

One downside of starting with bounds on $\ell_{1+\alpha}$ is that, extending a well-known linear seed length linear bound for ℓ_2-extractors (e.g. [35, Problem 6.4]), we show in the full version of this work [1] that for every $1 > \alpha > 0$, there is a constant $c_\alpha > 0$ such any $\ell_{1+\alpha}$ extractor with error smaller than $c_\alpha \cdot 2^{-m\alpha/(1+\alpha)}$ requires seed length linear in $\alpha \cdot \min(n-k, m)$, for $n-k$ the entropy deficiency and m the output length. One might hope that sending α to 0 would eliminate this linear lower bound but still bound the subgaussian distance, but phrased this way sending α to 0 just results in a total variation extractor.

However, with a shift in perspective essentially the same approach works: by Example 2.4, $d_{\ell_2}(P, U_m) \le \varepsilon \cdot 2^{-m/2}$ implies $\mathrm{D}_2(P \parallel U_m) \le \varepsilon^2/\ln 2$, and there is an analogous linear seed length lower bound on constant error $\mathrm{D}_{1+\alpha}$ extractors for every $\alpha > 0$. In this case, however, sending α to 0 results in the *KL divergence*, which does upper bound the subgaussian distance, and in fact with the same parameters as for total variation distance.

[2] Given a subgaussian extractor Ext with $d \ge \log(m/\varepsilon)$, adding a single extra seed $*$ to Ext such that $\mathrm{Ext}(x,*) = 0^m$ results in a subgaussian extractor with error at most $2^{-d} \cdot \sqrt{2m} + \varepsilon \le 3\varepsilon$ by convexity of $d_{\mathcal{G}}$ and the fact that $\left\| d_{\mathcal{G}_{\{0,1\}^m}} \right\|_\infty < \sqrt{2m}$.

▶ **Lemma 4.8** (cf. [10, Lemma 4.15], [18, Fact B.1]). *Let P be a distribution on $\{0,1\}^m$. Then*

$$d_{\mathcal{G}}(P, U_m) \leq \sqrt{\frac{\ln 2}{2} \cdot \mathrm{KL}(P \parallel U_m)}$$

$$d_{\mathcal{E}}(P, U_m) \leq \begin{cases} \sqrt{\frac{\ln 2}{2} \cdot \mathrm{KL}(P \parallel U_m)} & \text{if } \mathrm{KL}(P \parallel U_m) \leq \frac{1}{2 \ln 2} \\ \frac{\ln 2}{2} \cdot \mathrm{KL}(P \parallel U_m) + \frac{1}{4} & \text{if } \mathrm{KL}(P \parallel U_m) > \frac{1}{2 \ln 2} \end{cases}$$

where these bounds are concave in $\mathrm{KL}(P \parallel U_m)$. In the reverse direction, it holds that

$$\mathrm{KL}(P \parallel U_m) \leq m \cdot d_{TV}(P, U_m) + h(d_{TV}(P, U_m))$$

where $h(x) = x \log(1/x) + (1-x)\log(1/(1-x))$ is the (concave) binary entropy function.

Due to space constraints, we defer the proof to Appendix A.

5 Extractors for KL divergence

Since by Lemma 4.8 the subgaussian distance can be bounded in terms of the KL divergence to uniform, the following easy lemma shows that to construct subgaussian extractors it suffices to construct extractors for KL divergence.

▶ **Lemma 5.1.** *Let V_1 and V_2 be weak divergences on the set $\{0,1\}^m$ and $f : \mathbb{R} \to \mathbb{R}$ be a function such that $V_1(P \parallel U_M) \leq f(V_2(P \parallel U_m))$ for all distributions P on $\{0,1\}^m$. Then if f is increasing on $(0, \varepsilon)$, every (k, ε) extractor Ext for V_1 is also a $(k, f(\varepsilon))$-extractor for V_2, and if f is also concave, then if Ext is strong or average-case as a V_1-extractor, it has the same properties as a $(k, f(\varepsilon))$ extractor for V_2.*

Importantly, the KL divergence does not have the flaws of subgaussian distance discussed in Section 4.1. For instance, the classic *data-processing inequality* says that KL divergence is non-increasing under postprocessing by (possibly randomized) functions, and the *chain rule* for KL divergence says that

$$\mathrm{KL}(A, B \parallel X, Y) = \mathrm{KL}(A \parallel X) + \mathop{\mathbb{E}}_{a \sim A}[\mathrm{KL}(B|_{A=a} \parallel Y|_{X=a})]$$

for all distributions A, B, X, and Y, which implies for example that

$$\mathop{\mathbb{E}}_{s \sim U_d}[\mathrm{KL}(\mathrm{Ext}(X, s) \parallel U_m)] = \mathrm{KL}(U_d, \mathrm{Ext}(X, U_d) \parallel U_d, U_m).$$

Furthermore, KL divergence satisfies a type of triangle inequality when combined with higher Rényi divergences:

▶ **Lemma 5.2** (cf. [36, Lemma 6.6]). *Let P, Q, and R be distributions over a finite set $\mathcal{X}$. Then for all $\alpha > 0$, it holds that*

$$\mathrm{KL}(P \parallel R) \leq \left(1 + \frac{1}{\alpha}\right) \cdot \mathrm{KL}(P \parallel Q) + \mathrm{D}_{1+\alpha}(Q \parallel R)$$

We give the proof in Appendix A.

5.1 Composition

These properties imply that composition does work as we want (without any loss depending on the output length m) assuming we have extractors for KL and higher divergences.

▶ **Theorem 5.3** (Composition for high min-entropy Rényi entropy extractors, cf. [19]). *Suppose*
1. $\mathrm{Ext}_{out} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is an* $(n - \log(1/\delta), \varepsilon_{out})$ *extractor for* $\mathrm{D}_{1+\alpha}$ *with* $\alpha > 0$,
2. $\mathrm{Ext}_{in} : \{0,1\}^{n'} \times \{0,1\}^{d'} \to \{0,1\}^d$ *is an* $(n' - \log(1/\delta), \varepsilon_{in})$ *average-case KL-extractor, and define* $\mathrm{Ext} : \{0,1\}^{n+n'} \times \{0,1\}^{d'} \to \{0,1\}^m$ *by* $\mathrm{Ext}\big((x,y),s\big) = \mathrm{Ext}_{out}(x, \mathrm{Ext}_{in}(y,s))$.

Then Ext *is an* $(n + n' - \log(1/\delta), \varepsilon_{out} + (1 + 1/\alpha) \cdot \varepsilon_{in})$ *extractor for KL.*

We prove this in Appendix A.

5.2 Further theory

The reader is advised to consult the full version of this paper [1] for a more thorough development of the theory of KL-extractors, including an extension of the zig-zag product for extractors (Reingold, Vadhan, and Wigderson [30]), which allows us to avoid the $\log(1/\delta)$ entropy loss inherent in Theorem 5.3. We also give lower bounds, an optimal non-explicit construction, and interpretations of several existing extractor constructions as KL-extractors.

6 Constructions of subgaussian samplers

We can now establish a weak version of our explicit construction of subgaussian samplers with sample complexity having no dependence on m and sample complexity matching the best-known $[0,1]$-valued sampler when ε and δ are subconstant (up to the hidden polynomial in the sample complexity). Obtaining matching randomness complexity as well requires more technology from KL-extractors to develop, and as such we defer the proof to the full version of this paper [1].

▶ **Theorem 6.1.** *For all* $m \in \mathbb{N}$, $1 > \varepsilon, \delta > 0$, *and* $\alpha > 0$ *there is an explicit* (δ, ε) *absolute averaging sampler for subgaussian and subexponential functions* $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^D$ *with sample complexity* $D = \mathrm{poly}(\log(1/\delta), 1/\varepsilon)$ *and randomness complexity* $n = m + O(\log(1/\delta))$.

▶ Remark 6.2. In the full version of this paper, we show for every constant $\alpha > 0$ the existence of an explicit absolute subexponential sampler with the same sample complexity $D = \mathrm{poly}(\log(1/\delta), 1/\varepsilon)$ and randomness complexity $n = m + (1+\alpha)\log(1/\delta)$, and also an analogous result for strong subexponential samplers.

We will use essentially the same construction used for bounded samplers in this regime, combining the expander extractor of Goldreich and Wigderson [19] and an extractor with logarithmic seed length. However, as described in Section 4.1, this construction does not work for general subgaussian extractors, so we will instead use the analysis of Theorem 5.3. This requires a $\mathrm{D}_{1+\alpha}$-extractor for $\alpha > 0$, for this we note (following [35]) that the extractor of [19] is already an extractor for D_2 (see the full version of this work [1] for more details).

▶ **Theorem 6.3** ([19] [35, Discussion after Theorem 6.22]). *For all* $k \le n \in \mathbb{N}$ *and* $1/2 \ge \varepsilon > 0$ *there is an explicit* (k, ε) D_2*-extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with seed length* $d = O(n - k + \log(1/\varepsilon))$ *and output length* $m = n$.

We also need an average-case KL-extractor, which we can construct by reducing the error in the extractors of Guruswami–Umans–Vadhan [20]:

▶ **Theorem 6.4** (Akin to [20, Theorem 1.5]). *For every $\alpha, \varepsilon > 0$ and integers $k \leq n$, there is an explicit average-case (k, ε)-KL-extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with $d \leq \log n + O_\alpha(\log(k/\varepsilon))$ and $m \geq (1-\alpha)k$.*

Though Theorem 6.4 has seed length depending on n the input length, this is tolerable for us since we will apply it to Ext_{in} in the composition of Theorem 5.3 with $n = O(\log(1/\delta) + \log(1/\varepsilon))$:

Proof. Let $\varepsilon' = \frac{\min(\varepsilon, 1/2)}{48(m+\log(1/\varepsilon))}$ so that $m \cdot 3\varepsilon' + h(3\varepsilon') \leq \varepsilon$, where $h(x) = x\log(1/x) + (1-x)\log(1/(1-x))$ is the binary entropy function. By [20, Theorem 1.5] and [35, Problem 6.8] there is an explicit $(k, 3\varepsilon')$ extractor $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d \leq \log n + O_\alpha(\log(k/\varepsilon')) = \log n + O_\alpha(\log(k/\varepsilon))$ and $m \geq (1-\alpha)k$. By Lemmas 4.8 and 5.1, we also have that Ext is a $(k, m \cdot 3\varepsilon' + h(3\varepsilon'))$ average-case KL-extractor, and thus a (k, ε) average-case KL-extractor as desired. ◀

▶ **Theorem 6.5.** *For all integers m and $\delta, \varepsilon > 0$ there is an explicit (k, ε)-KL-extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with $n = m + O(\log(1/\delta))$, $k = n - \log(1/\delta)$, and $d = O(\log\log(1/\delta) + \log(1/\varepsilon))$.*

Proof. Let $\mathrm{Ext}_{out} : \{0,1\}^m \times \{0,1\}^{d_{out}} \to \{0,1\}^m$ the $(m - \log(1/\delta), \varepsilon/3)$ be the D_2-extractor from Theorem 6.3 with $d_{out} = O(\log(1/\delta) + \log(1/\varepsilon))$, and let$\mathrm{Ext}_{in} : \{0,1\}^{n_{in}} \times \{0,1\}^{d_{in}} \to \{0,1\}^{d_{out}}$ be the $(n_{in} - \log(1/\delta), \varepsilon/3)$ average-case KL-extractor from Theorem 6.4 with output length d_{out}, so that $n_{in} = O(\log(1/\delta) + \log(1/\varepsilon))$ and $d_{in} = O(\log\log(1/\delta) + \log(1/\varepsilon))$.

Then instantiating Theorem 5.3 with Ext_{out} and Ext_{in} gives an $(n' - \log(1/\delta), \varepsilon)$ KL-extractor $\mathrm{Ext}' : \{0,1\}^{n'} \times \{0,1\}^{d'} \to \{0,1\}^m$ with $n' = m + n_{in}$ $d' = d_{in}$. The result follows follows from defining $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ by $\mathrm{Ext}(x, (s,t)) = \mathrm{Ext}'((x,s), t)$ for s of length $O(\log(1/\varepsilon))$. ◀

We can now prove Theorem 6.1.

Proof of Theorem 6.1. Let $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be the explicit $(k, \varepsilon^2/2)$ KL-extractor of Theorem 6.5 with $n = O(m + \log(1/\delta') + \log(1/\varepsilon))$, $k = n - \log(1/\delta')$, and $d = O(\log\log(1/\delta') + \log(1/\varepsilon))$ for $\delta' = \delta/2$. Then by Lemmas 4.8 and 5.1, Ext is also a (k, ε) extractor for $d_{\mathcal{E}}$, so by Theorem 3.5 the function $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^D$ for $D = 2^d$ defined by $\mathrm{Samp}(x)_i = \mathrm{Ext}(x, i)$ is a (δ', ε) subexponential sampler. Finally, by Remark 3.4, we have that Samp is a (δ, ε) absolute subexponential sampler as desired. ◀

In the full version [1] of this paper, in addition to proving the stronger version of Theorem 6.1, we also discuss explicit samplers for other ranges of parameters and non-explicit constructions.

References

1 Rohit Agrawal. Samplers and Extractors for Unbounded Functions. *arXiv:1904.08391 [cs]*, July 2019. `arXiv:1904.08391`.

2 Syed Mumtaz Ali and Samuel David Silvey. A General Class of Coefficients of Divergence of One Distribution from Another. *Journal of the Royal Statistical Society. Series B (Methodological)*, 28(1):131–142, 1966.

3 Koenraad M. R. Audenaert and Jens Eisert. Continuity Bounds on the Quantum Relative Entropy. *Journal of Mathematical Physics*, 46(10):102104, October 2005. doi:10.1063/1.2044667.

4 Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert, and Yu Yu. Leftover Hash Lemma, Revisited. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, Lecture Notes in Computer Science, pages 1–20. Springer Berlin Heidelberg, 2011.

5 Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Randomness in Interactive Proofs. *computational complexity*, 3(4):319–354, December 1993. doi:10.1007/BF01275487.

6 Mihir Bellare and John Rompel. Randomness-Efficient Oblivious Sampling. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 276–287, November 1994. doi:10.1109/SFCS.1994.365687.

7 Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy Amplification by Public Discussion. *SIAM Journal on Computing*, 17(2):210–229, April 1988. doi:10.1137/0217014.

8 Jarosław Błasiok. Private Communication, 2018.

9 Jarosław Błasiok. Optimal Streaming and Tracking Distinct Elements with High Probability. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, Proceedings, pages 2432–2448. Society for Industrial and Applied Mathematics, January 2018. doi:10.1137/1.9781611975031.156.

10 Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 1 edition edition, February 2013. doi:10.1093/acprof:oso/9780199535255.001.0001.

11 Ran Canetti, Guy Even, and Oded Goldreich. Lower Bounds for Sampling Algorithms for Estimating the Average. *Information Processing Letters*, 53(1):17–25, January 1995. doi:10.1016/0020-0190(94)00171-T.

12 Benny Chor and Oded Goldreich. On the Power of Two-Point Based Sampling. *Journal of Complexity*, 5(1):96–106, March 1989. doi:10.1016/0885-064X(89)90015-0.

13 Imre Csiszár. Eine Informationstheoretische Ungleichung Und Ihre Anwendung Auf Den Beweis Der Ergodizität von Markoffschen Ketten. *Magyar Tud. Akad. Mat. Kutató Int. Közl.*, 8:85–108, 1963.

14 Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM Journal on Computing*, 38(1):97–139, January 2008. doi:10.1137/060651380.

15 Monroe D. Donsker and S. R. Srinivasa Varadhan. Asymptotic Evaluation of Certain Markov Process Expectations for Large Time—III. *Communications on Pure and Applied Mathematics*, 29(4):389–461, 1976. doi:10.1002/cpa.3160290405.

16 David Gillman. A Chernoff Bound for Random Walks on Expander Graphs. *SIAM Journal on Computing*, 27(4):1203–1220, August 1998. doi:10.1137/S0097539794268765.

17 Oded Goldreich. A Sample of Samplers: A Computational Perspective on Sampling. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation: In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, Lecture Notes in Computer Science, pages 302–332. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-22670-0_24.

18 Oded Goldreich and Salil Vadhan. Comparing Entropies in Statistical Zero Knowledge with Applications to the Structure of SZK. In *Proceedings of the Fourteenth Annual IEEE Conference on Computational Complexity*, pages 54–73, May 1999. doi:10.1109/CCC.1999.766262.

19 Oded Goldreich and Avi Wigderson. Tiny Families of Functions with Random Properties: A Quality-Size Trade-off for Hashing. *Random Structures & Algorithms*, 11(4):315–343, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4<315::AID-RSA3>3.0.CO;2-1.

20 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced Expanders and Randomness Extractors from Parvaresh–Vardy Codes. *Journal of the ACM*, 56(4):20:1–20:34, July 2009. doi:10.1145/1538902.1538904.

21 Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-Random Generation from One-Way Functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, pages 12–24, New York, NY, USA, 1989. ACM. doi:10.1145/73007.73009.

22 Richard Karp, Nicholas Pippenger, and Michael Sipser. A Time-Randomness Tradeoff. In *AMS Conference on Probabilistic Computational Complexity*, Durham, New Hampshire, 1985.

23 James Lawrence McInnes. Cryptography Using Weak Sources of Randomness. Technical Report 194/87, University of Toronto, 1987.

24 Tetsuzo Morimoto. Markov Processes and the H-Theorem. *Journal of the Physical Society of Japan*, 18(3):328–331, March 1963. doi:10.1143/JPSJ.18.328.

25 Alfred Müller. Integral Probability Metrics and Their Generating Classes of Functions. *Advances in Applied Probability*, 29(2):429–443, 1997. doi:10.2307/1428011.

26 Noam Nisan and David Zuckerman. Randomness Is Linear in Space. *Journal of Computer and System Sciences*, 52(1):43–52, February 1996. doi:10.1006/jcss.1996.0004.

27 Jaikumar Radhakrishnan and Amnon Ta-Shma. Bounds for Dispersers, Extractors, and Depth-Two Superconcentrators. *SIAM Journal on Discrete Mathematics*, 13(1):2–24, January 2000. doi:10.1137/S0895480197329508.

28 Ran Raz, Omer Reingold, and Salil Vadhan. Extracting All the Randomness and Reducing the Error in Trevisan's Extractors. *Journal of Computer and System Sciences*, 65(1):97–128, August 2002. doi:10.1006/jcss.2002.1824.

29 Omer Reingold, Luca Trevisan, Madhur Tulsiani, and Salil Vadhan. New Proofs of the Green-Tao-Ziegler Dense Model Theorem: An Exposition. *arXiv:0806.0381 [math]*, June 2008. arXiv:0806.0381.

30 Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy Waves, the Zig-Zag Graph Product, and New Constant-Degree Expanders and Extractors. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 3–13, November 2000. doi:10.1109/SFCS.2000.892006.

31 Alfréd Rényi. On Measures of Entropy and Information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California, 1961.

32 Ofer Shayevitz. On Rényi Measures and Hypothesis Testing. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 894–898, July 2011. doi:10.1109/ISIT.2011.6034266.

33 Aravind Srinivasan and David Zuckerman. Computing with Very Weak Random Sources. *SIAM Journal on Computing*, 28(4):1433–1459, January 1999. doi:10.1137/S009753979630091X.

34 Amnon Ta-Shma, David Zuckerman, and Shmuel Safra. Extractors from Reed–Muller Codes. *Journal of Computer and System Sciences*, 72(5):786–812, August 2006. doi:10.1016/j.jcss.2005.05.010.

35 Salil P. Vadhan. *Pseudorandomness*. Now Publishers Inc, Boston, Mass., October 2012.

36 Tim van Erven. *When Data Compression and Statistics Disagree: Two Frequentist Challenges for the Minimum Description Length Principle*. PhD thesis, Leiden University, 2010. OCLC: 673140651.

37 Tim van Erven and Peter Harremoës. Rényi Divergence and Kullback-Leibler Divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, July 2014. doi:10.1109/TIT.2014.2320500.

38 Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Number 47 in Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 2018.

39 Vladimir Mikhailovich Zolotarev. Probability Metrics. *Theory of Probability & Its Applications*, 28(2):278–302, January 1984. doi:10.1137/1128025.

40 David Zuckerman. Randomness-Optimal Oblivious Sampling. *Random Structures & Algorithms*, 11(4):345–367, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4<345::AID-RSA4>3.0.CO;2-Z.

41 David Zuckerman. Linear Degree Extractors and the Inapproximability of Max Clique and Chromatic Number. *Theory of Computing*, 3(1):103–128, August 2007. doi:10.4086/toc.2007.v003a006.

A Missing proofs

In this section, we include some proofs that were omitted from the main text due to space constraints.

Proof of Proposition 2.11. As mentioned this is just the standard fact that the ℓ_p and ℓ_q norms are dual, but for completeness we include a proof in our language using the extremal form of Hölder's inequality (note that since we are dealing with finite probability spaces the extremal equality holds even for $p = \infty$ and $q = 1$). Given probability distributions A and B over $\{0,1\}^m$, we have that

$$\begin{aligned}
d_{\ell_p}(A,B) &= \left(\sum_x |A_x - B_x|^p\right)^{1/p} \\
&= 2^{m/p} \mathop{\mathbb{E}}_{x\sim U_m}\left[|A_x - B_x|^p\right]^{1/p} \\
&= 2^{m/p} \max_{\substack{f:\{0,1\}^m\to\mathbb{R} \\ \|f(U_m)\|_q \le 1}} \left| \mathop{\mathbb{E}}_{x\sim U_m}\left[f(x)(A_x - B_x)\right]\right| && \text{(Hölder's extremal equality)} \\
&= 2^{-m+m/p} \max_{\substack{f:\{0,1\}^m\to\mathbb{R} \\ \|f(U_m)\|_q \le 1}} \left|\mathbb{E}[f(A)] - \mathbb{E}[f(B)]\right| \\
&= 2^{-m/q} \cdot d_{\mathcal{M}_q}(A,B) && \text{(by symmetry of } \mathcal{M}_q)
\end{aligned}$$

as desired. ◀

Proof of Theorem 3.5. The proof is essentially the same as that of [40].

Fix a collection of test functions $f_1, \ldots, f_D \in \mathcal{F}$, where if Ext is not strong we restrict to $f_1 = \cdots = f_D$, and let $B_{f_1,\ldots,f_D} \subseteq \{0,1\}^n$ be defined as

$$\begin{aligned}
B_{f_1,\ldots,f_D} &\stackrel{\text{def}}{=} \left\{ x \in \{0,1\}^n \;\middle|\; \mathop{\mathbb{E}}_{i\sim U_{[D]}}\left[f_i(\mathrm{Ext}(x,i)) - \mathbb{E}[f_i(U_m)]\right] > \varepsilon \right\} \\
&= \left\{ x \in \{0,1\}^n \;\middle|\; \mathop{\mathbb{E}}_{i\sim U_{[D]}}\left[\mathrm{D}^{\{f_i\}}\left(U_{\{\mathrm{Ext}(x,i)\}} \parallel U_m\right)\right] > \varepsilon \right\},
\end{aligned}$$

where $U_{\{z\}}$ is the point mass on z. Then if X is uniform over $B_{f_1,\ldots,f_D}$, we have

$$\begin{aligned}
\varepsilon &< \mathop{\mathbb{E}}_{x\sim X}\left[\mathop{\mathbb{E}}_{i\sim U_{[D]}}\left[f_i(\mathrm{Ext}(x,i)) - \mathbb{E}[f_i(U_m)]\right]\right] \\
&= \mathop{\mathbb{E}}_{i\sim U_{[D]}}\left[\mathrm{D}^{\{f_i\}}(\mathrm{Ext}(X,i) \parallel U_m)\right]
\end{aligned}$$

$$\cdots = \begin{cases} \mathrm{D}^{\{f_1\}}(\mathrm{Ext}(X,U_d) \parallel U_m) & \text{if } f_1 = \cdots = f_D \\ \mathbb{E}_{i\sim U_{[D]}}\left[\mathrm{D}^{\{f_i\}}(\mathrm{Ext}(X,i) \parallel U_m)\right] & \text{always} \end{cases}$$
$$\leq \begin{cases} \mathrm{D}^{\mathcal{F}}(\mathrm{Ext}(X,U_d) \parallel U_m) & \text{if } f_1 = \cdots = f_D \\ \mathbb{E}_{i\sim U_{[D]}}\left[\mathrm{D}^{\mathcal{F}}(\mathrm{Ext}(X,i) \parallel U_m)\right] & \text{always} \end{cases}$$

Since Ext is an $(n-\log(1/\delta),\varepsilon)$-extractor (respectively strong extractor) for $\mathrm{D}^{\mathcal{F}}$ we must have $\mathrm{H}_\infty(X) < n - \log(1/\delta)$. But $\mathrm{H}_\infty(X) = \log|B_{f_1,\ldots,f_D}|$ by definition, so we have $|B_{f_1,\ldots,f_D}| < \delta 2^n$. Hence, the probability that a random $x \in \{0,1\}^n$ lands in $B_{f_1,\ldots,f_D}$ is less than δ, and since $B_{f_1,\ldots,f_D}$ is exactly the set of seeds which are bad for Samp, this concludes the proof. ◀

Proof of Theorem 3.7. Again the proof is analogous to the one in [40].

Fix a distribution X over $\{0,1\}^m$ with $\mathrm{H}_\infty(X) \geq k$ and a collection of test functions $f_1,\ldots,f_D \in \mathcal{F}$, where if Samp is not strong we restrict to $f_1 = \cdots = f_D$. Then since Samp is a (δ,ε) $\mathcal{F}$-sampler, we know that the set of seeds for which the sampler is bad must be small. Formally, the set

$$B_{f_1,\ldots,f_D} \stackrel{\text{def}}{=} \left\{ x \in \{0,1\}^n \;\middle|\; \mathop{\mathbb{E}}_{i\sim U_d}\left[f_i(\mathrm{Samp}(x)_i) - \mathbb{E}[f_i(U_m)]\right] > \varepsilon \right\}$$
$$= \left\{ x \in \{0,1\}^n \;\middle|\; \mathop{\mathbb{E}}_{i\sim U_d}\left[f_i(\mathrm{Ext}(x,i)) - \mathbb{E}[f_i(U_m)]\right] > \varepsilon \right\}$$

has size $|B_{f_1,\ldots,f_D}| \leq \delta 2^n$. Thus, since X has min-entropy at least k we know that $\Pr[X \in B_{f_1,\ldots,f_D}] \leq 2^{-k}\cdot\delta 2^n$, so we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{i\sim U_d}\left[\mathbb{E}\left[f_i(\mathrm{Ext}(X,i)) - \mathbb{E}[f_i(U_m)]\right]\right] \\
&\quad= \mathop{\mathbb{E}}_{X}\left[\mathop{\mathbb{E}}_{i\sim U_d}\left[f_i(\mathrm{Ext}(X,i)) - \mathbb{E}[f_i(U_m)]\right]\right] \\
&\quad= \Pr[X \in B_{f_1,\ldots,f_D}]\cdot \mathop{\mathbb{E}}_{X}\left[\mathop{\mathbb{E}}_{i\sim U_d}\left[f_i(\mathrm{Ext}(X,i)) - \mathbb{E}[f_i(U_m)]\right] \;\middle|\; X \in B_{f_1,\ldots,f_D}\right] \\
&\quad+ \Pr[X \notin B_{f_1,\ldots,f_D}]\cdot \mathop{\mathbb{E}}_{X}\left[\mathop{\mathbb{E}}_{i\sim U_d}\left[f_i(\mathrm{Ext}(X,i)) - \mathbb{E}[f_i(U_m)]\right] \;\middle|\; X \notin B_{f_1,\ldots,f_D}\right] \\
&\quad\leq \Pr[X \in B_{f_1,\ldots,f_D}]\cdot \max \mathrm{dev}(\mathcal{F}) + \Pr[X \notin B_{f_1,\ldots,f_D}]\cdot\varepsilon \\
&\quad\leq 2^{-k}\cdot\delta 2^n \cdot \max \mathrm{dev}(\mathcal{F}) + \varepsilon
\end{aligned}$$

completing the proof of the main claim. The "in particular" statement follows since if (Z,X) are jointly distributed with $\tilde{\mathrm{H}}_\infty(X|Z) \geq n - \log(1/\delta) + \log(1/\eta)$ we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{z\sim Z}\left[\varepsilon + \delta\cdot 2^{n-\mathrm{H}_\infty(X|_{Z=z})}\cdot \max \mathrm{dev}(\mathcal{F})\right] &= \varepsilon + \delta\cdot 2^{n-\tilde{\mathrm{H}}_\infty(X|Z)}\cdot \max \mathrm{dev}(\mathcal{F}) \\
&\leq \varepsilon + \eta\cdot \max \mathrm{dev}(\mathcal{F})
\end{aligned}$$

by definition of conditional min-entropy. ◀

Proof of Lemma 4.6. That $d_{TV} \leq d_{\mathcal{G}}$ is immediate from Hoeffding's lemma and the discussion in Remark 4.5. The reverse bound holds since any subgaussian function takes values at most $\sqrt{\ln 2/2\cdot m}$ away from the mean by the tail bounds from part 3 of Lemma 4.3, and so any subgaussian test function f has the property that $1/2 + f/\sqrt{2\ln 2\cdot m}$ is $[0,1]$-valued and thus lower bounds the total variation distance. ◀

Proof of Lemma 4.7. By Proposition 2.11, for any function $f : \{0,1\}^m \to \mathbb{R}$ it holds that

$$\mathrm{D}^{\{f\}}(P \parallel Q) \le \|f(U_m)\|_{1+\frac{1}{\alpha}} \cdot d_{\mathcal{M}_{1+\frac{1}{\alpha}}}(P,Q) = \|f(U_m)\|_{1+\frac{1}{\alpha}} \cdot 2^{m\alpha/(1+\alpha)} \cdot d_{\ell_{1+\alpha}}(P,Q).$$

The result follows since $[-1,1]$-valued functions f satisfy moment bounds $\|f(U_m)\|_q \le 1$ for all $q \ge 1$, and functions f which are subgaussian satisfy moment bounds $\|f(U_m)\|_q \le \sqrt{q}$ by Lemma 4.3. ◀

Proof of Lemma 4.8. The upper bound on subgaussian distance follows from a general form of Pinsker's inequality as in [10, Lemma 4.18], but for the extension to subexponential functions we reproduce its proof here, based on the Donsker–Varadhan "variational" formulation of KL divergence [15] (cf. [10, Corollary 4.15])

$$\mathrm{KL}(P \parallel U_m) = \frac{1}{\ln 2} \cdot \sup_{g:\{0,1\}^m \to \mathbb{R}} \Big(\mathbb{E}[g(P)] - \ln \mathbb{E}\Big[e^{g(U_m)}\Big] \Big).$$

Now if $f : \{0,1\}^m \to \mathbb{R}$ satisfies $\mathbb{E}[f(U_m)] = 0$, then by letting $g(x) = t \cdot f(x)$, this implies

$$\mathbb{E}[f(P)] - \mathbb{E}[f(U_m)] = \frac{1}{t} \cdot \mathbb{E}[g(P)] \le \frac{\ln 2 \cdot \mathrm{KL}(P \parallel U_m) + \ln \mathbb{E}\big[e^{t \cdot f(U_m)}\big]}{t}$$

for all $t > 0$. Thus, when $\ln \mathbb{E}\big[e^{t \cdot f(U_m)}\big] \le t^2/8$, we have $\mathbb{E}[f(P)] - \mathbb{E}[f(U_m)] \le \ln 2 \cdot \mathrm{KL}(P \parallel U_m)/t + t/8$.

Then since subgaussian random variables satisfy such a bound for all t, we can make the optimal choice $t = \sqrt{8 \ln 2 \cdot \mathrm{KL}(P \parallel U_m)}$ to get the claimed bound on $d_{\mathcal{G}}$. For subexponential random variables, which satisfy such a bound only for $|t| \le 2$, we choose $t = \min(\sqrt{8 \ln 2 \cdot \mathrm{KL}(P \parallel U_m)}, 2)$, which gives

$$d_{\mathcal{E}}(P, U_m) \le \begin{cases} \sqrt{\frac{\ln 2}{2} \cdot \mathrm{KL}(P \parallel U_m)} & \text{if } \mathrm{KL}(P \parallel U_m) \le \frac{1}{2 \ln 2} \\ \frac{\ln 2}{2} \cdot \mathrm{KL}(P \parallel U_m) + \frac{1}{4} & \text{if } \mathrm{KL}(P \parallel U_m) > \frac{1}{2 \ln 2} \end{cases}$$

as desired. The concavity of this bound follows by noting that it has a continuous and nonincreasing derivative.

For the reverse inequality, we use a bound on the difference in entropy between distributions P and Q on a set of size S which states

$$|H(P) - H(Q)| \le \lg(S-1) \cdot d_{TV}(P,Q) + h(d_{TV}(P,Q)).$$

This inequality is a simple consequence of Fano's inequality as noted by Goldreich and Vadhan [18, Fact B.1], and implies the desired result by taking $Q = U_m$ as $\mathrm{KL}(P \parallel U_m) = H(U_m) - H(P)$ and $|\{0,1\}^m| = 2^m$. ◀

▶ Remark A.1. There are sharper upper bounds on the KL divergence than given in Lemma 4.8, such as the bound of Audenaert and Eisert [3, Theorem 6], but the bound we use has the advantage of being defined for the entire range of the total variation distance and being everywhere concave.

Proof of Lemma 5.2. This follows from a characterization of Rényi divergence due to van Erven and Harremoës [36, Lemma 6.6] [37, Theorem 30] and Shayevitz [32, Theorem 1], who prove that for for every positive real $\beta \ne 1$ and distributions X and Y that

$$(1-\beta)\,\mathrm{D}_\beta(X \parallel Y) = \inf_Z \big\{ \beta\,\mathrm{KL}(Z \parallel X) + (1-\beta)\,\mathrm{KL}(Z \parallel Y) \big\}.$$

In particular, choosing $\beta = 1 + \alpha$, $X = Q$, and $Y = R$ and upper bounding the infimum by the particular choice of $Z = P$ gives the claim. ◀

Proof of Theorem 5.3. Let (X, Y) be jointly distributed random variables with X distributed over $\{0,1\}^n$ and Y over $\{0,1\}^{n'}$ such that $\tilde{\mathrm{H}}_\infty(X, Y|Z) \geq n + n' - \log(1/\delta)$. Then by Lemma 5.2 and the data-processing inequality for KL divergence we have that

$$\begin{aligned}
&\mathrm{KL}(\mathrm{Ext}((X,Y), U_d) \parallel U_m) \\
&\quad = \mathrm{KL}(\mathrm{Ext}_{out}(X, \mathrm{Ext}_{in}(Y, U_d)) \parallel U_m) \\
&\quad \leq (1 + 1/\alpha) \cdot \mathrm{KL}(\mathrm{Ext}_{out}(X, \mathrm{Ext}_{in}(Y, U_d)) \parallel \mathrm{Ext}_{out}(X, U_d)) \\
&\qquad\quad + \mathrm{D}_{1+\alpha}(\mathrm{Ext}_{out}(X, U_d) \parallel U_m) \\
&\quad \leq (1 + 1/\alpha) \cdot \mathrm{KL}(X, \mathrm{Ext}_{in}(Y, U_d) \parallel X, U_d) + \mathrm{D}_{1+\alpha}(\mathrm{Ext}_{out}(X, U_d) \parallel U_m) \\
&\quad = (1 + 1/\alpha) \cdot \mathop{\mathbb{E}}_{x \sim X}[\mathrm{KL}(\mathrm{Ext}_{in}(Y|_{X=x}, U_d) \parallel U_d)] + \mathrm{D}_{1+\alpha}(\mathrm{Ext}_{out}(X, U_d) \parallel U_m)
\end{aligned}$$

where the last equality follows from the chain rule for KL divergence. Now by standard properties of conditional min-entropy (see for example [14, Lemma 2.2]), we know that $\mathrm{H}_\infty(X) \geq \mathrm{H}_\infty(X, Y) - \log|\mathrm{Supp}(Y)| \geq n - \log(1/\delta)$ and $\tilde{\mathrm{H}}_\infty(Y|X) \geq \mathrm{H}_\infty(X, Y) - \log|\mathrm{Supp}(X)| \geq n' - \log(1/\delta)$. Thus, since by assumption Ext_{in} is an average-case $(n' - \log(1/\delta), \varepsilon_{in})$ KL-extractor the first term is bounded by $(1 + 1/\alpha) \cdot \varepsilon_{in}$, and similarly since Ext_{out} is an $(n - \log(1/\delta), \varepsilon_{out})$ $\mathrm{D}_{1+\alpha}$-extractor we have that the second term is bounded by ε_{out} as desired. ◀

Successive Minimum Spanning Trees



Svante Janson
Department of Mathematics, Uppsala University, PO Box 480, SE-751 06 Uppsala, Sweden
http://www.math.uu.se/svante-janson/
svante.janson@math.uu.se



Gregory B. Sorkin
Department of Mathematics, The London School of Economics and Political Science, Houghton Street, London WC2A 2AE, England
http://personal.lse.ac.uk/sorkin/
g.b.sorkin@lse.ac.uk

Abstract

In a complete graph K_n with edge weights drawn independently from a uniform distribution $U(0,1)$ (or alternatively an exponential distribution Exp(1)), let T_1 be the MST (the spanning tree of minimum weight) and let T_k be the MST after deletion of the edges of all previous trees T_i, $i < k$. We show that each tree's weight $w(T_k)$ converges in probability to a constant γ_k with $2k - 2\sqrt{k} < \gamma_k < 2k + 2\sqrt{k}$, and we conjecture that $\gamma_k = 2k - 1 + o(1)$. The problem is distinct from that of Frieze and Johansson [6], finding k MSTs of combined minimum weight, and the combined cost for two trees in their problem is, asymptotically, strictly smaller than our $\gamma_1 + \gamma_2$.

Our results also hold (and mostly are derived) in a multigraph model where edge weights for each vertex pair follow a Poisson process; here we additionally have $\mathbb{E}(w(T_k)) \to \gamma_k$. Thinking of an edge of weight w as arriving at time $t = nw$, Kruskal's algorithm defines forests $F_k(t)$, each initially empty and eventually equal to T_k, with each arriving edge added to the first $F_k(t)$ where it does not create a cycle. Using tools of inhomogeneous random graphs we obtain structural results including that $C_1(F_k(t))/n$, the fraction of vertices in the largest component of $F_k(t)$, converges in probability to a function $\rho_k(t)$, uniformly for all t, and that a giant component appears in $F_k(t)$ at a time $t = \sigma_k$. We conjecture that the functions ρ_k tend to time translations of a single function, $\rho_k(2k + x) \to \rho_\infty(x)$ as $k \to \infty$, uniformly in $x \in \mathbb{R}$.

Simulations and numerical computations give estimated values of γ_k for small k, and support the conjectures stated above.

2012 ACM Subject Classification Mathematics of computing → Random graphs; Mathematics of computing → Paths and connectivity problems; Mathematics of computing → Combinatorial optimization; Mathematics of computing → Matroids and greedoids

Keywords and phrases miminum spanning tree, second-cheapest structure, inhomogeneous random graph, optimization in random structures, discrete probability, multi-type branching process, functional fixed point, robust optimization, Kruskal's algorithm

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.60

Category RANDOM

Related Version A full version of the paper is available at https://arxiv.org/abs/1906.01533.

Funding The work was partly supported by the Knut and Alice Wallenberg Foundation.

Acknowledgements We thank Oliver Riordan for helpful comments which simplified our proof, and Balázs Mezei for assistance with Julia programming.


© Svante Janson and Gregory B. Sorkin;
licensed under Creative Commons License CC-BY

1 Introduction

1.1 Problem definition and main results

Consider the complete graph K_n with edge costs that are i.i.d. random variables, with a uniform distribution $U(0,1)$ or, alternatively, an exponential distribution Exp(1). A well-known problem is to find the minimum (cost) spanning tree T_1, and its cost or "weight" $w(T_1)$. A famous result by Frieze [7] shows that as $n \to \infty$, $w(T_1)$ converges in probability to $\zeta(3)$, in both the uniform and exponential cases.

Suppose now that we want a second spanning tree T_2, edge-disjoint from the first, and that we do this in a greedy fashion by first finding the minimum spanning tree T_1, and then the minimum spanning tree T_2 using only the remaining edges. (I.e., T_2 is the minimum spanning tree in $K_n \setminus T_1$, meaning the graph with edge set $E(K_n) \setminus E(T_1)$.) We then continue and define T_3 as the minimum spanning tree in $K_n \setminus (T_1 \cup T_2)$, and so on. The main purpose of the present paper is to show that the costs $w(T_2)$, $w(T_3)$, ... also converge in probability to some constants.

▶ **Theorem 1.** *For each $k \geqslant 1$, there exists a constant γ_k such that, as $n \to \infty$, $w(T_k) \xrightarrow{\mathrm{p}} \gamma_k$ (for both uniform and exponential cost distributions).*

The result extends easily to other distributions of the edge costs (see full version for details), but we consider in this paper only the uniform and exponential cases.

A minor technical problem is that T_2 and subsequent trees do not always exist; it may happen that T_1 is a star and then $K_n \setminus T_1$ is disconnected. This happens only with a small probability, and w.h.p. (with high probability, i.e., with probability $1 - o(1)$ as $n \to \infty$) T_k is defined for every fixed k; see the full version for details. However, in the main part of the paper we avoid this problem completely by modifying the model: we assume that we have a multigraph, which we denote by K_n^∞, with an infinite number of copies of each edge in K_n, and that each edge's copies' costs are given by the points in a Poisson process with intensity 1 on $[0,\infty)$. (The Poisson processes for different edges are, of course, independent.) Note that when finding T_1, we only care about the cheapest copy of each edge, and its cost has an Exp(1) distribution, so the problem for T_1 is the same as the original one. However, on K_n^∞ we never run out of edges and we can define T_k for all integers $k = 1, 2, 3, \dots$. Asymptotically, the three models are equivalent (see full version for details), and Theorem 1 holds for any of the models. In particular:

▶ **Theorem 2.** *For each $k \geqslant 1$, as $n \to \infty$, $w(T_k) \xrightarrow{\mathrm{p}} \gamma_k$ also for the multigraph model with Poisson process costs.*

Frieze [7] also proved that the expectation $\mathbb{E}\, w(T_1)$ converges to $\zeta(3)$. For the multigraph model just described, this too extends.

▶ **Theorem 3.** *For the Poisson multigraph model, $\mathbb{E}\, w(T_k) \to \gamma_k$ for each $k \geqslant 1$ as $n \to \infty$.*

1.2 Motivations

Frieze and Johansson [6] recently considered a related problem, where instead of choosing spanning trees $T_1, T_2, \dots$ greedily one by one, they choose k edge-disjoint spanning trees with minimum total cost. It is easy to see, by small examples, that selecting k spanning trees greedily one by one does not always give a set of k edge-disjoint spanning trees with minimum cost, so the problems are different.

We show in Theorem 19 that, at least for $k = 2$, the two problems also asymptotically have different answers, in the sense that the limiting values of the minimum cost – which exist for both problems – are different. (Also, as discussed in Section 3.1, we improve on the upper bound from [6, Section 3] on the cost of the net cheapest k trees, since our upper bound (3.1) on the cost of the first k trees is smaller.)

Both our question and that of Frieze and Johansson [6] are natural, both seem generally relevant to questions of robust network design, and both have mathematically interesting answers.

Another reason for interest in T_2 comes from the field of algorithmic mechanism design. Imagine that each edge of $G = K_n$ is owned by a different "agent"; the agent owning edge e values it at $w(e)$, an amount known only to them. We, an "auctioneer", want to buy a spanning tree, at low cost. One "mechanism" for doing so is a sealed-bid auction where each agent posts a price $w'(e)$ for their edge, and we buy the tree that is cheapest according to these prices. Here, agents will naturally inflate their prices, posting prices $w'(e) > w(e)$.

One alternative is a VCG (Vickrey–Clarke–Groves) auction, a generalization of a single-item second-price auction. Here, we again buy the tree that is cheapest according to the posted prices w', but for each edge e purchased, we pay an amount that is a function of w'_{-e}, i.e., of all posted prices *except* that of e; for details see for example [16, Chapter 9]. This means that varying $w'(e)$ affects only whether edge e is purchased, not how much is paid for it if it is, and results in the mechanism being *truthful*: it is in each agent's selfish interest to set $w'(e) = w(e)$. Thus, the tree purchased is simply T_1, the tree cheapest according to the values w. However, the amount paid for it is more than $w(T_1)$, as the mechanism ensures the amount paid for each edge e purchased is at least $w(e)$ and typically more. A central question is the extent of this overpayment, measured by the "frugality ratio" of the VCG cost V (or that of any mechanism) to some benchmark.

The question applies of course to problems other than MSTs, including the purchase of a cheapest path between two given points in a graph, or of a basis in a bridgeless matroid. In any of these contexts, let us continue to use T_1 for the cheapest structure and T_2 for the cheapest structure disjoint from T_1. The cost $w(T_1)$ is not a useful benchmark because $V/w(T_1)$ is unbounded in even the simplest examples (such as buying one of two identical items).

Instead, Talwar [17] and Archer and Tardos [1] propose $w(T_2)$ as the benchmark. (An often-equivalent benchmark, based on a Nash equilibrium, is given by [14] and [16, Chapter 13].) [17] shows that for any bridgeless matroid, $V/w(T_2) \leqslant 1$, and, focusing on the worst case over all weights w, this bound is achieved by some weights (namely weights 0 on T_1, 1 on T_2, and infinity elsewhere). By contrast, for paths the ratio is unbounded. The interpretation, based on worst-case weights, is that this frugality ratio is 1 for amenable problems like MSTs and other matroids, and larger for other problems.

In our setting of an MST in K_n with random weights, though, the frugality ratio is naturally less than its maximum of 1. Specifically, [4] and [11] show that the VCG cost is typically $2w(T_1)$, which by [7] is $2\zeta(3) \doteq 2.4041$. We show here that $w(T_2)$ is typically γ_2, which by Remark 21 is at least 2.9683, making the frugality typically at most 0.8099. (We estimate non-rigorously that γ_2 is about 3.09 – see Table 1 – in which case the frugality ratio is typically about 0.78.) Specifically, this holds w.h.p. for n large, and also holds for the ratio between the expected VCG cost and the expected cost $w(T_2)$.

1.3 Further results, structural properties, and conjectures

It is well known that the minimum spanning tree (with any given costs, obtained randomly or deterministically) can be found by *Kruskal's algorithm* [15], which processes the edges in order of increasing cost and keeps those that join two different components in the forest obtained so far. (I.e., it keeps each edge that does not form a cycle together with previously chosen edges.) As in many other previous papers on the random minimum spanning tree problem, from [7] on, our proofs are based on analyzing the behavior of this algorithm.

Rescale weight as time, thinking of an edge of weight w as arriving at time $t = nw$. Kruskal's algorithm allows us to construct all trees T_k simultaneously by growing forests $F_k(t)$, with $F_k(0)$ empty and $F_k(\infty) = T_k$: taking the edges of K_n (or K_n^∞) in order of time arrival (increasing cost), an edge is added to the first forest F_k where it does not create a cycle. We will also consider a sequence of graphs $G_k(t) \supseteq F_k(t)$, where when we add an edge to F_k we also add it to all the graphs $G_1, \dots, G_k$; see Section 2.2 for details.

The proof of Theorem 1 is based on a detailed structural characterization of the graphs $G_k(t)$, given by Theorem 9 (too detailed to set forth in full here in the introduction), relying heavily on the theory of inhomogeneous random graphs from [3] and related works. Where $C_1(G_k(t))$ denotes the number of vertices in the largest component of $G_k(t)$ (or equivalently of $F_k(t)$, as by construction they have the same components), Theorem 9 shows that $C_1(G_k(t))/n$ converges in probability to some function $\rho_k(t)$, uniformly for all times t. Moreover, each G_k has its own giant-component threshold: $\rho_k(t)$ is 0 until some time σ_k, and strictly positive thereafter.

The functions $\rho_k(t)$ are of central interest. For one thing, an edge is rejected from F_k, making it a candidate for F_{k+1}, precisely if its two endpoints are within the same component of F_k, and we show that this is essentially equivalent to the two endpoints both being within the largest component. This line of reasoning yields the constants γ_k explicitly, albeit not in a form that is easily evaluated. We are able, at least, to re-prove that $\gamma_1 = \zeta(3)$, as first shown in [7].

The functions ρ_k also appear to have a beautiful structure, tending to time-translated copies of a single universal function:

▶ **Conjecture 4.** *There exists a continuous increasing function $\rho_\infty(x) : (-\infty, \infty) \to [0, 1)$ such that $\rho_k(2k + x) \to \rho_\infty(x)$ as $k \to \infty$, uniformly in $x \in \mathbb{R}$.*

This suggests, though does not immediately imply, another conjecture.

▶ **Conjecture 5.** *For some δ, as $k \to \infty$, $\gamma_k = 2k + \delta + o(1)$.*

If this conjecture holds, then necessarily $\delta \in [-1, 0]$, see Remark 17.

A variety of computational results are given in Section 5. They are supportive of Conjecture 4 and a stronger version of Conjecture 5 where we take $\delta = -1$:

▶ **Conjecture 6.** *As $k \to \infty$, $\gamma_k = 2k - 1 + o(1)$.*

Although we cannot prove these conjectures, some bounds on γ_k are obtained in Section 3 by a more elementary analysis of the sequence of forests F_k. In particular, Theorem 12 and Corollary 13 lead to the following, implying that $\gamma_k \sim 2k$ as $k \to \infty$.

▶ **Corollary 7.** *For every $k \geqslant 1$,*

$$2k - 2k^{1/2} < \gamma_k < 2k + 2k^{1/2}. \tag{1.1}$$

▶ Remark 8. For the minimum spanning tree T_1, various further results are known, including refined estimates for the expectation of the cost $w(T_1)$ [5], a normal limit law [9], and asymptotics for the variance [9, 13, 18]. It seems challenging to show corresponding results for T_2 or later trees. ◀

1.4 Notes on this extended abstract

A full version of this work can be found as [12]. The present extended abstract omits most proofs as well as many further results. However, Sections 2 and 3 here are reasonably complete. We will say a few words in Section 2.5 on the approach to proving Theorem 9, but the technicalities are substantial.

2 Model and main structural results

2.1 Some notation

We use $:=$ as defining its left-hand side, and $\stackrel{\text{def}}{=}$ as a reminder that equality of the two sides is by definition. We write $\doteq$ for numerical approximate equality, and $\approx$ for approximate equality in an asymptotic sense (details given where used).

If x and y are real numbers, then $x \vee y := \max(x, y)$ and $x \wedge y := \min(x, y)$. Furthermore, $x_+ := x \vee 0$. These operators bind most strongly, e.g., $t - \tau(i) \vee \tau(j)$ means $t - (\tau(i) \vee \tau(j))$.

We use "increasing" and "decreasing" in their weak senses; for example, a function f is increasing if $f(x) \leqslant f(y)$ whenever $x \leqslant y$.

Unspecified limits are as $n \to \infty$. As said above, w.h.p. means with probability $1 - o(1)$. Convergence in probability is denoted $\xrightarrow{\text{p}}$. Furthermore, if X_n are random variables and a_n are positive constants, $X_n = o_{\text{p}}(a_n)$ means, as usual, $X_n/a_n \xrightarrow{\text{p}} 0$; this is also equivalent to: for every $\varepsilon > 0$, w.h.p. $|X_n| < \varepsilon a_n$.

Graph means, in general, multigraph. (It is usually clear from the context whether we consider a multigraph or simple graph.) If G is a multigraph, then $\dot{G}$ denotes the simple graph obtained by merging parallel edges and deleting loops. (Loops do not appear in the present paper.) The number of vertices in a graph G is denoted by $|G|$, and the number of edges by $e(G)$.

For a graph G, let $\mathcal{C}_1(G)$, $\mathcal{C}_2(G)$, ... be the largest component, the second largest component, and so on, using any rule to break ties. (If there are less than k components, we define $\mathcal{C}_k(G) = \emptyset$.) Furthermore, let $C_i(G) := |\mathcal{C}_i(G)|$; thus $C_1(G)$ is the the number of vertices in the largest component, and so on. We generally regard components of a graph G as sets of vertices.

2.2 Model

We elaborate the multigraph model in the introduction.

We consider (random) (multi)graphs on the vertex set $[n] := \{1, \dots, n\}$; we usually omit n from the notation. The graphs will depend on time, and are denoted by $G_k(t)$ and $F_k(t)$, where $k = 1, 2, 3, \dots$ and $t \in [0, \infty]$; they all start as empty at time $t = 0$ and grow as time increases. We will have $G_k(t) \supseteq G_{k+1}(t)$ and $F_k(t) \subseteq G_k(t)$ for all k and t. Furthermore, $F_k(t)$ will be a forest. As $t \to \infty$, $F_k(t)$ will eventually become a spanning tree, $F_k(\infty)$, which is the kth spanning tree T_k produced by the greedy algorithm in the introduction, operating on the multigraph $G_1(\infty)$.

Since the vertex set is fixed, we may when convenient identify the multigraphs with sets of edges. We begin by defining $G_1(t)$ by letting edges arrive as independent Poisson processes with rate $1/n$ for each pair $\{i, j\}$ of vertices; $G_1(t)$ consists of all edges that have arrived at

or before time t. (This scaling of time turns out to be natural and useful. In essence this is because what is relevant is the cheapest edges on each vertex, and these have expected cost $\Theta(1/n)$ and thus appear at expected time $\Theta(1)$.) We define the cost of an edge arriving at time t to be t/n, and note that in $G_1(\infty)$, the costs of the edges joining two vertices form a Poisson process with rate 1. Hence, $G_1(\infty)$ is the multigraph model defined in Section 1.

Thus, for any fixed $t \geqslant 0$, $G_1(t)$ is a multigraph where the number of edges between any two fixed vertices is $\mathrm{Po}(t/n)$, and these numbers are independent for different pairs of vertices. This is a natural multigraph version of the Erdős–Rényi graph $G(n,t)$. (The process $G_1(t)$, $t \geqslant 0$, is a continuous-time version of the multigraph process in e.g. [2] and [10, Section 1], ignoring loops.) Note that $\dot{G}_1(t)$, i.e., $G_1(t)$ with multiple edges merged, is simply the random graph $G(n,p)$ with $p = 1 - e^{-t/n}$.

Next, we let $F_1(t)$ be the subgraph of $G_1(t)$ consisting of every edge that has arrived at some time $s \leqslant t$ and at that time joined two different components of $G_1(s)$. Thus, this is a subforest of $G_1(t)$, as stated above, and it is precisely the forest constructed by Kruskal's algorithm (recalled in the introduction) operating on $G_1(\infty)$, at the time all edges with cost $\leqslant t/n$ have been considered. Hence, $F_1(\infty)$ is the minimum spanning tree T_1 of $G_1(\infty)$.

Let $G_2(t) := G_1(t) \setminus F_1(t)$, i.e., the subgraph of $G_1(t)$ consisting of all edges rejected from $F_1(t)$; in other words $G_2(t)$ consists of the edges that, when they arrive to $G_1(t)$, have their endpoints in the same component.

We continue recursively. $F_k(t)$ is the subforest of $G_k(t)$ consisting of all edges in $G_k(t)$ that, when they arrived at some time $s \leqslant t$, joined two different components in $G_k(s)$. And $G_{k+1}(t) := G_k(t) \setminus F_k(t)$, consisting of the edges rejected from $F_k(t)$.

Hence, the kth spanning tree T_k produced by Kruskal's algorithm equals $F_k(\infty)$, as asserted above.

Note that $F_k(t)$ is a spanning subforest of $G_k(t)$, in other words, the components of $F_k(t)$ (regarded as vertex sets) are the same as the components of $G_k(t)$; this will be used frequently below. Moreover, each edge in $G_{k+1}(t)$ has endpoints in the same component of $G_k(t)$; hence, each component of $G_{k+1}(t)$ is a subset of a component of $G_k(t)$. It follows that an edge arriving to $G_1(t)$ will be passed through $G_2(t), \dots, G_k(t)$ and to $G_{k+1}(t)$ (and possibly further) if and only if its endpoints belong to the same component of $G_k(t)$, and thus if and only if its endpoints belong to the same component of $F_k(t)$.

2.3 More notation

We say that a component $\mathcal{C}$ of a graph G is the *unique giant* of G if $|\mathcal{C}| > |\mathcal{C}'|$ for every other component $\mathcal{C}'$; if there is no such component (i.e., if the maximum size is tied), then we define the unique giant to be $\emptyset$.

We say that a component $\mathcal{C}$ of $F_k(t)$ is the *permanent giant* of $F_k(t)$ (or of $G_k(t)$) if it is the unique giant of $F_k(t)$ and, furthermore, it is a subset of the unique giant of $F_k(u)$ for every $u > t$; if there is no such component then the permanent giant is defined to be $\emptyset$.

Let $\mathfrak{C}_k(t)$ denote the permanent giant of $F_k(t)$. Note that the permanent giant either is empty or the largest component; thus $|\mathfrak{C}_k(t)|$ is either 0 or $C_1(F_k(t)) = C_1(G_k(t))$. Note also that the permanent giant $\mathfrak{C}_k(t)$ is an increasing function of t: $\mathfrak{C}_k(t) \subseteq \mathfrak{C}_k(u)$ if $t \leqslant u$. Furthermore, for sufficiently large t (viz. t such that $G_k(t)$ is connected, and thus $F_k(t)$ is the spanning tree T_k), $\mathfrak{C}_k(t) = \mathfrak{C}_k(\infty) = [n]$.

2.4 A structure theorem

The basis of our proof of Theorems 1 and 2 is the following theorem on the structure of the components of $G_k(t)$. Recall that $F_k(t)$ has the same components as $G_k(t)$, so the theorem applies as well to $F_k(t)$.

For $k = 1$, the theorem collects various known results for $G(n, p)$. Our proof includes this case too, making the proof more self-contained.

▶ **Theorem 9.** *With the definitions above, the following hold for every fixed $k \geqslant 1$ as $n \to \infty$.*

(i) *There exists a continuous increasing function $\rho_k : [0, \infty) \to [0, 1)$ such that*

$$C_1(G_k(t))/n \xrightarrow{\mathrm{p}} \rho_k(t), \tag{2.1}$$

uniformly in $t \in [0, \infty)$; in other words, for any $\varepsilon > 0$, w.h.p., for all $t \geqslant 0$,

$$\rho_k(t) - \varepsilon \leqslant C_1(G_k(t))/n \leqslant \rho_k(t) + \varepsilon. \tag{2.2}$$

(ii) $\sup_{t \geqslant 0} C_2(G_k(t))/n \xrightarrow{\mathrm{p}} 0$.

(iii) *There exists a threshold $\sigma_k > 0$ such that $\rho_k(t) = 0$ for $t \leqslant \sigma_k$, but $\rho_k(t) > 0$ for $t > \sigma_k$. Furthermore, ρ_k is strictly increasing on $[\sigma_k, \infty)$.*

(iv) *There exist constants $b_k, B_k > 0$ such that*

$$\rho_k(t) \geqslant 1 - B_k e^{-b_k t}, \qquad t \geqslant 0. \tag{2.3}$$

In particular, $\rho_k(t) \to 1$ as $t \to \infty$.

(v) *If $t > \sigma_k$, then w.h.p. $G_k(t)$ has a non-empty permanent giant. Hence, for every $t \geqslant 0$,*

$$|\mathfrak{C}_k(t)|/n \xrightarrow{\mathrm{p}} \rho_k(t). \tag{2.4}$$

We note also a formula for the number of edges in $G_k(t)$, and two simple inequalities relating different k.

▶ **Theorem 10.** *For each fixed $k \geqslant 1$ and uniformly for t in any finite interval $[0, T]$,*

$$e(G_k(t))/n \xrightarrow{\mathrm{p}} \frac{1}{2} \int_0^t \rho_{k-1}(s)^2 \, \mathrm{d}s. \tag{2.5}$$

▶ **Theorem 11.** *$\rho_k(t) \leqslant \rho_{k-1}(t)$ for every $t \geqslant 0$, with strict inequality when $\rho_{k-1}(t) > 0$ (equivalently, when $t > \sigma_{k-1}$). Furthermore,*

$$\sigma_k \geqslant \sigma_{k-1} + 1. \tag{2.6}$$

Inequality (2.6) is weak in that we conjecture that as $k \to \infty$, $\sigma_k = \sigma_{k-1} + 2 + o(1)$.

2.5 The proof approach

Proofs of the results in this section are by induction on k, relying heavily on the theory of inhomogeneous random graphs by Bollobás, Janson and Riordan in [3]. When an edge is passed on by G_k this is almost always because it is contained in $\mathcal{C}_1(G_k)$; it is only rarely because it is contained in some other component, and this case is treatable as a perturbation within the theory. Thus, vertices "appear" in $G_{k+1}(t)$ as governed by $\rho_k(t)$; this is formalized as a "vertex space" in the theory. Once two vertices u and v are both present in $G_{k+1}(t)$, edges between them arrive at rate $1/n$. So, if they arrive at times τ_u and τ_v, the probability they are connected at time t is asymptotically $\frac{1}{n}(t - (\tau_u \vee \tau_v))_+ =: \frac{1}{n}\kappa_t(\tau_u, \tau_v)$; κ_t is the "kernel" in the inhomogeneous random graph framework. The framework then shows that $C_1(G_{k+1}(t))/n$ converges in probability to a certain $\rho(\kappa_t)$, the survival probability of a related inhomogeneous branching process, and this $\rho(\kappa_t)$ is precisely the desired next function $\rho_{k+1}(t)$.

3 Bounds on the expected cost

3.1 Total cost of the first k trees

The following theorem gives lower and upper bounds on the total cost of the first k spanning trees.

▶ **Theorem 12.** *Letting $W_k = \sum_{i=1}^{k} w(T_i)$ be the total cost of the first k spanning trees, for every $k \geqslant 1$,*

$$k^2 \frac{n-1}{n} \leqslant \mathbb{E}\, W_k \leqslant k(k+1)\frac{n-1}{n} < k^2 + k. \tag{3.1}$$

Comparing with Frieze and Johansson [6, Section 3], our upper bound is smaller than their $k^2 + 3k^{5/3}$ despite the fact that they considered a more relaxed minimization problem (see Section 4); as such ours is a strict improvement. In both cases the lower bound is simply the expected total cost of the cheapest $k(n-1)$ edges in G, with (3.2) matching [6, (3.1)].

Proof. The minimum possible cost of the k spanning trees is the cost of the cheapest $k(n-1)$ edges. Since each edge's costs (plural, in our model) are given by a Poisson process of rate 1, the set of all edge costs is given by a Poisson process of rate $\binom{n}{2}$. Recall that in a Poisson process of rate λ, the interarrival times are independent exponential random variables with mean $1/\lambda$, so that the ith arrival, at time Z_i, has $\mathbb{E}\, Z_i = i/\lambda$. It follows in this case that $W_k \geqslant \sum_{i=1}^{k(n-1)} Z_i$ and

$$\mathbb{E}\, W_k \geqslant \sum_{i=1}^{k(n-1)} \frac{i}{\binom{n}{2}} = \frac{(k(n-1))(k(n-1)+1)}{n(n-1)} \geqslant k^2 \frac{n-1}{n}. \tag{3.2}$$

We now prove the upper bound. An arriving edge is rejected from F_i iff both endpoints lie within its "forbidden" set B_i of edges, namely those edges with both endpoints in one component. The nesting property of the components means that $B_1 \supseteq B_2 \supseteq \cdots$. An arriving edge e joins F_k if it is rejected from all previous forests, i.e., $e \in B_{k-1}$ (in which case by the nesting property, e also belongs to all earlier Bs) but can be accepted into F_k, i.e., $e \notin B_k$. The idea of the proof is to show that the first k forests fill reasonably quickly with $n-1$ edges each, and we will do this by coupling the forest-creation process (Kruskal's algorithm) to a simpler, easily analyzable random process.

Let $\mathbf{s}(\tau) = \{s_k(\tau)\}_{k=0}^{\infty}$ denote the vector of the sizes (number of edges) of each forest after arrival of the τ'th edge; we may drop the argument τ when convenient. Let $p_k = |B_k|/\binom{n}{2}$, the rejection probability for F_k. For any τ, by the nesting property of the components and in turn of the B_k,

$$s_1 \geqslant s_2 \geqslant \cdots \qquad \text{and} \qquad p_1 \geqslant p_2 \geqslant \cdots. \tag{3.3}$$

The MST process can be simulated by using a sequence of i.i.d. random variables $\alpha(\tau) \sim U(0,1)$, incrementing $s_k(\tau)$ if both $\alpha(\tau) \leqslant p_{k-1}(\tau)$ (so that e is rejected from F_{k-1} and thus from all previous forests too) and $\alpha(\tau) > p_k(\tau)$ (so that e is accepted into F_k). We take the convention that $p_0(\tau) = 1$ for all τ. For intuition, note that when $s_k = 0$ an edge is never rejected from F_k ($p_k = 0$, so $\alpha \sim U(0,1)$ is never smaller); when $s_k = 1$ it is rejected with probability $p_k = 1/\binom{n}{2}$; and when $s_k = n-1$ it is always rejected ($|B_k|$ must be $\binom{n}{2}$, so $p_k = 1$).

Given the size $s_k = \sum_{i=1}^{\infty}(C_i(F_k)-1)$ of the kth forest, $|B_k| = \sum_{i=1}^{\infty}\binom{C_i(F_k)}{2}$ is maximized (thus so is p_k) when all the edges are in one component, i.e.,

$$p_k \leqslant \binom{s_k+1}{2} \Big/ \binom{n}{2} \tag{3.4}$$

$$\leqslant \frac{s_k}{n-1} =: \bar{p}_k. \tag{3.5}$$

The size vector $\mathbf{s}(\tau)$ thus determines the values $\bar{p}_k(\tau)$ for all k.

Let $\mathbf{r}(\tau)$ denote a vector analogous to $\mathbf{s}(\tau)$, but with $r_k(\tau)$ incremented if $\hat{p}_k(\tau) < \alpha(\tau) \leqslant \hat{p}_{k-1}(\tau)$, with

$$\hat{p}_k := \frac{r_k}{n-1}. \tag{3.6}$$

By construction,

$$r_1 \geqslant r_2 \geqslant \cdots \qquad \text{and} \qquad \hat{p}_1 \geqslant \hat{p}_2 \geqslant \cdots. \tag{3.7}$$

For intuition, here note that when $r_k = 0$ an arrival is never rejected from r_k ($\bar{p}_k = 0$); when $s_k = 1$ it is rejected with probability $\bar{p}_k = 1/(n-1) > p_k = 1/\binom{n}{2}$; and when $s_k = n-1$ it is always rejected ($\bar{p}_k = 1$).

Taking each $F_i(0)$ to be an empty forest (n isolated vertices, no edges) and accordingly $\mathbf{s}(0)$ to be an infinite-dimensional 0 vector, and taking $\mathbf{r}(0)$ to be the same 0 vector, we claim that for all τ, $\mathbf{s}(\tau)$ majorizes $\mathbf{r}(\tau)$, which we will write as $\mathbf{s}(\tau) \succeq \mathbf{r}(\tau)$. That is, the prefix sums of $\mathbf{s}$ dominate those of $\mathbf{r}$: for all τ and k, $\sum_{i=1}^{k} s_i(\tau) \geqslant \sum_{i=1}^{k} r_i(\tau)$.

We first prove this; then use it to argue that edge arrivals to the first k forests, i.e., to $\mathbf{s}$, can only precede arrivals to the first k elements of $\mathbf{r}$; and finally analyze the arrival times of all $k(n-1)$ elements to the latter to arrive at an upper bound on the total cost of the first k trees.

We prove $\mathbf{s}(\tau) \succeq \mathbf{r}(\tau)$ by induction on τ, the base case with $\tau = 0$ being trivial. Figure 1 may be helpful in illustrating the structure of this inductive proof. Suppose the claim holds for τ. The probabilities $p_k(\tau)$ are used to determine the forests $F_k(\tau+1)$ and in turn the size vector $\mathbf{s}(\tau+1)$. Consider an intermediate object $\mathbf{s}'(\tau+1)$, the size vector that would be given by incrementing $\mathbf{s}(\tau)$ using the upper-bound values $\bar{p}_k(\tau)$ taken from $\mathbf{s}(\tau)$ by (3.5). Then, $s_i(\tau+1)$ receives the increment if $p_{i-1} \geqslant \alpha > p_i$, and $s'_j(\tau+1)$ receives the increment if $\bar{p}_{j-1} \geqslant \alpha > \bar{p}_j$; hence, from $\bar{p}_{i-1} \geqslant p_{i-1} \geqslant \alpha$ it is immediate that $i \leqslant j$ and thus $\mathbf{s}(\tau+1) \succeq \mathbf{s}'(\tau+1)$.

It suffices then to show that $\mathbf{s}'(\tau+1) \succeq \mathbf{r}(\tau+1)$. These two vectors are obtained respectively from $\mathbf{s}(\tau)$ and $\mathbf{r}(\tau)$, with $\mathbf{s}(\tau) \succeq \mathbf{r}(\tau)$ by the inductive hypothesis, using probability thresholds $\bar{p}_k(\tau) = f(s_k(\tau))$ and $\hat{p}_k(\tau) = f(r_k(\tau))$ respectively, applied to the common random variable α, where $f(s) = s/(n-1)$ (but all that is important is that f is a monotone function of s). Suppose that

$$f(s_{i-1}) \geqslant \alpha > f(s_i) \qquad \text{and} \qquad f(r_{j-1}) \geqslant \alpha > f(r_j), \tag{3.8}$$

so that elements i in $\mathbf{s}$ and j in $\mathbf{r}$ are incremented. If $i \leqslant j$, we are done. (Prefix sums of $\mathbf{s}(\tau)$ dominated those of $\mathbf{r}(\tau)$, and an earlier element is incremented in $\mathbf{s}'(\tau+1)$ than $\mathbf{r}(\tau+1)$, thus prefix sums of $\mathbf{s}'(\tau+1)$ dominate those of $\mathbf{r}(\tau+1)$.) Consider then the case that $i > j$.

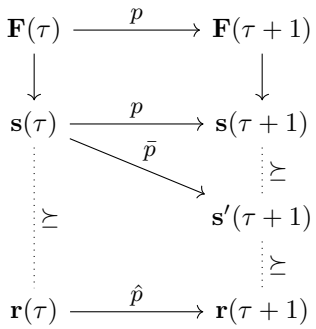


Figure 1 Coupling of the forests' sizes $\mathbf{s}(\tau)$ to a simply analyzable random process $\mathbf{r}(\tau)$, showing the structure of the inductive proof (on τ) that $\mathbf{s}(\tau)$ majorizes $\mathbf{r}(\tau)$.

In both processes the increment falls between indices j and i, so the k-prefix sum inequality continues to hold for $k < j$ and $k \geqslant i$. Thus, for $j \leqslant k < i$,

$$\begin{aligned} \sum_{\ell=1}^{k} s'_\ell(\tau+1) &= \sum_{\ell=1}^{j-1} s_\ell(\tau) + \sum_{\ell=j}^{k} s_\ell(\tau) \\ \sum_{\ell=1}^{k} r_\ell(\tau+1) &= \sum_{\ell=1}^{j-1} r_\ell(\tau) + 1 + \sum_{\ell=j}^{k} r_\ell(\tau). \end{aligned} \tag{3.9}$$

From $j < i$, (3.8), and (3.3) and (3.7) we have that when $j \leqslant \ell \leqslant i-1$,

$$s_\ell \geqslant s_{i-1} \geqslant f^{-1}(\alpha) > r_j \geqslant r_\ell,$$

implying

$$s_\ell \geqslant r_\ell + 1. \tag{3.10}$$

In (3.9), we have $\sum_{\ell=1}^{i-1} s_\ell(\tau) \geqslant \sum_{\ell=1}^{i-1} r_\ell(\tau)$ from the inductive hypothesis that $\mathbf{s}(\tau) \succeq \mathbf{r}(\tau)$, while using (3.10) gives

$$\sum_{\ell=j}^{k} s_\ell(\tau) \geqslant \sum_{\ell=j}^{k} (1 + r_\ell(\tau)) \geqslant 1 + \sum_{\ell=j}^{k} r_\ell(\tau),$$

from which it follows that $\mathbf{s}'(\tau+1) \succeq \mathbf{r}(\tau+1)$, completing the inductive proof that $\mathbf{s}(\tau) \succeq \mathbf{r}(\tau)$.

Having shown that the vector $\mathbf{s}(\tau)$ of component sizes majorizes $\mathbf{r}(\tau)$, it suffices to analyze the latter. Until this point we could have used (3.4) rather than (3.5) to define $\bar{p}_k$, $\hat{p}_k$, and the function f, but now we take advantage of the particularly simple nature of the process governing $\mathbf{r}(\tau)$. Recall that a new edge increments r_i for the first i for which the $U(0,1)$ "coin toss" $\alpha(\tau)$ has $\alpha(\tau) > \hat{p}_i \stackrel{\text{def}}{=} r_i/(n-1)$. Equivalently, consider an array of cells $n-1$ rows high and infinitely many columns wide, generate an "arrival" at a random row or "height" $X(\tau)$ uniform on $1, \ldots, n-1$, and let this arrival occupy the first unoccupied cell i at this height, thus incrementing the occupancy r_i of column i. This is equivalent because if r_i of the $n-1$ cells in column i are occupied, the chance that i is rejected – that $X(\tau)$ falls into this set and thus the arrival moves along to test the next column $i+1$ – is $r_i/(n-1)$, matching (3.6).

Recalling that the cost of an edge arriving at time t is t/n in the original graph problem, the combined cost W_k of the first k spanning trees is $1/n$ times the sum of the arrival times of their $k(n-1)$ edges. The majorization $\sum_{i=1}^{k} s_i(\tau) \geqslant \sum_{i=1}^{k} r_i(\tau)$ means that the ℓ'th arrival

to the first k forests comes no later than the ℓ'th arrival to the first k columns of the cell array. Thus, the cost W_k of the first k trees is at most $1/n$ times the sum of the times of the $k(n-1)$ arrivals to the array's first k columns.

The continuous-time edge arrivals are a Poisson process with intensity $1/n$ on each of the $\binom{n}{2}$ edges, thus intensity $(n-1)/2$ in all; it is at the Poisson arrival times that the discrete time τ is incremented and $X(\tau)$ is generated. Subdivide the "X" process into the $n-1$ possible values that X may take on, so that arrivals at each value (row in the cell array) are a Poisson process of intensity $\lambda = \frac{1}{2}$. The sum of the first k arrival times in a row is the sum of the first k arrival times in its Poisson process. The ith such arrival time is the sum of i exponential random variables, and has expectation i/λ. The expected sum of k arrival times of a line is thus $\binom{k+1}{2}/\lambda = k(k+1)$, and (remembering that cost is time divided by n), the expected total cost of all $n-1$ lines is

$$\frac{n-1}{n}k(k+1),$$

yielding the upper bound in (3.1) and completing the proof of the theorem. ◀

▶ **Corollary 13.** *Let* $\Gamma_k := \sum_{i=1}^{k} \gamma_i$. *Then, for every* $k \geqslant 1$,

$$k^2 \leqslant \Gamma_k = \sum_{i=1}^{k} \gamma_i \leqslant k^2 + k. \tag{3.11}$$

Proof. Immediate from Theorems 12 and 3. ◀

▶ **Example 14.** In particular, Corollary 13 gives $1 \leqslant \gamma_1 \leqslant 2$ and $4 \leqslant \gamma_1 + \gamma_2 \leqslant 6$. In fact, we know that $\gamma_1 = \zeta(3) \doteq 1.2021$ [7] and $\gamma_1 + \gamma_2 > 4.1704$ by [6] and Section 4, see Corollary 20. Numerical estimates suggest $\gamma_1 + \gamma_2 \doteq 4.30$; see Section 5, including Table 1, for various estimates of γ_2. ◀

3.2 Corollaries and conjectures for the kth tree

Turning to individual γ_k instead of their sum Γ_k, we obtain Corollary 7, namely that $2k - 2k^{1/2} < \gamma_k < 2k + 2k^{1/2}$.

Proof of Corollary 7. For the upper bound, we note that obviously $\gamma_1 \leqslant \gamma_2 \leqslant \dots$, and thus, for any $\ell \geqslant 1$, using both the upper and lower bound in (3.11),

$$\begin{aligned}\ell\,\gamma_k &\leqslant \sum_{i=k}^{k+\ell-1} \gamma_i = \Gamma_{k+\ell-1} - \Gamma_{k-1} \leqslant (k+\ell-1)(k+\ell) - (k-1)^2 \\ &= \ell^2 + \ell(2k-1) + k - 1 \end{aligned} \tag{3.12}$$

and hence

$$\gamma_k \leqslant 2k - 1 + \ell + \frac{k-1}{\ell}. \tag{3.13}$$

Choosing $\ell = \lceil \sqrt{k} \rceil$ gives the upper bound in (1.1).

For the lower bound we similarly have, for $1 \leqslant \ell \leqslant k$,

$$\ell\gamma_k \geqslant \Gamma_k - \Gamma_{k-\ell} \geqslant k^2 - (k-\ell)(k-\ell+1) = -\ell^2 - (2k+1)\ell - k \tag{3.14}$$

and hence

$$\gamma_k \geqslant 2k + 1 - \ell - \frac{k}{\ell}. \tag{3.15}$$

Choosing, again, $\ell = \lceil \sqrt{k} \rceil$ gives the lower bound in (1.1). ◀

▶ Remark 15. For a specific k, we can improve (1.1) somewhat by instead using (3.13) and (3.15) with $\ell = \lfloor\sqrt{k}\rfloor$ or $\ell = \lceil\sqrt{k}\rceil$. For example, for $k = 2$, taking $\ell = 1$ yields $2 \leqslant \gamma_2 \leqslant 5$. For $k = 3$, taking $\ell = 2$ yields $3.5 \leqslant \gamma_3 \leqslant 8$. ◀

Besides these rigorous results, taking increments of the left and right-hand sides of (3.11) also suggests the following conjecture.

▶ **Conjecture 16.** *For $k \geqslant 1$, $2k - 1 \leqslant \gamma_k \leqslant 2k$.*

▶ Remark 17. Moreover, if $\gamma_k = 2k + \delta + o(1)$ holds, as conjectured in Conjecture 5, then $\Gamma_k = k^2 + k(\delta + 1) + o(k)$, and thus necessarily $\delta \in [-1, 0]$ as a consequence of Corollary 13. In fact, the numerical estimates described in Section 5, suggest that $\delta = -1$; see Conjecture 6. ◀

3.3 Improved upper bounds

The upper bounds in Theorem 12 and Corollary 13 were proved using the bound (3.5). A stronger, but less explicit, bound can be proved by using instead the sharper (3.4). That is, we consider the random vectors $\mathbf{r}(\tau)$ defined as above but with (3.6) replaced by

$$\hat{p}_k := \binom{r_k + 1}{2} \Big/ \binom{n}{2}. \tag{3.16}$$

As remarked before (3.4), this approximation comes from imagining all edges in each F_k to be in a single component; this overestimates the probability that an arriving edge is rejected from F_k and, as developed in the previous subsection, gives $\mathbf{s}(\tau) \succeq \mathbf{r}(\tau)$ just as when $\hat{p}_k$ was defined by (3.5).

Using for consistency our usual time scaling in which edges arrive at rate $(n-1)/2$, by a standard martingale argument one can show that, for each $k \geqslant 1$,

$$\frac{1}{n} r_k(\lfloor \tfrac{1}{2} n t \rfloor) \xrightarrow{\text{p}} g_k(t), \qquad \text{uniformly for } t \geqslant 0, \tag{3.17}$$

for some continuously differentiable functions $g_k(t)$ satisfying the differential equations, with $g_0(t) := 1$,

$$g_k'(t) = \tfrac{1}{2}\left(g_{k-1}(t)^2 - g_k(t)^2\right), \qquad g_k(0) = 0, \qquad k \geqslant 1. \tag{3.18}$$

Moreover, using $\mathbf{s}(\tau) \succeq \mathbf{r}(\tau)$ and taking limits, it can be shown that

$$\Gamma_k := \sum_{i=1}^{k} \gamma_i \leqslant \frac{1}{2} \int_0^\infty t\big(1 - g_k(t)^2\big)\, \mathrm{d}t. \tag{3.19}$$

We omit the details, but roughly, in time dt, $\frac{1}{2} n\, \mathrm{d}t$ edges arrive, all costing about t/n, and a $g_k(t)^2$ fraction of them pass beyond the first k graphs (to the degree that we are now modeling graphs).

For $k = 1$, (3.18) has the solution $g_1(t) = \tanh(t/2)$, and (3.19) yields the bound $\Gamma_1 = \gamma_1 \leqslant 2 \ln 2 \doteq 1.3863$. This is better than the bound 2 given by (3.11), but still far from precise since $\gamma_1 = \zeta(3) \doteq 1.2021$.

For $k \geqslant 2$ we do not know any exact solution to (3.18), but numerical solution of (3.18) and calculation of (3.19) (see Section 5) suggests that $\Gamma_k < k^2 + 1$. We leave the proof of this as an open problem. If proved, this would be a marked improvement on $\Gamma_k \leqslant k^2 + k$, which was the exact expectation of the random process given by (3.5) (that part of the analysis was tight). In particular, it would establish that $2k - 2 \leqslant \gamma_k \leqslant 2k$.

For $k = 2$, the numerical calculations in Section 5 give $\gamma_1 + \gamma_2 \leqslant 4.5542\ldots$ and thus $\gamma_2 \leqslant 3.3521\ldots$. The same value was also obtained using Maple's numerical differential equation solver, with Maple giving greater precision but the two methods agreeing in the digits shown here.

4 A related problem by Frieze and Johansson

As said in the introduction, Frieze and Johansson [6] recently considered the problem of finding the minimum total cost of k edge-disjoint spanning trees in K_n, for a fixed integer $k \geqslant 2$. (They used random costs with the uniform model; we may consider all three models described in Section 1.1.) We denote this minimum cost by mst_k, following [6]. Trivially,

$$\mathrm{mst}_k \leqslant \sum_{i=1}^{k} w(T_i), \tag{4.1}$$

and as said in the introduction, it is easy to see that strict inequality may hold when $k \geqslant 2$, i.e., that our greedy procedure of choosing $T_1, T_2, \ldots$ successively does not yield the minimum cost set of k disjoint spanning trees.

We assume in this section that $n \geqslant 2k$; then k edge-disjoint spanning trees exist and thus $\mathrm{mst}_k < \infty$.

▶ Remark 18. As observed by Frieze and Johansson [6], the problem is equivalent to finding the minimum cost of a basis in the matroid $\mathcal{M}_k$, defined as the union matroid of k copies of the cycle matroid of K_n. This means that the elements of $\mathcal{M}_k$ are the edges in K_n, and a set of edges is independent in $\mathcal{M}_k$ if and only if it can be written as the union of k forests, see e.g. [20, Chapter 8.3]. (Hence, the bases, i.e., the maximal independent sets, are precisely the unions of k edge-disjoint spanning trees. For the multigraph version in the Poisson model, of course we use instead the union matroid of k copies of the cycle matroid of K_n^∞; we use the same notation $\mathcal{M}_k$.) We write r_k for rank in this matroid. ◀

For $k = 2$, Frieze and Johansson [6] show that

$$\mathbb{E}\,\mathrm{mst}_2 \to \mu_2 \doteq 4.1704. \tag{4.2}$$

This is strictly smaller than our numerical estimates from Table 1 for the total cost of two edge-disjoint spanning trees chosen successively, $\gamma_1 + \gamma_2 \doteq 1.20 + 3.09 > 4.29$; we show this calculation to only two digits as we are confident of this level of precision. This would show that choosing minimum spanning trees one by one is not optimal, even asymptotically, except that our estimates are not rigorous. The following theorem is less precise but establishes rigorously that the values are indeed different. (We rely only on $\sigma_2 < \mu_2$, coming from the estimate of μ_2 above, and our estimate $\sigma_2 \doteq 2.69521$, obtained as the numerical solution to a differential equation; see the full version for details.)

▶ **Theorem 19.** *There exists $\delta > 0$ such that, for any of the three models, w.h.p. $w(T_1) + w(T_2) \geqslant \mathrm{mst}_2 + \delta$.*

This can be restated in the following equivalent form.

▶ **Corollary 20.** $\gamma_1 + \gamma_2 > \mu_2$.

Proof. The equivalence of the statements in Theorem 19 and Corollary 20 is immediate since $w(T_1) \xrightarrow{\mathrm{p}} \gamma_1$ and $w(T_2) \xrightarrow{\mathrm{p}} \gamma_2$ by Theorem 1 or 2 (depending on the choice of model), and $\mathrm{mst}_2 \xrightarrow{\mathrm{p}} \mu_2$ by [6] and justification that this holds in all three models (see the full version). ◀

▶ Remark 21. Numerically, $\gamma_2 > 2.9683$. This is immediate from Corollary 20, the value of μ_2 given by [6], and (by [7]) $\gamma_1 = \zeta(3)$. ◀

The proof of Theorem 19 is based on the fact that many edges are rejected from T_1 and T_2 after time σ_2, but none is rejected from the union matroid until a later time c_3, namely the threshold for appearance of a 3-core in a random graph.

5 Computational results

A variety of computations were performed, all of which will be mentioned here but only one presented in any detail; for the rest see [12].

1. We performed naive simulations, generating edge-weighted random graphs and finding the successive trees.
2. We performed a similar simulation, but instead of introducing edges in order of increasing weight, we simply generate random edges. The details are below.
3. We solved the differential equations (3.18) numerically up to $k = 50$, to get upper bounds on Γ_k as in (3.19). The results suggest that $\Gamma_k < k^2 + 1$ (perhaps $\Gamma_k < k^2 + 0.743$). If proved, this would be a marked improvement on $\Gamma_k \leqslant k^2 + k$, which was the exact expectation of the random process given by (3.5) (that part of the analysis was tight). In particular, it would establish that $2k - 2 \leqslant \gamma_k \leqslant 2k$.
4. Finally, the functions $\rho_k(t)$ can be obtained, recursively on k, through the solution to certain functional fixed-point equations. We solved these numerically, getting results consistent with those in the set of simulations listed as (2) above.

We now detail the set of simulations listed as (2) above, done with reference to the Poisson multigraph model introduced in Section 2.2 and used throughout. We begin with k empty graphs of order n. At each step we introduce a random edge e and, in the first graph G_i for which e does not lie within a component, we merge the two components given by its endpoints. (If this does not occur within the k graphs under consideration, we do nothing, just move on to the next edge.) For each graph we simulate only the components (i.e., the sets of vertices comprised by each component); there is no need for any more detailed structure. The edge arrivals should be regarded as occurring as a Poisson process of intensity $(n-1)/2$ but instead we simply treat them as arriving at times $2/n$, $4/n$, etc.

Figure 2 depicts the result of a single such simulation with $n = 1\,000\,000$, showing for each k from 1 to 5 the size of the largest component of G_k (as a fraction of n) against time. Similar experiments with multiple simulations and larger values of n support Conjecture 6 that $\gamma_k = 2k - 1 + o(1)$. The largest experiment's results are shown in part in Table 1; its support for the conjecture continues through $k = 29$, the last value for which it gives good data.

Table 1 Estimates of $\gamma_1, \ldots, \gamma_9$ from 10 simulations each with $n = 10\,000\,000$, through time $t = 40$.

10 simulations each with $n = 10\,000\,000$									
	γ_1	γ_2	γ_4	γ_4	γ_5	γ_6	γ_7	γ_8	γ_9
mean	1.2020	3.0921	5.0482	7.0253	9.0169	11.0091	13.0067	15.0035	17.0039
std err	0.0002	0.0003	0.0005	0.0008	0.0010	0.0012	0.0016	0.0010	0.0015

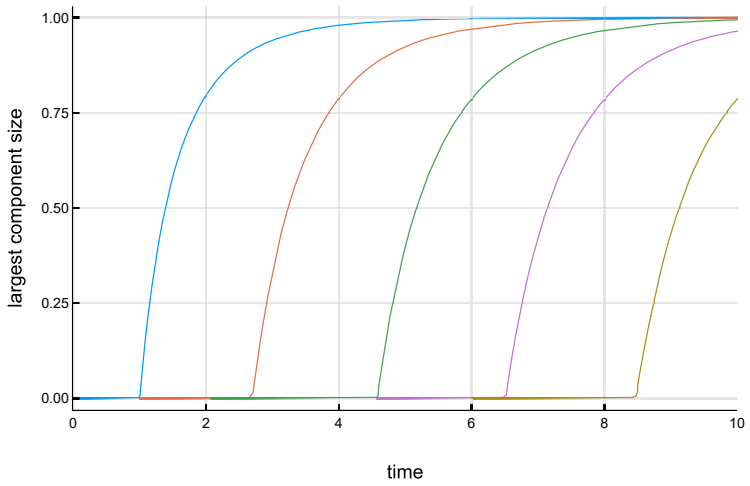


Figure 2 Largest component sizes, as a fraction of n, for graphs $G_1, \dots, G_5$, based on a single simulation with $n = 1\,000\,000$.

6 Open questions

We would be delighted to confirm the various conjectures above, in particular Conjectures 4–6, and to get a better understanding of (and ideally a closed form for) ρ_∞ (provided it exists).

It is also of natural interest to ask this kth-minimum question for structures other than spanning trees. Subsequent to this work, the length X_k of the kth shortest s–t path in a complete graph with random edge weights has been studied by Mezei, Gerke and Sorkin [8]. They show that $X_k/(2k/n + \ln n/n) \xrightarrow{\mathrm{p}} 1$ for all k from 1 to $n-1$. In particular, the first few paths all cost nearly identical amounts, quite different from the situation for successive MSTs.

The "random assignment problem" is to determine the cost of a minimum-cost perfect matching in a complete bipartite graph with random edge weights. A great deal is known about it, by a variety of methods; for one relatively recent work, with references to others, see Wästlund [19]. It would be interesting to understand the kth cheapest matching.

It could also be interesting to consider other variants of all these questions. Frieze and Johansson [6] considered the k disjoint structures which together have the smallest possible total cost, where we consider disjoint structures generated successively. In either case, instead of asking for disjoint structures, we could require structures which are merely distinct, or perhaps which differ in some adversarially specified elements.

References

1 Aaron Archer and Éva Tardos. Frugal path mechanisms. *ACM Transactions on Algorithms*, 3(1):3:1–3:22, 2007.

2 Béla Bollobás and Alan M. Frieze. On matchings and Hamiltonian cycles in random graphs. In Michał Karoński and Andrzej Ruciński, editors, *Random Graphs '83 (Poznań, 1983), Ann. Discrete Math.* ***28***, volume 118 of *North-Holland Mathematics Studies*, pages 23–46. North-Holland, 1985.

3 Béla Bollobás, Svante Janson, and Oliver Riordan. The phase transition in inhomogeneous random graphs. *Random Struct. Alg.*, 31(1):3–122, 2007.

4 Prasad Chebolu, Alan Frieze, Páll Melsted, and Gregory B Sorkin. Average-case analyses of Vickrey costs. In *Proceedings of APPROX / RANDOM 2009*, volume 5687 of *Lecture Notes in Comput. Sci.*, pages 434–447. Springer, Berlin, Heidelberg, 2009.

5 Colin Cooper, Alan Frieze, Nate Ince, Svante Janson, and Joel Spencer. On the length of a random minimum spanning tree. *Combin. Probab. Comput.*, 25(1):89–107, 2016.

6 Alan Frieze and Tony Johansson. On edge disjoint spanning trees in a randomly weighted complete graph. *Combin. Probab. Comput.*, 27(2):228–244, 2018.

7 Alan M. Frieze. On the value of a random minimum spanning tree problem. *Discrete Applied Mathematics*, 10:47–65, 1985.

8 Stefanie Gerke, Balázs Mezei, and Gregory B. Sorkin. Successive shortest paths, 2019. Manuscript in preparation.

9 Svante Janson. The minimal spanning tree in a complete graph and a functional limit theorem for trees in a random graph. *Random Struct. Alg.*, 7:337–355, 1995.

10 Svante Janson, Donald E. Knuth, Tomasz Łuczak, and Boris Pittel. The birth of the giant component. *Random Struct. Alg.*, 4(3):231–358, 1993.

11 Svante Janson and Gregory B. Sorkin. VCG auction mechanism cost expectations and variances, 2013. `arXiv:1310.1777`.

12 Svante Janson and Gregory B Sorkin. Successive minimum spanning trees, 2019. `arXiv:1310.1777`.

13 Svante Janson and Johan Wästlund. Addendum to "The minimal spanning tree in a complete graph and a functional limit theorem for trees in a random graph". *Random Struct. Alg.*, 28(4):511–512, 2006.

14 Anna R. Karlin, David Kempe, and Tami Tamir. Beyond VCG: Frugality of truthful mechanisms. In *Proceedings of 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 615–624. IEEE, 2005.

15 Joseph B. Kruskal, Jr. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.*, 7:48–50, 1956.

16 Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory.* Cambridge Univ. Press, New York, 2007.

17 Kunal Talwar. The price of truth: frugality in truthful mechanisms. In *Proceedings of STACS 2003, Lecture Notes in Comput. Sci.*, volume 2607, pages 608–619. Springer, 2003.

18 Johan Wästlund. Evaluation of Janson's constant for the variance in the random minimum spanning tree problem, 2005. *Linköping Studies in Mathematics* No. 7.

19 Johan Wästlund. An easy proof of the $\zeta(2)$ limit in the random assignment problem. *Elect. Comm. in Probab.*, 14:261—269, 2009.

20 Dominic J. A. Welsh. *Matroid theory.* Academic Press, New York, 1976. Reprinted, Dover, Mineola, New York, 2010.

Simple Analysis of Sparse, Sign-Consistent JL

Meena Jagadeesan
Harvard University, Cambridge, Massachusetts, USA
mjagadeesan@college.harvard.edu

Abstract

Allen-Zhu, Gelashvili, Micali, and Shavit construct a sparse, sign-consistent Johnson-Lindenstrauss distribution, and prove that this distribution yields an essentially optimal dimension for the correct choice of sparsity. However, their analysis of the upper bound on the dimension and sparsity requires a complicated combinatorial graph-based argument similar to Kane and Nelson's analysis of sparse JL. We present a simple, combinatorics-free analysis of sparse, sign-consistent JL that yields the same dimension and sparsity upper bounds as the original analysis. Our analysis also yields dimension/sparsity tradeoffs, which were not previously known.

As with previous proofs in this area, our analysis is based on applying Markov's inequality to the pth moment of an error term that can be expressed as a quadratic form of Rademacher variables. Interestingly, we show that, unlike in previous work in the area, the traditionally used Hanson-Wright bound is *not* strong enough to yield our desired result. Indeed, although the Hanson-Wright bound is known to be optimal for gaussian degree-2 chaos, it was already shown to be suboptimal for Rademachers. Surprisingly, we are able to show a simple moment bound for quadratic forms of Rademachers that is sufficiently tight to achieve our desired result, which given the ubiquity of moment and tail bounds in theoretical computer science, is likely to be of broader interest.

2012 ACM Subject Classification Theory of computation → Random projections and metric embeddings

Keywords and phrases Dimensionality reduction, Random projections, Johnson-Lindenstrauss distribution, Hanson-Wright bound, Neuroscience-based constraints

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.61

Category RANDOM

Related Version A full version of the paper is available at `https://arxiv.org/abs/1708.02966`.

Funding *Meena Jagadeesan*: Supported in part by a Harvard PRISE fellowship, Herchel-Smith Fellowship, and an REU supplement to NSF IIS-1447471.

Acknowledgements I would like to thank Prof. Jelani Nelson for advising this project.

1 Introduction

In many modern algorithms that process high dimensional data, it is beneficial to preprocess the data through a dimensionality reduction scheme that preserves the geometry of the data. Dimensionality reduction schemes have been applied in streaming algorithms [22] as well as algorithms for numerical linear algebra [29], feature hashing [27], graph sparsification [25], and many other areas. The geometry-preserving objective can be expressed mathematically as follows. The goal is to construct a probability distribution $\mathcal{A}$ over $m \times n$ real matrices that satisfies the following condition for any $x \in \mathbb{R}^n$:

$$\mathbb{P}_{A \in \mathcal{A}}[(1-\epsilon)||x||_2 \leq ||Ax||_2 \leq (1+\epsilon)||x||_2] > 1-\delta. \tag{1}$$

An upper bound on the dimension m achievable by a probability distribution $\mathcal{A}$ that satisfies (1) is given in the following lemma, which is a central result in the area of dimensionality reduction:


© Meena Jagadeesan;
licensed under Creative Commons License CC-BY



▶ **Lemma 1** (Johnson-Lindenstrauss [15]). *For any positive integer n and parameters $0 < \epsilon, \delta < 1$, there exists a probability distribution $\mathcal{A}$ over $m \times n$ real matrices with $m = \Theta(\epsilon^{-2} \log(1/\delta))$ that satisfies* (1).

The optimality of the dimension m achieved by Lemma 1 was recently proven in [16, 14].

For many applications of dimensionality reduction schemes, it can be useful to consider probability distributions over sparse matrices in order to speed up the projection time. Here, sparsity refers to the constraint that there are a small number of nonzero entries in each column. In this context, Kane and Nelson [18] constructed a sparse JL distribution, improving the work of Achlioptas [1] and Dasgupta et al. [6], and proved the following:

▶ **Theorem 2** (Sparse JL [18]). *For any positive integer n and $0 < \epsilon, \delta < 1$, there exists a probability distribution $\mathcal{A}$ over $m \times n$ real matrices with $m = \Theta(\epsilon^{-2} \log(1/\delta))$ and sparsity $s = \Theta(\epsilon^{-1} \log(1/\delta))$ that satisfies* (1).

Notice that this probability distribution, even with its sparsity guarantee, achieves the same dimension as Lemma 1. The proof of Theorem 2 presented in [18] involved complicated combinatorics; however, Cohen, Jayram, and Nelson [4] recently constructed two simple, combinatorics-free proofs of this result. The first approach, which is most relevant to the approach taken in this paper, used the Hanson-Wright bound on moments of quadratic forms. An analysis similar to the second approach can be recovered by specializing the analysis of Cohen [3] for sparse oblivious subspace embeddings to the case of "1-dimensional subspaces." In fact, though this recovered analysis is more complex, it has the advantage of yielding dimension-sparsity tradeoffs that were not produced through any of the previous approaches: for $B \geq e$, the sparsity s can be set to $\Theta(\epsilon^{-1} \log_B(1/\delta))$ if m is set to $\Theta(B\epsilon^{-2} \log(1/\delta))$, enabling a $\log B$ factor reduction in sparsity at the expense of a B factor gain in dimension.

JL with sign-consistency constraints

Neuroscience-based constraints give rise to the additional condition of sign-consistency on the matrices in the probability distribution. Sign-consistency refers to the constraint that the nonzero entries of each column are either all positive or all negative. The relevance of dimensionality reduction schemes in neuroscience is described in a survey by Ganguli and Sompolinsky [9]. In convergent pathways in the brain, information stored in a massive number of neurons is compressed into a small number of neurons, and nonetheless the ability to perform the relevant computations is preserved. Modeling this information compression scheme requires a hypothesis regarding what properties of the original information must be accurately transmitted to the receiving neurons. A plausible minimum requirement is that convergent pathways preserve the similarity structure of neuronal representations at the source area.[1]

It remains to select the appropriate mathematical measure of similarity. The candidate similarity measure considered in [9] is vector inner product, which conveniently gives rise to a model based on the JL distribution.[2] Suppose there are n "input" neurons at a source area and m "output" neurons at a target area. In this framework, the information at the

[1] This requirement is based on the experimental evidence that semantically similar objects in higher perceptual or association areas in the brain elicit similar neural activity patterns [19] and on the hypothesis that the similarity structure of the neural code is the basis of our ability to categorize objects and generalize responses to new objects [24].

[2] It is not difficult to see that for vectors x and y in the ℓ_2 unit ball, a $(1+\varepsilon)$-approximation of $\|x\|_2$, $\|y\|_2$, and $\|x-y\|_2$ implies an additive error $\Theta(\epsilon)$ approximation of the inner product $\langle x, y \rangle$.

input neurons is represented as a vector in $\mathbb{R}^n$, the synaptic connections to output neurons are represented as a $m \times n$ matrix (with (i,j)th entry corresponding to the strength of the connection between input neuron j and output neuron i), and the information received by the output neurons is represented as a vector in $\mathbb{R}^m$. The similarity measure between two vectors v, w of neural information being taken to be $\langle v, w \rangle$ motivates modeling a synaptic connectivity matrix as a random $m \times n$ matrix drawn from a probability distribution that satisfies (1). Certain constraints on synaptic connectivity matrices arise from the biological limitations of neurons: the matrices must be *sparse* since a neuron is only connected to a small number (e.g. a few thousand) of postsynaptic neurons and *sign-consistent* since a neuron is usually purely excitatory or purely inhibitory.

This biological setting motivates the mathematical question: what is the optimal dimension and sparsity that can be achieved by a probability distribution over sparse, sign-consistent matrices that satisfies (1)? Allen-Zhu, Gelashvili, Micali, and Shavit [2] constructed a sparse, sign-consistent JL distribution[3] and proved the following:

▶ **Theorem 3** (Sparse, sign-consistent JL [2]). *For every $\varepsilon > 0$, and $0 < \delta < 1/e$, there exists a probability distribution $\mathcal{A}$ over $m \times n$ real, sign-consistent matrices with $m = \Theta(\varepsilon^{-2} \log^2(1/\delta))$ and sparsity $s = \Theta(\varepsilon^{-1} \log(1/\delta))$ that satisfies (1).*

In [2], it was also proven that the additional $\log(1/\delta)$ factor on m is essentially necessary: namely, any distribution over sign-consistent matrices satisfying (1) requires $m = \tilde{\Omega}(\varepsilon^{-2} \log(1/\delta) \min(\log(1/\delta), \log n))$. Thus, the dimension in Theorem 3 is essentially optimal. However, in order to achieve this upper bound on m, the proof presented in [2] involved complicated combinatorics even more delicate than in the analysis of sparse JL in [18].

We present a simpler, combinatorics-free proof of Theorem 3. Our analysis also yields dimension/sparsity tradeoffs, which were not previously known.[4] We prove the following:

▶ **Theorem 4.** *For every $\varepsilon > 0$, $0 < \delta < 1$, and $e \leq B \leq \frac{1}{\delta}$, there exists a probability distribution $\mathcal{A}$ over $m \times n$ real, sign-consistent matrices with $m = \Theta(B\varepsilon^{-2} \log_B^2(1/\delta))$ and sparsity $s = \Theta(\varepsilon^{-1} \log_B(1/\delta))$ that satisfies (1).*

Notice Theorem 3 is recovered if $B = e$. For larger B values, Theorem 4 enables a $\log B$ factor reduction in sparsity at the cost of a $B/\log^2 B$ factor gain in dimension.

To contextualize our tradeoff in Theorem 4, recall that the upper bounds on sparse (non-sign-consistent) JL dimension-sparsity tradeoffs by Cohen [3] take a similar form, allowing a $\log B$ factor reduction in s for a B factor gain in m. Moreover, in a recent follow-up work [13], we show lower bounds that indicate that the standard choice of sparse JL construction requires an exponential factor gain in dimension for a given reduction in sparsity, demonstrating that Cohen's dimension-sparsity tradeoffs are essentially tight.[5] Due to the structural similarity between sparse JL and sparse, sign-consistent JL, we believe this provides indication that our tradeoffs in Theorem 4 could be tight for this construction in many regimes.[6]

[3] Related mathematical work includes, in addition to sparse JL [18], a construction of a dense, sign-consistent JL distribution [23, 10].

[4] In Appendix A, we point out the limiting lemma in the combinatorial analysis in [2], which prevents dimension-sparsity tradeoffs from being attainable through this approach, due to an assumption that is implicitly used in the analysis. For sparse JL, it is similarly not known how to obtain these tradeoffs via the combinatorial approach of [18].

[5] More specifically, it follows from [3] and [13] that m is exactly $\min(\text{poly}(B)\epsilon^{-2}\log(1/\delta), 2\epsilon^{-2}/\delta)$ for the standard choice of sparse JL construction (uniformly choosing s nonzero entries per column).

[6] An interesting direction for future work could be to build upon the ideas in the follow-up work [13] to show lower bounds on the dimension-sparsity tradeoffs for this sparse, sign-consistent JL construction.

Proof Techniques

As in [2, 18, 4], our analysis is based on applying Markov's inequality to the pth moment of an error term. Like in the first combinatorics-free analysis of sparse JL in [4], we express this error term as a quadratic form of Rademachers (uniform ± 1 random variables), and our analysis then boils down to analyzing the moments of this quadratic form. While the analysis in [4] achieves the optimal dimension for sparse JL using an upper bound on the moments of quadratic forms of subgaussians due to Hanson and Wright [11], we give a counterexample in Section 3.2 that shows that the Hanson-Wright bound is too loose in the sign-consistent setting to result in the optimal dimension. Since the Hanson-Wright bound is tight for quadratic forms of gaussians, we thus require a separate treatment of quadratic forms of Rademachers.

We construct a simple bound on moments of quadratic forms of Rademachers that, unlike the Hanson-Wright bound, is sufficiently tight in our setting to prove Theorem 4. Our bound borrows some of the ideas from Latała's tight bound on the moments of quadratic forms of Rademachers [21]. Although our bound is much weaker than the bound in [21] in the general case, it has the advantage of providing a greater degree of simplicity by consisting of easier-to-analyze terms; this simplicity is critical since our quadratic form coefficients are themselves random variables. The crux is that while the bound in [21] is focused on obtaining tight estimates for quadratic forms with scalar coefficients, our bound is much more tractable for quadratic forms with random variable coefficients. As a result, our bound enables a simple proof of Theorem 4, while retaining the necessary precision to recover the optimal dimension.

We build upon these ideas in our recent follow-up work [13], where the Hanson-Wright bound also turns out to be too loose. The work studies sparse JL performance in feature hashing and considers the restricted set of vectors with small ℓ_∞-to-ℓ_2 norm ratio, continuing a line of work [27, 6, 17, 5, 18, 7]. The main result is a tight tradeoff between ℓ_∞-to-ℓ_2 norm ratio and ϵ, δ, s, and m, and the lower bounds on dimension-sparsity tradeoffs mentioned before are shown as a corollary. Similar to this work, the proof boils down to a tight bound on pth moment of an error term, and it also turns out that the Hanson-Wright bound is too loose here. The work solves this issue by building upon ideas from this work, utilizing a separate treatment of Rademachers that is tractable for random variable coefficients. While the analysis in [13] does not use the exact quadratic form bound presented here, it uses intuition and generalizations of the moment bounding techniques presented in this work.

1.1 Notation

The main building blocks for our expressions are the following two types of random variables: Rademacher variables, which are uniform ± 1 random variables, and Bernoulli random variables, which have support $\{0, 1\}$. For any random variable X and value $p \geq 1$, we use the notation $\|X\|_p$ to denote the p-norm $(\mathbb{E}[|X|^p])^{1/p}$, where $\mathbb{E}$ denotes the expectation. Similarly, for any random variable X and value $p \geq 1$ and any event E, we use the notation $\|X \mid E\|_p$ to denote the conditional p-norm $(\mathbb{E}[|X|^p \mid E])^{1/p}$, which is equivalent to the p-norm of the random variable $(X \mid E)$. We use the following notation to discuss certain asymptotics: given two scalar quantities Q_1 and Q_2 that are functions of some parameters, we use the notation $Q_1 \simeq Q_2$ to denote that there exist positive universal constants $C_1 \leq C_2$ such that $C_1 Q_2 \leq Q_1 \leq C_2 Q_2$, and we use the notation $Q_1 \lesssim Q_2$ to denote that there exists a positive universal constant C such that $Q_1 \leq C Q_2$.

1.2 A digression on Rademachers versus gaussians

The concept that drives our moment bound can be illustrated in the linear form setting. Suppose $\sigma_1, \sigma_2 \ldots, \sigma_n$ are i.i.d Rademachers, $x = [x_1, \ldots, x_n]$ is a vector in $\mathbb{R}^n$ such that $|x_1| \geq |x_2| \geq \ldots \geq |x_n|$, and $2 \leq p \leq n$. The Khintchine inequality, which is tight for linear forms of gaussians, yields the ℓ_2-norm bound $\|\sum_{i=1}^n \sigma_i x_i\|_p \lesssim \sqrt{p}\,\|x\|_2$. However, this bound *cannot* be a tight bound on $\|\sum_{i=1}^n \sigma_i x_i\|_p$ for the following reason: As $p \to \infty$, the quantity $\sqrt{p}\,\|x\|_2$ goes to infinity, while for any $p \geq 1$, the quantity $\|\sum_{i=1}^n \sigma_i x_i\|_p$ is bounded by $\|x\|_1$. Surprisingly, a result due to Hitczenko [12] indicates that the tight bound is actually the following combination of the ℓ_2 and ℓ_1 norm bounds:

$$\left\|\sum_{i=1}^n \sigma_i x_i\right\|_p \simeq \sum_{i=1}^p |x_i| + \sqrt{p}\sqrt{\sum_{i>p} x_i^2}.$$

In this bound, the "big" terms (i.e. terms involving x_1, x_2, ..., x_p) are handled with an ℓ_1-norm bound, while the remaining terms are approximated as gaussians and bounded with an ℓ_2-norm bound.

A similar complication arises when the Hanson-Wright bound on quadratic forms of subgaussians is applied to Rademachers. Let σ be a d-dimensional vector of independent Rademachers, and let $A = (a_{k,l})$ be a symmetric $d \times d$ matrix with zero diagonal. The Hanson-Wright bound [11], which is tight for gaussians, states for any $p \geq 1$,

$$\left\|\sigma^T A \sigma\right\|_p \lesssim \sqrt{p}\sqrt{\sum_{k=1}^d \sum_{l=1}^d a_{k,l}^2} + p\left(\sup_{\|y\|_2=1} |y^T A y|\right).$$

Similar to the linear form setting, this bound *can't* be a tight bound on $\left\|\sigma^T A \sigma\right\|_p$ for the following reason: As $p \to \infty$, the quantity $\sqrt{p}\sqrt{\sum_{k=1}^d \sum_{l=1}^d a_{k,l}^2}$ goes to ∞, while for any $p \geq 1$, the quantity $\left\|\sigma^T A \sigma\right\|_p$ is bounded by the entrywise ℓ_1-norm $\sum_{k=1}^d \sum_{l=1}^d |a_{k,l}|$.

Our quadratic form bound is based on a degree-2 analog of Hitczenko's observation. We analogously handle the "big" terms with an ℓ_1-norm bound and bound the remaining terms by approximating some of the Rademachers by gaussians. From this, we obtain a combination of ℓ_2 and ℓ_1 norm bounds, similar to the linear form setting. Our simple bound has the surprising feature that it yields tighter guarantees than the Hanson-Wright bound yields for our error term. While our bound is weaker than Latała's tight bound [21] on the moments of quadratic forms of Rademachers in the general case, it provides a greater degree of simplicity: our bound avoids an operator-norm-like term in Latała's bound that is especially difficult to analyze when A is a random matrix, as is the case in this setting. Moreover, our bound still retains the necessary precision to recover the optimal dimension for sparse, sign-consistent JL.

Although our final analysis follows a style that this is perhaps less well-known within the TCS community, in the end, it is quite simple, relying only on our quadratic form bound coupled with a few standard tricks such as repeated use of triangle inequalities on $||\cdot||_p$ norms and standard moment bounds involving the binomial distribution. For this reason, we believe that it is likely to be of interest in other theoretical computer science settings involving moments or tail bounds of Rademacher forms.

1.3 Outline for the rest of the paper

In Section 2, we describe the construction and analysis of [2] for sparse, sign-consistent JL. In Section 3, we present the combinatorics-free approach in [4] for sparse JL that uses the Hanson-Wright bound, and we discuss why this approach does not yield the optimal dimension in the sign-consistent setting. In Section 4, we derive our bound on the moments of quadratic forms of Rademachers and use this bound to construct a combinatorics-free proof of Theorem 4.

2 Existing Analysis for Sparse, Sign-Consistent JL

In Section 2.1, we describe how to construct the probability distribution of sparse, sign-consistent matrices analyzed in Theorem 3. In Section 2.2, we briefly describe the combinatorial proof of Theorem 3 presented in [2].

2.1 Construction of Sparse, Sign-Consistent JL

The entries of a matrix $A \in \mathcal{A}$ are generated as follows.[7] Let $A_{i,j} = \eta_{i,j}\sigma_j/\sqrt{s}$ where $\{\sigma_i\}_{i\in[n]}$ and $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ are defined as follows:

- The families $\{\sigma_i\}_{i\in[n]}$ and $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ are independent from each other.
- The variables $\{\sigma_i\}_{i\in[n]}$ are i.i.d Rademachers.
- The variables $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ are identically distributed Bernoulli random variables with expectation s/m.
- The $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ are independent across columns but not independent within each column. For every column $1 \le i \le n$, it holds that $\sum_{r=1}^{m} \eta_{r,i} = s$. For every subset $S \subseteq [m]$ and every column $1 \le i \le n$, it holds that $\mathbb{E}\left[\prod_{r\in S} \eta_{r,i}\right] \le \prod_{r\in S} \mathbb{E}[\eta_{r,i}]$. (One common definition of $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ that satisfies these conditions is the distribution defined by uniformly choosing exactly s of these variables per column to be a 1.)

For every $x \in \mathbb{R}^n$ such that $\|x\|_2 = 1$, we need to analyze an error term, which for this construction is the following random variable:

$$Z := \|Ax\|_2^2 - 1 = \frac{1}{s}\sum_{i\neq j}\sum_{r=1}^{m} \eta_{r,i}\eta_{r,j}\sigma_i\sigma_j x_i x_j.$$

Proving that $\mathcal{A}$ satisfies (1) boils down to proving that $\mathbb{P}_{\eta,\sigma}[|Z| > \epsilon] < \delta$. The main technique to prove this tail bound is the moment method. Bounding a large moment of Z is useful since it follows from Markov's inequality that

$$\mathbb{P}_{\eta,\sigma}[|Z| > \epsilon] = \mathbb{P}_{\eta,\sigma}[|Z|^p > \epsilon^p] < \frac{\mathbb{E}[|Z|^p]}{\epsilon^p}.$$

The usual approach, used in the analyses in [2, 18, 4] as well as in our analysis, is to take $p = \Theta(\log(1/\delta))$ to be an even integer and analyze the p-norm $\|Z\|_p$ of the error term.

2.2 Discussion of the combinatorial analysis of [2]

In the analysis in [2], a complicated combinatorial argument was used to prove the following lemma, from which Theorem 3 follows:

▶ **Lemma 5** ([2]). *If $s^2 \le m$ and $p < s$, then $\|Z\|_p \lesssim \frac{p}{s}$.*

[7] See the appendix of the full version of the paper for a formal construction of the probability space.

The argument in [2] to prove Lemma 5 was based on expanding $\mathbb{E}[Z^p]$ into a polynomial with $\approx n^{2p}$ terms, establishing a correspondence between the monomials and the multigraphs, and then doing combinatorics to analyze the resulting sum. The approach of mapping monomials to graphs is commonly used in analyzing the eigenvalue spectrum of random matrices [28, 8] and was also used in [18] to analyze sparse JL. The analysis in [2] borrowed some methods from the analysis in [18]; however, the additional correlations between the Rademachers imposed by sign-consistency forced the analysis in [2] to require more delicate manipulations at several stages of the computation.

The expression to be analyzed was $s^p\mathbb{E}[Z^p]$, which was written as:

$$\sum_{i_1,\ldots,i_p,j_1,\ldots,j_p\in[n],i_1\neq j_1,\ldots,i_p\neq j_p}\left(\prod_{u=1}^{p}x_{i_u}x_{j_u}\right)\left(\mathbb{E}_\sigma\prod_{u=1}^{p}\sigma_{i_u}\sigma_{j_u}\right)\left(\mathbb{E}_\eta\prod_{u=1}^{t}\sum_{r=1}^{m}\eta_{r,i_u}\eta_{r,j_u}\right).$$

After layers of computation, it was shown that

$$s^p\mathbb{E}[Z^p]\leq e^p\sum_{v=2}^{p}\sum_{G\in\mathcal{G}_{v,p}}\left((1/p^p)\prod_{q=1}^{v}\sqrt{d_q}^{d_q}\right)\sum_{r_1,\ldots,r_p\in[m]}\prod_{i=1}^{w}(s/m)^{v_i}$$

where $\mathcal{G}_{v,p}$ is a set of directed multigraphs with v labeled vertices and t labeled edges, where d_q is the total degree of vertex $q\in[v]$ in a graph $\mathcal{G}_{v,p}$, and where w and $v_1,\ldots,v_w$ are defined by G and the edge colorings $r_1,\ldots,r_t$. The problem then boiled down to carefully enumerating the graphs in $\mathcal{G}_{v,p}$ in six stages and analyzing the resulting expression.

3 Discussion of Combinatorics-Free Approaches

The main ingredient of the first combinatorics-free approach for sparse JL presented in [4] is the Hanson-Wright bound on the moments of quadratic forms of subgaussians. In Section 3.1, we discuss the approach in [4]. In Section 3.2, we discuss why this approach, if applied to sparse, sign-consistent JL, fails to yield the optimal dimension.

3.1 Hanson-Wright approach for sparse JL in [4]

The relevant random variable for sparse JL is

$$Z'=||Ax||^2-1=\frac{1}{s}\sum_{r=1}^{m}\sum_{i\neq j}\eta_{r,i}\eta_{r,j}\sigma_{r,i}\sigma_{r,j}x_ix_j$$

where the n independent Rademachers $\{\sigma_i\}_{i\in[n]}$ from the sign-consistent case are replaced by the mn independent Rademachers $\{\sigma_{r,i}\}_{i\in[n],r\in[m]}$. The main idea in [4] was to view Z' as a quadratic form $\frac{1}{s}\sigma^TA\sigma$. Here, σ is a mn-dimensional vector of independent Rademachers and $A=(A_{k,l})$ is a symmetric, zero diagonal, block diagonal $mn\times mn$ matrix with m blocks of size $n\times n$, where the (i,j)th entry (for $i\neq j$) of the rth block is $\eta_{r,i}\eta_{r,j}x_ix_j$. The quantity $\left\|\sigma^TA\sigma\right\|_p$ was analyzed using the Hanson-Wright bound. In order to bound $\left\|\sigma^TA\sigma\right\|_p$, since A is a random matrix whose entries depend on the η values, an expectation had to be taken over η in the expression given by the Hanson-Wright bound. This resulted in the following:

$$\left\|\sigma^TA\sigma\right\|_p\lesssim\left\|\sqrt{p}\sqrt{\sum_{k=1}^{mn}\sum_{l=1}^{mn}A_{k,l}^2}+p\sup_{\|y\|_2=1}|y^TAy|\right\|_p. \tag{2}$$

The remainder of the analysis boiled down to bounding the RHS of (2), and it successfully recovered Theorem 2.

3.2 Failure of the Hanson-Wright approach for sparse, sign-consistent JL

The Hanson-Wright-based approach for sparse JL in [4] cannot be applied to the sign-consistent case to obtain a tight bound on $\|Z\|_p$. The loss arises from the fact that while the Hanson-Wright bound is tight for quadratic forms of gaussians, it is not guaranteed to be tight for quadratic forms of Rademachers. As discussed in Section 1.2, when $p \to \infty$, the Hanson-Wright bound goes to ∞, while $||\sigma^T A\sigma||_p$ can be bounded by the entrywise ℓ_1 norm of the matrix A. Although approximating the error term Rademachers by gaussians happened to be sufficiently tight for sparse JL, this loss results in a suboptimal dimension for sparse, sign-consistent JL.[8] We give a counterexample, i.e. a vector x, that shows that the Hanson-Wright bound is too loose to give the optimal dimension (when $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ are defined by uniformly choosing exactly s of the variables per column to be a 1). We present the details in Appendix E.

4 Simple Proof of Theorem 4

The main ingredient in our proof of Theorem 4 is the following bound on $\|Z\|_p$:

▶ **Lemma 6.** *Let $B = m/s^2$. If $p \geq 2$, then*

$$\|Z\|_p \lesssim \begin{cases} \frac{p}{s \log B}, & \text{if } B \geq e \\ \frac{p}{sB} & \text{if } B < e. \end{cases}$$

We will later show that Theorem 4 follows from Lemma 6 via Markov's inequality.

In order to analyze $\|Z\|_p$, we view Z as a quadratic form $\frac{1}{s}\sigma^T A\sigma$, where the vector σ is an n-dimensional vector of independent Rademachers, and $A = (a_{i,j})$ is a symmetric, zero-diagonal $n \times n$ matrix where the (i,j)th entry (for $i \neq j$) is $x_i x_j \sum_{r=1}^m \eta_{r,i}\eta_{r,j}$. Since Z is symmetric in $x_1, \ldots, x_n$, we can assume WLOG that $|x_1| \geq |x_2| \geq \ldots \geq |x_n|$. For convenience, we define, like in [4],

$$Q_{i,j} := \sum_{r=1}^{m} \eta_{r,i}\eta_{r,j} \tag{3}$$

to be the number of collisions between the nonzero entries of the ith column and the nonzero entries of the jth column. Now, the (i,j)th entry of A (for $i \neq j$) can be written as $Q_{i,j}x_i x_j$.

As discussed in Section 3.2, we cannot apply the Hanson-Wright bound to tightly analyze $\|Z\|_p$ and thus require a separate treatment of Rademachers. We derive the following moment bound on quadratic forms of Rademachers[9] that yields tighter guarantees than the Hanson-Wright bound yields for $\|Z\|_p$:

[8] The difference results from the correlations between the signs resulting in more "tightly packed" coefficients in the error term quadratic form in the sign-consistent case.

[9] As mentioned before, Latała [21] provides a tight bound on the moments of $\sigma^T A\sigma$ (and on the moments of more general quadratic forms). However, his bound consists of terms that are difficult to analyze when the quadratic form coefficients are random variables. Moreover, his proof is quite complicated, though the bound can be used in a black box to generate a much messier solution (by unravelling some of his proof to avoid the operator-norm-like term).

▶ **Lemma 7.** *If $A = (a_{i,j})$ is a symmetric square $n \times n$ matrix with zero diagonal, $\{\sigma_i\}_{i\in[n]}$ is a set of independent Rademachers, and $q \geq 1$, then*

$$\left\|\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}\sigma_i\sigma_j\right\|_q \lesssim \left(\sum_{i=1}^{\min(q,n)}\sum_{j=1}^{\min(q,n)} |a_{i,j}|\right) + \sqrt{q}\sqrt{\sum_{i=1}^{n}\left\|\sum_{j>q} a_{i,j}\sigma_j\right\|_q^2}.$$

Observe that our bound avoids the weakness of the Hanson-Wright bound in the limit as $q \to \infty$. As discussed in Section 1.2, $\left\|\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}\sigma_i\sigma_j\right\|_q$ can be bounded by the entrywise ℓ_1-norm bound $\sum_{i=1}^{n}\sum_{j=1}^{n}|a_{i,j}|$ for any $q \geq 1$. While the Hanson-Wright bound goes to ∞ as $q \to \infty$, the bound in Lemma 7 approaches the entrywise ℓ_1 bound in the limit: for $q > n$, the second term in Lemma 7 vanishes since the summand $\sum_{j>q}$ is empty. As a result, the bound becomes the first-term, which becomes $\sum_{i=1}^{n}\sum_{j=1}^{n}|a_{i,j}|$ as desired. For $1 \leq q < n$, our bound becomes an interpolation of ℓ_1 and ℓ_2 norm bounds that bears resemblance to Hitczenko's Rademacher linear form bound in [12] discussed in Section 1.2.

Although our bound is weaker than Latała's bound in [21] in the general case, it is much simpler to analyze, especially when A is a random matrix. While the bound in [21] is focused on obtaining tight estimates for quadratic forms where A is a scalar matrix, our bound is much more tractable when A is a random matrix. The main complication in the bound in [21] arises from the operator-norm-like term $\sup_{||y||_2=1,||y||_\infty\leq\frac{1}{\sqrt{q}}}|y^TAy|$. Due to the asymmetrical geometry of the ℓ_2 ball truncated by ℓ_∞ planes, this term becomes especially messy in our setting when A is a random matrix. Observe that our bound in Lemma 7 manages to avoid this term altogether. Moreover, our ℓ_1 norm term is straightforward to calculate, and our ℓ_2 norm term can be handled cleanly through a bound (Lemma 15) from [20] on the q-norm $\left\|\sum_{j>q} a_{i,j}\sigma_j\right\|_q$ that is tractable even when the $a_{i,j}$ are themselves random variables.

We defer our proof of Lemma 7 to Section 4.1. We now use Lemma 7 and the triangle inequality to obtain the following bound on $||Z||_p$:

$$\begin{aligned}
||Z||_p &= \frac{1}{s}\left(\mathbb{E}_\eta\mathbb{E}_\sigma\left[\sum_{i=1}^{n}\sum_{j\leq n, j\neq i} Q_{i,j}x_ix_j\sigma_i\sigma_j\right]^p\right)^{1/p} \\
&\lesssim \frac{1}{s}\left(\mathbb{E}_\eta\left[\sum_{i=1}^{p}\sum_{\substack{j\leq p\\ j\neq i}}|Q_{i,j}x_ix_j| + \sqrt{p}\sqrt{\sum_{i=1}^{n}\left(\mathbb{E}_\sigma\left[\sum_{\substack{j>p\\ j\neq i}} Q_{i,j}x_ix_j\sigma_j\right]^p\right)^{2/p}}\right]^p\right)^{1/p} \\
&\leq \frac{1}{s}\left(\underbrace{\left\|\sum_{i=1}^{p}\sum_{\substack{j\leq p\\ j\neq i}}|Q_{i,j}x_ix_j|\right\|_p}_{(*)} + \underbrace{\sqrt{p}\sqrt{\sum_{i=1}^{n}\left\|\sum_{\substack{j>p\\ j\neq i}} Q_{i,j}x_ix_j\sigma_j\right\|_p^2}}_{(**)}\right).
\end{aligned}$$

We first discuss some intuition for why using this bound to analyze $\|Z\|_p$ avoids the loss incurred by the Hanson-Wright bound here. In the Hanson-Wright bound, all of the Rademachers are essentially approximated by gaussians. In our bound, we make use of Rademachers in the appropriate places to avoid loss. For $1 \leq i \leq p$ and $1 \leq j \leq p$ (the upper left $p \times p$ minor where the $|x_i|$ and $|x_j|$ values are the largest), our approach utilizes

an ℓ_1-norm bound rather than $\sqrt{p}$ times an ℓ_2 bound, which turns out to allow us to save a factor of $\sqrt{p}$ in the resulting bound on $\|Z\|_p$. Now, since the original matrix is symmetric, it only remains to consider $1 \le i \le n$ and $p+1 \le j \le n$. In this range, we approximate the σ_i Rademachers by gaussians and use an ℓ_2-norm bound. It turns out that approximating the σ_j Rademachers by gaussians as well would yield too loose of a bound for our application, so we preserve the σ_j Rademachers. For the remaining Rademacher linear forms, the interaction between the x_j values (all of which are upper bounded in magnitude by $\frac{1}{\sqrt{p}}$) and the σ_j Rademachers yields the desired bound.

In order to prove Lemma 6, it remains to prove Lemma 7 as well as to bound $(*)$ and $(**)$. In Section 4.1, we prove Lemma 7. In Section 4.2 and Section 4.3, we bound $(*)$ and $(**)$. Since the building blocks of $(*)$ and $(**)$ are weighted sums of the $Q_{i,j}$ random variables, we first bound moments of these random variables separately. In Section 4.2, we use the binomial-like properties of the $Q_{i,j}$s coupled with standard moment bounds involving the binomial distribution to analyze the moments. In Section 4.3, we use these moment bounds to bound $(*)$ and $(**)$, and then finish our proof of Lemma 6. In Section 4.4, we show how Lemma 6 implies Theorem 4.

4.1 Proof of Lemma 7

We use the following standard lemmas in our proof of Lemma 7.

The first lemma allows us to decouple the two sets of Rademachers in our quadratic form so that we can reduce analyzing the moments of the quadratic form to analyzing the moments of a linear form.

▶ **Lemma 8** (Decoupling, Theorem 6.1.1 of [26]). *If $A = (a_{i,j})$ is a symmetric, zero-diagonal $n \times n$ matrix and $\{\sigma_i\}_{i\in[n]} \cup \{\sigma'_i\}_{i\in[n]}$ are independent Rademachers, then*

$$\left\| \sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}\sigma_i\sigma_j \right\|_q \lesssim \left\| \sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}\sigma'_i\sigma_j \right\|_q .$$

The next lemma is due to Khintchine and gives an ℓ_2-norm bound on linear forms of Rademachers. Since the Khintchine bound is derived from approximating $\sigma_1, \ldots, \sigma_n$ by i.i.d gaussians, we only use this bound outside of the upper left $p \times p$ minor of our matrix A.

▶ **Lemma 9** (Khintchine). *If $\sigma_1, \sigma_2, \ldots, \sigma_n$ are independent Rademachers, then for all $q \ge 1$ and $a \in \mathbb{R}^n$,*

$$\left\| \sum_{i=1}^{n} \sigma_i a_i \right\|_q \lesssim \sqrt{q}||a||_2.$$

Now, we are ready to prove Lemma 7.

Proof of Lemma 7. By Lemma 8 and the triangle inequality, we know

$$\left\| \sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}\sigma_i\sigma_j \right\|_q \lesssim \underbrace{\left\| \sum_{i=1}^{\min(q,n)}\sum_{j=1}^{\min(q,n)} a_{i,j}\sigma'_i\sigma_j \right\|_q}_{\alpha} + \underbrace{\left\| \sum_{i=1}^{n}\sum_{j>q} a_{i,j}\sigma'_i\sigma_j \right\|_q}_{\beta} + \underbrace{\left\| \sum_{i>q}\sum_{j=1}^{q} a_{i,j}\sigma'_i\sigma_j \right\|_q}_{\gamma} .$$

We first bound α. Since a Rademacher σ satisfies $|\sigma| = 1$, it follows that α can be upper bounded by the entrywise ℓ_1-norm bound $\sum_{i=1}^{\min(q,n)} \sum_{j=1}^{\min(q,n)} |a_{i,j}|$ as desired. Using Lemma 9, we know that β can be upper bounded by:

$$\sqrt{q}\left\|\sqrt{\sum_{i=1}^{n}\left(\sum_{j>q} a_{i,j}\sigma_j\right)^2}\right\|_q = \sqrt{q}\sqrt{\left\|\sum_{i=1}^{n}\left(\sum_{j>q} a_{i,j}\sigma_j\right)^2\right\|_{q/2}} \leq \sqrt{q}\sqrt{\sum_{i=1}^{n}\left\|\sum_{j>q} a_{i,j}\sigma_j\right\|_q^2}.$$

We now bound γ. An analogous argument shows $\gamma \leq \sqrt{q}\sqrt{\sum_{j=1}^{q}\left\|\sum_{i>q} a_{i,j}\sigma_i\right\|_q^2}$. Thus:

$$\gamma \leq \sqrt{q}\sqrt{\sum_{j=1}^{q}\left\|\sum_{i>q} a_{i,j}\sigma_i\right\|_q^2} \leq \sqrt{q}\sqrt{\sum_{j=1}^{n}\left\|\sum_{i>q} a_{i,j}\sigma_i\right\|_q^2} = \sqrt{q}\sqrt{\sum_{i=1}^{n}\left\|\sum_{j>q} a_{i,j}\sigma_j\right\|_q^2}. \quad \blacktriangleleft$$

4.2 Moments of Weighted Sums of $Q_{i,j}$ Random Variables

Recall that for $1 \leq i \neq j \leq n$, the $Q_{i,j}$ random variables count the number of collisions between the nonzero entries in the ith column and jth column. We first prove that these sets of random variables satisfy (conditional) independence properties, when conditioned on any choice of nonzero entries in the ith column. We also show that the moments of the random variables obtained through this conditioning are bounded by binomial moments.

▶ **Proposition 10.** *Let X be a random variable distributed as $\mathrm{Bin}(s, s/m)$. For any $1 \leq i \leq n$, given any choice of s nonzero rows $r_1 \neq r_2 \neq \ldots \neq r_s$ in the ith column, the set of $n-1$ random variables*[10] *$\{(Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i} = 1)\}_{1\leq j\leq n, j\neq i}$ are independent. Moreover, for any $q \geq 1$ and any $j \neq i$:*

$$\|Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i} = 1\|_q \leq \|X\|_q.$$

The independence properties use that the nonzero entries in different columns are independent. Moreover, the binomial bound on the moments of $Q_{i,j}$ follows from the decomposition of $Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i} = 1$ into a sum of Bernoulli random variables.

Proof of Proposition 10. Let A be a matrix drawn from $\mathcal{A}$, and pick any $1 \leq i \leq n$. We condition on the event that the s nonzero entries in column i of A occur at rows $r_1, \ldots, r_s$. For $1 \leq j \leq n$, $j \neq i$ and $1 \leq k \leq s$, let $Y_{k,j} = \eta_{r_k,j}$, so that $(Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i}) = \sum_{k=1}^{s} Y_{k,j}$. Notice that the sets $\{Y_{k,j}\}_{k\in[s]}$ for $1 \leq j \leq n, j \neq i$ are independent from each other, which means random variables in the set $\{Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i} = 1\}_{1\leq j\leq n, j\neq i}$ are independent. For $1 \leq j \leq n, j \neq i$, and $1 \leq k \leq s$, let $Z_{k,j}$ be distributed as i.i.d Bernoulli random variables with expectation s/m. Notice that for a fixed j, each $Y_{k,j}$ is distributed as $Z_{k,j}$ and the random variables $\{Y_{k,j}\}_{1\leq k\leq s}$ are negatively correlated (and nonnegative), which means

$$\|Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i} = 1\|_q = \left\|\sum_{k=1}^{s} Y_{k,j}\right\|_q \leq \left\|\sum_{k=1}^{s} Z_{k,j}\right\|_q = \|X\|_q. \quad \blacktriangleleft$$

[10] See the appendix in the full version of the paper for a formal discussion of viewing these quantities as random variables over a different probability space.

Now, we need to analyze the moments of weighted sums of $Q_{i,j}$ random variables. Using the independence properties and the fact that the moments of the $Q_{i,j}$ are upper bounded by binomial moments as given in Proposition 10, this boils down to studying the moments of weighted sums of binomial random variables. The main tools that we use in analyzing these moments are bounds on moments of sums of nonnegative random variables and sums of symmetric random variables due to Latała [20] that we state in Appendix B.[11]

Our first estimate is an upper bound on the moments of binomial random variables, which also gives bounds on moments of the $Q_{i,j}$ by Proposition 10. We defer the proof to Appendix D.

▶ **Proposition 11.** *Suppose that X is a random variable distributed as $\mathrm{Bin}(N,\alpha)$ for any $\alpha \in (0,1)$ and any integer $N \geq 1$. If $q \geq 1$ and $B = \frac{q}{\alpha \max(N,q)}$, then*

$$\|X\|_q \lesssim \begin{cases} \frac{q}{\log B} & \text{if } B \geq e \\ \frac{q}{B} & \text{if } B < e \end{cases}.$$

Our next estimate is essentially an upper bound on the moments of sums of binomial random variables weighted by Rademachers. We defer the proof to Appendix C.

▶ **Proposition 12.** *Suppose that $q \geq 2$ is even and $y = [y_1, \ldots, y_M]$ is a vector that satisfies $||y||_2 \leq 1$ and $||y||_\infty \leq \frac{1}{\sqrt{q}}$. Let X be a random variable distributed as $\mathrm{Bin}(N,\alpha)$ for some $\alpha \in (0,1)$ and some integer $N \geq 1$. Suppose that $Y_1, \ldots, Y_M$ are independent random variables that satisfy $||Y_k||_l \leq ||X||_l$ for $1 \leq k \leq M$ and for $l \geq 1$. Suppose that $\sigma_1, \ldots, \sigma_M$ are independent Rademachers, also independent of $\{Y_k\}_{k\in[M]}$. If $B = \frac{q}{\alpha \max(N,q)}$, then*

$$\left\| \sum_{k=1}^{M} Y_k y_k \sigma_k \right\|_q \lesssim \begin{cases} \frac{\sqrt{q}}{\log B} & \text{if } B \geq e \\ \frac{\sqrt{q}}{B} & \text{if } B < e \end{cases}.$$

4.3 Bounding (∗) and (∗∗) to prove Lemma 6

We bound the quantities $(*)$ and $(**)$ in the following sublemmas, which assume the notation used throughout the paper:

▶ **Lemma 13.** *If $m/s^2 = B$, then*

$$\left\| \sum_{i=1}^{p} \sum_{j \leq p, j \neq i} |Q_{i,j} x_j x_i| \right\|_p \lesssim \begin{cases} \frac{p}{\log B} & \text{if } B \geq e \\ \frac{p}{B} & \text{if } B < e \end{cases}.$$

▶ **Lemma 14.** *If $m/s^2 = B$, then*

$$\sqrt{p} \sqrt{\sum_{i=1}^{n} \left\| \sum_{j>p, j\neq i} Q_{i,j} x_i x_j \sigma_j \right\|_p^2} \lesssim \begin{cases} \frac{p}{\log B} & \text{if } B \geq e \\ \frac{p}{B} & \text{if } B < e \end{cases}.$$

We now use Proposition 10 as well as the moment bound on binomial random variables from Proposition 11 to prove Lemma 13 and thus bound $(*)$.

[11] The proofs of these bounds given in [20] are not complicated; for the sake of being self-contained, we give sketches of these proofs in the appendix of the full version of the paper.

Proof of Lemma 13. We carefully use the triangle inequality to see[12]:

$$\left\|\sum_{i=1}^{p}\sum_{\substack{j\le p\\ j\neq i}}|Q_{i,j}x_jx_i|\right\|_p \le 2\left\|\sum_{i=1}^{p}\sum_{\substack{j\le p\\ j>i}}Q_{i,j}|x_j||x_i|\right\|_p \lesssim \left\|\sum_{i=1}^{p}x_i^2\sum_{\substack{j\le p\\ j>i}}Q_{i,j}\right\|_p \lesssim \sum_{i=1}^{p}x_i^2\left\|\sum_{\substack{j\le p\\ j>i}}Q_{i,j}\right\|_p.$$

Let $X \sim \mathrm{Bin}(s, s/m)$ and $Y \sim \mathrm{Bin}(sp, s/m)$. By Proposition 10, for any i and any $r_1 \neq r_2 \neq \ldots \neq r_s$, the random variables $\{Q_{i,j} \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\}_{j\neq i}$ are independent and $\|Q_{i,j} \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\|_p \le \|X\|_p$. It follows from taking pth powers of both sides that

$$\left\|\left(\sum_{\substack{j\le p\\ j>i}}Q_{i,j}\right) \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\right\|_p = \left\|\sum_{\substack{j\le p\\ j>i}}(Q_{i,j} \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1)\right\|_p \le \|Y\|_p.$$

Now, Proposition 11 gives us a bound on $\|Y\|_p$, and the result follows from the law of total expectation.[13] ◀

We now use Proposition 10 as well as the moment bound on weighted sums of binomial random variables from Proposition 12 to prove Lemma 14 and thus bound (∗∗).

Proof of Lemma 14. Observe that

$$\sqrt{p}\sqrt{\sum_{i=1}^{n}\left\|\sum_{\substack{j>p\\ j\neq i}}Q_{i,j}x_ix_j\sigma_j\right\|_p^2} = \sqrt{p}\sqrt{\sum_{i=1}^{n}x_i^2\left\|\sum_{\substack{j>p\\ j\neq i}}Q_{i,j}x_j\sigma_j\right\|_p^2} \le \sqrt{p}\max_{1\le i\le n}\left\|\sum_{\substack{j>p\\ j\neq i}}Q_{i,j}x_j\sigma_j\right\|_p.$$

Let $X \sim \mathrm{Bin}(s, s/m)$ and $Y \sim \mathrm{Bin}(sp, s/m)$. By Proposition 10, for any i and any $r_1 \neq r_2 \neq \ldots \neq r_s$, the random variables $\{Q_{i,j} \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\}_{j\neq i}$ are independent and $\|Q_{i,j} \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\|_p \le \|X\|_p \le \|Y\|_p$. Moreover, $|x_j| \le \frac{1}{\sqrt{p}}$ for $j > p$. Now, we consider $\left\|\sum_{j>p,j\neq i}Q_{i,j}x_j\sigma_j \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\right\|_p$ which is equal to $\left\|\sum_{j>p,j\neq i}(Q_{i,j} \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1)(\sigma_j \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1)x_j\right\|_p$. Since each $(\sigma_j \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1)$ is distributed as a Rademacher and since the set of $n-1$ random variables $\{\sigma_j \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\}_{j\neq i}$ are independent and also independent of $\{Q_{i,j} \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\}_{j\neq i}$, we can apply Proposition 12 to this expression and thus get a bound[14] on the conditional p-norm $\left\|\sum_{j>p,j\neq i}Q_{i,j}x_j\sigma_j \mid \eta_{r_1,i} = \ldots = \eta_{r_s,i} = 1\right\|_p$. Now, the result follows from the law of total expectation. ◀

We now show the bound on $||Z||_p$ follows from the bounds on (∗) and (∗∗) in Lemmas 13, 14.

[12] Naively applying the triangle inequality yields a suboptimal bound, so we require this more careful treatment.

[13] See the appendix in the full version of the paper for a formal discussion of why a uniform bound on the conditional p-norm implies a bound on the p-norm here.

[14] Approximating the σ_j by gaussians yields a suboptimal bound, so we require the bound given in Proposition 12.

Proof of Lemma 6. Applying Lemmas 13,14 after the following simplification proves the lemma:

$$\|Z\|_p \lesssim \frac{1}{s}\left\|\sum_{i=1}^{p}\sum_{j\le p, j\neq i}|Q_{i,j}x_ix_j|\right\|_p + \frac{\sqrt{p}}{s}\sqrt{\sum_{i=1}^{n}\left\|\sum_{j>p,j\neq i}Q_{i,j}x_ix_j\sigma_j\right\|_p^2}. \quad \blacktriangleleft$$

4.4 Proof of Theorem 4

We show Lemma 6 implies Theorem 4, completing the proof.

Proof of Theorem 4. It suffices to show $\mathbb{P}_{\eta,\sigma}[|Z| > \epsilon] < \delta$. By Markov's inequality, we know

$$\mathbb{P}_{\eta,\sigma}[|Z| > \epsilon] = \mathbb{P}_{\eta,\sigma}[|Z|^p > \epsilon^p] < \epsilon^{-p}\mathbb{E}[|Z|^p] = \left(\frac{\|Z\|_p}{\epsilon}\right)^p.$$

Suppose that $B \geq e$. Then by Lemma 6, we know

$$\left(\frac{\|Z\|_p}{\epsilon}\right)^p \leq \left(\frac{Cp}{(\log B)s\epsilon}\right)^p.$$

Thus, to upper bound this quantity by δ, we can set $s = \Theta(\varepsilon^{-1}p/\log B) = \Theta(\varepsilon^{-1}\log_B(1/\delta))$ and $m = \Theta(Bs^2)$. We impose the additional constraint that $B \leq \frac{1}{\delta}$ to guarantee that $s \geq 1$. This proves the desired result.[15] $\blacktriangleleft$

References

1 D. Achlioptas. Database-friendly Random Projections: Johnson-Lindenstrauss with Binary Coins. *J. Comput. Syst. Sci.*, 66(4):671–687, June 2003.

2 Z. Allen-Zhu, R. Gelashvili, S. Micali, and N. Shavit. Sparse sign-consistent Johnson–Lindenstrauss matrices: Compression with neuroscience-based constraints. In *Proceedings of the National Academy of Sciences (PNAS)*, volume 111, pages 16872–16876, 2014.

3 M. B. Cohen. Nearly tight oblivious subspace embeddings by trace inequalities. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 278–287, 2016.

4 M. B. Cohen, T. S. Jayram, and J. Nelson. Simple Analyses of the Sparse Johnson-Lindenstrauss Transform. In *Proceedings of the 1st Symposium on Simplicity in Algorithms (SOSA)*, pages 1–9, 2018.

5 S. Dahlgaard, M. Knudsen, and M. Thorup. Practical Hash Functions for Similarity Estimation and Dimensionality Reduction. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, pages 6618–6628, 2017.

6 A. Dasgupta, R. Kumar, and T. Sarlos. A Sparse Johnson-Lindenstrauss Transform. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 341–350, 2010.

7 C. Freksen, L. Kamma, and K. G. Larsen. Fully Understanding the Hashing Trick. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*, pages 5394–5404, 2018.

8 Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 62:233–241, 1981.

[15] If we set $B < e$, if we use Lemma 6, we know that in order to obtain an upper bound of δ, we would have to set $s = \Theta(\varepsilon^{-1}p/B) = \Theta(\varepsilon^{-1}\log(1/\delta)/B)$ and $m = \Theta(\varepsilon^{-1}\log^2(1/\delta)/B)$. This yields no better s or m values than those achieved when $B = e$.

9 S. Ganguli and H. Sompolinsky. Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis. *Annual Review of Neuroscience*, 35:485–508, 2012.

10 R.T. Gray and P.A. Robinson. Stability and structural constraints of random brain networks with excitatory and inhibitory neural populations. *Journal of Computational Neuroscience*, 27(1):81–101, 2009.

11 D. L. Hanson and F. T. Wright. A bound on tail probabilities for quadratic forms in independent random variables. *Annals of Mathematical Statistics*, 42(3):1079–1083, 1971.

12 P. Hitczenko. Domination inequality for martingale transforms of Rademacher sequence. *Israel Journal of Mathematics*, 84:161–178, 1993.

13 M. Jagadeesan. Understanding Sparse JL for Feature Hashing. *CoRR*, abs/1903.03605, 2019. `arXiv:1903.03605`.

14 T.S. Jayram and D. P. Woodruff. Optimal bounds for Johnson-Lindenstrauss transforms and steaming problems with subconstant error. In *ACM Transactions on Algorithms (TALG) - Special Issue on SODA'11*, volume 9, pages 1–26, 2013.

15 W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

16 D. M. Kane, R. Meka, and J. Nelson. Almost optimal explicit Johnson-Lindenstrauss families. In *Proceedings of the 14th International Workshop and 15th International Conference on Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques (RANDOM)*, pages 628–639, 2011.

17 D. M. Kane and J. Nelson. A Derandomized Sparse Johnson-Lindenstrauss Transform. *CoRR*, abs/1006.3585, 2010. `arXiv:1006.3585`.

18 D. M. Kane and J. Nelson. Sparser Johnson-Lindenstrauss transforms. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 16872–16876. ACM Press, 2012.

19 R. Kiani, H. Esteky, K. Mirpour, and K. Tanaka. Object category structure in response patterns of neuronal population in monkey inferior temporal cortex. *Journal of Neurophysiology*, 97:4296–4309, 2007.

20 R. Latała. Estimation of moments of sums of independent real random variables. *Annals of Probability*, 25(3):1502–1513, 1997.

21 R. Latała. Tail and moment estimates for some types of chaos. *Studia Mathematica*, 135(1):39–53, 1999.

22 S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1, 2005.

23 K. Rajan and L.F. Abbot. Eigenvalue spectra of random matrices for neural networks. *Physical Review Letters*, 97:188104, 2006.

24 T. Rogers and J. McClelland. *Semantic Cognition: A Parallel Distributed Processing Approach*. MIT Press, 2004.

25 D. Spielman and N. Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing (SICOMP)*, 40:1913–1926, 2011.

26 R. Vershynin. *High-Dimensional Probability*. Cambridge University Press, 2018.

27 K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature Hashing for Large Scale Multitask Learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 1113–1120, 2009.

28 E.P. Wigner. Characteristic vectors of bordered matrices with infinite dimensions. *Annals of Mathematics*, 62:548–564, 1955.

29 D.P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10:1–157, 2014.

A Limitations of the Combinatorial Approach

At first glance, it appears that the bound of $\mathbb{P}[|Z| > \epsilon] \leq \left(\frac{Cts}{\epsilon m}\right)^t$ of [2] in the proof of the main theorem (p. 9 of the arXiv version) implies the desired dimension-sparsity tradeoffs by setting $s = \epsilon^{-1}t$, $m = \frac{B\epsilon^{-1}ts}{2C}$, and $t = \log_B(1/\delta)$ (this t value is equivalent to the p value in this paper). However, this does not actually follow from the analysis in [2]: there is an assumption made in one of the lemmas, which is not stated explicitly in the statement of the lemma, that does not allow the parameters to be set in this way. The limiting factor is the lemma that states that

$$s^t\mathbb{E}[Z^t] \leq 2^{O(t)}t^t\left(\frac{s^2}{m}\right)^t.$$

This is Lemma 3 in the conference version of [2], and Lemma 4.3 in the arXiv version of [2]. Here, Z is defined analogously as in section 2.1 of this paper.

The proof of this lemma, given in Appendix A.3 in [2], implicitly relies on the fact that $\frac{s^2}{m} \geq 1$, although this condition is not explicitly stated in the lemma statement. This assumption arises from the last line of the proof, where the sum $\sum_{w=1}^{t}\left(\frac{s^2}{m}\right)^w$ is upper bounded by $t\left(\frac{s^2}{m}\right)^t$. Following the end of the proof of Theorem 1 (the top of p. 9 of the arXiv version), this yields $\mathbb{P}[|Z| > \epsilon] \leq \left(\frac{Cts}{\epsilon m}\right)^t$. Now, suppose we instead set $m = Bs^2$ (where $B \leq 1$ as required by the assumption). Then we obtain $\left(\frac{Cts}{\epsilon m}\right)^t = \left(\frac{Ct}{\epsilon Bs}\right)^t$. Thus, we can set s to be $C\epsilon^{-1}B^{-1}\log(1/\delta)$ and m to be $C^2\epsilon^{-2}\log^2(1/\delta)B^{-1}$. Since $B \leq 1$, this is no better than the original theorem statement and thus yields no dimension-sparsity tradeoff.

Now, suppose we instead let $\frac{s^2}{m} \leq 1$. Then we can modify the proof of Lemma 4.3 to obtain the weaker upper bound of $\sum_{w=1}^{t}\left(\frac{s^2}{m}\right)^w$ by $t\frac{s^2}{m}$. Let $B = m/s^2$ where $B \geq 1$. In order to ensure that m is polynomial in $\log(1/\delta)$, we assume that $B \leq \delta$. In this case, mimicking the calculation at the end of the proof of Theorem 1, we obtain $\mathbb{P}[|Z| > \epsilon] \leq \frac{1}{B}\left(\frac{Ct}{\epsilon s}\right)^t = \left(\frac{Ct}{\epsilon s B^{1/t}}\right)^t$. Thus, we can set $s = C\log(1/\delta)\epsilon^{-1}e^{-\log B/t}$. Observe that $0 \leq \log B \leq t$, so $1 \geq e^{-\log B/t} \geq e^{-1}$. Thus, $s = \Theta(\log(1/\delta)\epsilon^{-1})$ and $m = \Theta(Bs^2)$, which does not yield a dimension-sparsity tradeoff.

Thus, it is not clear how to directly obtain the dimension-sparsity tradeoff from the combinatorial approach of [2]. Some intuition for this limitation is that the moment bounds on Z obtained by the combinatorial approach are not sufficiently tight for varying values of B due to the fact that the bounding techniques are implicitly tailored to the case of $B = \Theta(1)$. The combinatorics-free approach in this paper avoids this issue through making use of a more structured method to bound the moments of Z.

B Latała's Moment Bounds

The following bounds on sums of independent random variables are due to Latała [20]. These proofs are not complicated: for sake of being self-contained, in the appendix of the full version of the paper, we sketch proofs of these bounds. Full proofs of these lemmas can be found in [20].

▶ **Lemma 15** ([20]). *If $q \geq 2$ and $X, X_1, \ldots, X_n$ are independent symmetric random variables, then*

$$\left\|\sum_{i=1}^{n}X_i\right\|_q \simeq \inf\left\{T > 0 \text{ such that } \sum_{i=1}^{n}\log\left(\mathbb{E}\left[\left(1+\frac{X_i}{T}\right)^q\right]\right) \leq q\right\}.$$

▶ **Lemma 16** ([20][16]). *If $1 \leq q \leq n$ and $X, X_1, \ldots, X_n$ are i.i.d nonnegative random variables, then*

$$\left\| \sum_{i=1}^{n} X_i \right\|_q \simeq \sup_{1 \leq t \leq q} \frac{q}{t} \left(\frac{n}{q}\right)^{1/t} \|X\|_t .$$

C Proof of Proposition 12

The main ingredient in this proof is Lemma 15 (Latała'a bound on moments of sums of symmetric random variables).

Proof of Proposition 12. Since the Y_i are independent random variables, we can apply Lemma 15 to obtain:

$$\left\| \sum_{k=1}^{M} Y_k y_k \sigma_k \right\|_q \lesssim \inf \left\{ T > 0 \mid \sum_{k=1}^{M} \log \left(\mathbb{E} \left[\left| 1 + \frac{Y_k \sigma_k y_k}{T} \right|^q \right] \right) \leq q \right\}.$$

Thus, it suffices to show

$$T \simeq \begin{cases} \frac{\sqrt{q}}{\log B} & \text{if } B \geq e \\ \frac{\sqrt{q}}{B} & \text{if } B < e \end{cases}$$

satisfies $\sum_{k=1}^{M} \log \left(\mathbb{E} \left[\left(1 + \frac{Y_k \sigma_k y_k}{t} \right)^q \right] \right) \leq q$. We see

$$\begin{aligned} \sum_{k=1}^{M} \log \left(\mathbb{E} \left[\left(1 + \frac{Y_k \sigma_k y_k}{T} \right)^q \right] \right) &= \sum_{k=1}^{M} \log \left(1 + \sum_{l=1}^{q} \binom{q}{l} \frac{(\mathbb{E}[(Y_k \sigma_k)^l]) y_k^l}{T^l} \right) \\ &= \sum_{k=1}^{M} \log \left(1 + \sum_{l=1}^{q/2} \binom{q}{2l} \frac{\|Y_k\|_{2l}^{2l} y_k^{2l}}{T^{2l}} \right) \\ &\leq \sum_{k=1}^{M} \log \left(1 + \sum_{l=1}^{q/2} \left(\frac{qe}{2l} \right)^{2l} \left(\frac{\|Y_k\|_{2l} y_k}{T} \right)^{2l} \right) \end{aligned}$$

By the bound on moments of binomial random variables in Proposition 11, we know if $B \geq e$ that there exists a universal constant C such that $||Q_{i,j}||_{2l} \leq \frac{2lC}{\log B}$. Thus, we obtain

$$\begin{aligned} \sum_{k=1}^{M} \log \left(\mathbb{E} \left[\left(1 + \frac{Y_k \sigma_k y_k}{T} \right)^q \right] \right) &\leq \sum_{k=1}^{M} \log \left(1 + \sum_{l=1}^{q/2} \left(\frac{qe}{2l} \right)^{2l} \left(\frac{2lCy_k}{T \log B} \right)^{2l} \right) \\ &\leq \sum_{k=1}^{M} \log \left(1 + \sum_{l=1}^{q/2} \left(\frac{qeCy_k}{T \log B} \right)^{2l} \right). \end{aligned}$$

[16] This result was actually first due to S.J. Montgomery-Smith through a private communication with Latała. Nonetheless, it is also a corollary of a result in [20].

Since $|y_k| \le \frac{1}{\sqrt{q}}$, if we set $T = \frac{2eC\sqrt{q}}{\log B}$, then we obtain

$$\sum_{k=1}^{M} \log\left(1 + \sum_{l=1}^{q/2}\left(\frac{\sqrt{q}y_k}{2}\right)^{2l}\right) \le \sum_{k=1}^{M} \log\left(1 + (\sqrt{q}y_k)^2 \sum_{l=1}^{q/2}\left(\frac{1}{2}\right)^{2l}\right).$$

This can be bounded by

$$\sum_{k=1}^{M} \log\left(1 + (\sqrt{q}y_k)^2\right) = \sum_{k=1}^{M} \log\left(1 + qy_k^2\right) \le \sum_{i=1}^{n} qy_k^2 \le q$$

as desired. An analogous argument shows that if $B < e$, we can set $T = \frac{2eC\sqrt{q}}{B}$. ◀

D Proof of Proposition 11

The main tool that we use in this proof is Lemma 16 (Latała's bound on moments of sums of i.i.d nonnegative random variables).

Proof of Proposition 11. Notice that it suffices to obtain an upper bound on $\|X\|_q$ for all $N \ge q$. (Since $\|X\|_q$ is an increasing function of N, an upper bound on $\|X\|_q$ at $N = q$ is also an upper bound on $\|X\|_q$ for all $N < q$). For the rest of the proof, we assume $N \ge q$.

Notice X has the same distribution as $\sum_{j=1}^{N} Z_j$ where $Z, Z_1, \ldots, Z_N$ are i.i.d Bernoulli random variables with expectation α. Since $\|Z\|_t = \alpha^{1/t}$, we know by Lemma 16,

$$\begin{aligned}\|X\|_q &\simeq \sup_{1\le t\le q} \frac{q}{t}\left(\frac{N}{q}\right)^{1/t} \alpha^{1/t} \\ &= \sup_{1\le t\le q} \frac{q}{t}\left(\frac{1}{B}\right)^{1/t}\end{aligned}$$

At $t = 1$, this quantity is equal to $\frac{q}{B}$, and at $t = q$, this quantity is equal to $\left(\frac{1}{B}\right)^{1/q} = e^{\log(1/B)/q}$. The only $t \in \mathbb{R}$ for which this quantity has derivative 0 is $t = \log B$. Notice that $1 \le \log B \le q$ if and only if $e \le B \le e^q$. Thus

$$\|X\|_q \simeq \begin{cases} \max(\frac{q}{B}, \frac{q}{\log B}, e^{\log(1/B)/q}) & \text{if } e \le B \le e^q \\ \max(\frac{q}{B}, e^{\log(1/B)/q}) & \text{if } B < e \text{ or if } B > e^q. \end{cases}.$$

For $B \ge e$, we want to show $\|X\|_q \lesssim q/\log B$. Since $\log B > 0$, we see $e^{\log(1/B)/q} = e^{-\log B/q} \le q/\log B$ and $q/B \le q/\log B$.

For $B < e$, we want to show $\|X\|_q \lesssim q/B$. Since $\frac{1}{B} > \frac{1}{e}$, we see $e^{\log(1/B)/q} = \left(\frac{1}{B}\right)^{1/q} \le \frac{e}{B} \lesssim \frac{q}{B}$. ◀

E Weakness of bound on $\|Z\|_p$ from Equation (4)

Like in Section 4, we view the random variable Z as a quadratic form $\frac{1}{s}\sigma^T A\sigma$, where σ an n-dimensional vector of independent Rademachers and A is a symmetric, zero-diagonal $n \times n$ matrix where the (i,j)th entry (for $i \ne j$) is $x_i x_j \sum_{r=1}^{m} \eta_{r,i}\eta_{r,j} = Q_{i,j}x_i x_j$. Applying the Hanson-Wright bound followed by an expectation over the η values yields

$$\left\|\sigma^T A\sigma\right\|_p \lesssim \left\| \sqrt{p}\sqrt{\sum_{i=1}^{n}\sum_{j\le n, j\ne i} Q_{i,j}^2 x_i^2 x_j^2} + p \sup_{\|y\|_2=1}\left|\sum_{i=1}^{n}\sum_{j\le n, j\ne i} Q_{i,j}x_i x_j y_i y_j\right| \right\|_p =: U_p. \quad (4)$$

We show that the vector $x = [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, \ldots, 0] \in \mathbb{R}^n$ forces U_p to be too large to yield the optimal m value, thus proving that the Hanson-Wright bound does not provide a sufficiently tight bound on $\|Z\|_p$ to achieve Theorem 3. The main ingredient in our proof is the following lemma, which we prove in subsection C.1:

▶ **Lemma 17.** *For every column $1 \leq i \leq n$, suppose that the random variables $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ have the distribution defined by uniformly choosing exactly s of the variables per column. If $x = \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, \ldots, 0\right]$, $p < s$ and $B = m/s^2 \leq \frac{e^p}{p}$, then*

$$U_p \simeq \begin{cases} \frac{p^2}{\log Bp} & \text{if } B \geq \frac{e}{p} \\ \frac{p}{B} & \text{if } B < \frac{e}{p}. \end{cases}$$

We can obtain bounds on s and m from Lemma 17 via Markov's inequality. We disregard the case where $B \geq \frac{e^p}{p}$, since this case would yield a value for m that is not polynomial in $\log(1/\delta)$. If $B < e/p$, then it follows that $s = \Theta(\varepsilon^{-1}B^{-1}\log(1/\delta)) = \Omega(\varepsilon^{-1}\log^2(1/\delta))$ and $m = \Theta(\varepsilon^{-2}B^{-1}\log^2(1/\delta)) = \Omega(\varepsilon^{-2}\log^3(1/\delta))$. If $B \geq e/p$, then it follows that $s = \Theta(\varepsilon^{-1}p^2/\log(Bp)) = \Omega(\varepsilon^{-1}\log(1/\delta))$ and $m = \Theta(\varepsilon^{-2}p^4B/\log^2(Bp)) = \Omega(\varepsilon^{-2}\log^3(1/\delta))$. These bounds on m incur an extra $\log(1/\delta)$ factor, and thus the Hanson-Wright bound is too weak for this setting. Now, it suffices to prove Lemma 17, which we do in the next section.

E.1 Proof of Lemma 17

In this section, we assume that $x = \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, \ldots, 0\right]$ and that the random variables $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ have the distribution defined by uniformly choosing exactly s of the variables per column. We first show the following computation of $||Q_{i,j}||_p$.

▶ **Proposition 18.** *Assume that the random variables $\{\eta_{r,i}\}_{r\in[m],i\in[n]}$ have the distribution defined by uniformly choosing exactly s of the variables per column. Then, if $p < s$ and $X \sim \mathrm{Bin}(s, s/m)$, we have that $||Q_{i,j}||_p \simeq ||X||_p$.*

Proof. We condition on the even that the nonzero locations in column i are at $r_1, r_2, \ldots, r_s$. Notice that the random variable $(Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i} = 1)$ is distributed as $Z_{r_1} + Z_{r_2} + \ldots + Z_{r_s}$ where Z_{r_k} is an indicator for the kth entry in the jth column being nonzero. Let Z'_{r_k} for $1 \leq k \leq s$ be i.i.d random variables distributed as $\mathrm{Bern}(s/m)$. Now, observe that

$$\mathbb{E}[(Z_{r_1} + Z_{r_2} + \ldots + Z_{r_s})^p] = \sum_{\substack{0\leq t_1,t_2,\ldots,t_s\leq p \\ t_1+t_2+\ldots+t_s=p}} \mathbb{E}[\prod_{i=1}^{s} Z_{r_i}^{t_i}] = \sum_{\substack{0\leq t_1,t_2,\ldots,t_s\leq p \\ t_1+t_2+\ldots+t_s=p}} \mathbb{E}[\prod_{i|t_i>0} Z_{r_i}].$$

Notice that $\mathbb{E}[(Z'_{r_1} + Z'_{r_2} + \ldots + Z'_{r_s})^p] = \sum_{0\leq t_1,t_2,\ldots,t_s\leq p, t_1+t_2+\ldots+t_s=p} \mathbb{E}[\prod_{i|t_i>0} Z'_{r_i}]$. Thus, it suffices to compare $\mathbb{E}[\prod_{i|t_i>0} Z_{r_i}]$ and $\mathbb{E}[\prod_{i|t_i>0} Z'_{r_i}]$. We see that $\mathbb{E}[\prod_{i|t_i>0} Z'_{r_i}] = \left(\frac{s}{m}\right)^{|\{i|t_i>0\}|}$. Since $p < s$, we see that $\mathbb{E}[\prod_{i|t_i>0} Z_{r_i}] = \prod_{j=0}^{|\{i|t_i>0\}|-1} \frac{s-j}{m-j}$. It is not difficult to verify that this ratio is bounded by $2^{O(p)}$ as desired, so

$$\frac{\mathbb{E}[(Q_{i,j} \mid \eta_{r_1,i} = \eta_{r_2,i} = \ldots = \eta_{r_s,i} = 1)^p]}{\mathbb{E}[X^p]} = \frac{\mathbb{E}[(Z_{r_1} + Z_{r_2} + \ldots + Z_{r_s})^p]}{\mathbb{E}[X^p]} \geq 2^{-O(p)}.$$

Now, by the law of total expectation, we know that

$$\frac{\mathbb{E}[Q_{i,j}^p]}{\mathbb{E}[X^p]} \geq 2^{-O(p)}$$

as desired. ◀

We now prove the following relation between U_p and $\|Q_{1,2}\|_p$:

▶ **Lemma 19.** *Assume the notation and restrictions above. Then* $U_p \simeq p\,\|Q_{1,2}\|_p$.

Proof of Lemma 19. For ease of notation, we define

$$S_1 := p \sup_{\|y\|_2=1} \left| \sum_{i=1}^{n} \sum_{j \le n, j \neq i} Q_{i,j} x_i x_j y_i y_j \right|$$

$$S_2 := \sqrt{p} \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} Q_{i,j}^2 x_i^2 x_j^2}.$$

Our goal is to calculate $U_p = \|S_1 + S_2\|_p$. We make use of the following upper and lower bounds on $\|S_1 + S_2\|_p$:

$$\left| \|S_1\|_p - \|S_2\|_p \right| \le \|S_1 - S_2\|_p \le \|S_1 + S_2\|_p \le \|S_1\|_p + \|S_2\|_p . \tag{5}$$

In order to compute $\left| \|S_1\|_p - \|S_2\|_p \right|$ and $\|S_1\|_p + \|S_2\|_p$, we first compute $\|S_1\|_p$ and $\|S_2\|_p$. For our choice of x, notice

$$\|S_1\|_p \simeq p \left\| \sup_{\|y\|_2=1} |Q_{1,2} y_1 y_2| \right\|_p \simeq p\,\|Q_{1,2}\|_p$$

$$\|S_2\|_p \simeq \sqrt{p} \left\| \sqrt{Q_{1,2}^2} \right\|_p = \sqrt{p}\,\|Q_{1,2}\|_p .$$

From these bounds, coupled with (5), it follows that $\|U\|_p \simeq p\,\|Q_{1,2}\|_p$ as desired. ◀

We now show Lemma 17 follows from Lemma 19 and Proposition 18.

Proof of Lemma 17. After applying Lemma 19, it suffices to calculate $\|Q_{1,2}\|_p$. It follows from Proposition 18 that $\|Q_{1,2}\|_p \simeq \|X\|_p$ where X is distributed as $\mathrm{Bin}(s, s/m)$. Now, the following calculation $\|X\|_p$ for $p < s$ and $B = m/s^2 \le \frac{e^p}{p}$ follows from the lower and upper bounds of Lemma 16 (Latała's bound on moments of sums of i.i.d nonnegative random variables):

$$\|X\|_p \simeq \begin{cases} \frac{p}{\log Bp} & \text{if } B \ge \frac{e}{p} \\ \frac{1}{B} & \text{if } B < \frac{e}{p} \end{cases} .$$

From this, Lemma 17 follows. ◀

Streaming Coreset Constructions for M-Estimators

Vladimir Braverman
Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA
vova@jhu.edu

Dan Feldman
Department of Computer Science, University of Haifa, Israel
dannyf.post@gmail.com

Harry Lang
MIT CSAIL, Cambridge, MA, USA
harry1@mit.edu

Daniela Rus
MIT CSAIL, Cambridge, MA, USA
rus@mit.edu

Abstract

We introduce a new method of maintaining a (k, ϵ)-coreset for clustering M-estimators over insertion-only streams. Let (P, w) be a weighted set (where $w : P \to [0, \infty)$ is the weight function) of points in a ρ-metric space (meaning a set $\mathcal{X}$ equipped with a positive-semidefinite symmetric function D such that $D(x, z) \leq \rho(D(x, y) + D(y, z))$ for all $x, y, z \in \mathcal{X}$). For any set of points C, we define $\mathsf{COST}(P, w, C) = \sum_{p \in P} w(p) \min_{c \in C} D(p, c)$. A (k, ϵ)-coreset for (P, w) is a weighted set (Q, v) such that for every set C of k points, $(1 - \epsilon)\mathsf{COST}(P, w, C) \leq \mathsf{COST}(Q, v, C) \leq (1 + \epsilon)\mathsf{COST}(P, w, C)$. Essentially, the coreset (Q, v) can be used in place of (P, w) for all operations concerning the COST function. Coresets, as a method of data reduction, are used to solve fundamental problems in machine learning of streaming and distributed data.

M-estimators are functions $D(x, y)$ that can be written as $\psi(d(x, y))$ where $(\mathcal{X}, d)$ is a true metric (i.e. 1-metric) space. Special cases of M-estimators include the well-known k-median ($\psi(x) = x$) and k-means ($\psi(x) = x^2$) functions. Our technique takes an existing offline construction for an M-estimator coreset and converts it into the streaming setting, where n data points arrive sequentially. To our knowledge, this is the first streaming construction for any M-estimator that does not rely on the merge-and-reduce tree. For example, our coreset for streaming metric k-means uses $O(\epsilon^{-2} k \log k \log n)$ points of storage. The previous state-of-the-art required storing at least $O(\epsilon^{-2} k \log k \log^4 n)$ points.

2012 ACM Subject Classification Theory of computation → Streaming models; Theory of computation → Facility location and clustering; Information systems → Query optimization

Keywords and phrases Streaming, Clustering, Coresets

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.62

Category RANDOM

Funding *Vladimir Braverman*: This research was supported in part by NSF CAREER grant 1652257, ONR Award N00014-18-1-2364, DARPA/ARO award W911NF1820267.
Harry Lang: This material is based upon work supported by the Franco-American Fulbright Commission. The author thanks INRIA (l'Institut national de recherche en informatique et en automatique) for hosting him during part of the writing of this paper.
Daniela Rus: This research was supported in part by NSF 1723943, NVIDIA, and J.P. Morgan Chase & Co.



© Vladimir Braverman, Dan Feldman, Harry Lang, and Daniela Rus;
licensed under Creative Commons License CC-BY



1 Introduction

In the streaming model of computation, the input arrives sequentially. This differs from the random-access memory model (i.e. the offline setting) where the algorithm may freely and repeatedly access the entire input. The goal of a streaming algorithm is to perform the computation using a sublinear amount of memory.

A stream consists of n elements $p_1, \ldots, p_n$. Sometimes the algorithm will be allowed to pass over the stream multiple times, resulting in another parameter called the number of passes. Our algorithm uses $O(\log n)$ memory and requires only a single pass over the input stream.

Prior to the current work, the merge-and-reduce technique due to Har-Peled and Mazumdar [19] and Bentley and Sax [5] was used to maintain coresets on streams using an offline coreset construction as a blackbox. For a brief review of this technique, see Section 4.1. Suppose the offline construction's space depends on ϵ as ϵ^{-a}. In this paper we introduce an alternative technique that reduces the multiplicative overhead from $O(\log^{a+1} n)$ to $O(1)$ when moving to the streaming setting. For example, the state-of-the-art k-median offline coreset [6] has size $O(\epsilon^{-2} k \log k \log n)$. The current paper improves the space requirement from $O(\epsilon^{-2} k \log k \log^4 n)$ to $O(\epsilon^{-2} k \log k \log n)$ to maintain the coreset on a stream. While our method is not as general as merge-and-reduce (it requires a function to satisfy more than just the "merge" and "reduce" properties, defined in Section 4.1), it is general enough to apply to all M-estimators.

2 Definitions

We begin by defining a ρ-metric space. Let $\mathcal{X}$ be a set. If $D : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ is a symmetric positive-semidefinite function such that for every $x, y, z \in \mathcal{X}$ we have that $D(x, z) \leq \rho(D(x, y) + D(y, z))$ then we call $(\mathcal{X}, D)$ a ρ-metric space. Note that this is a weakening of the triangle inequality, and at $\rho = 1$ we recover the definition of a metric space. Clustering M-estimators in a metric space can be re-cast as k-median in a ρ-metric space. For example, metric k-means in the space $(\mathcal{X}, d)$ is reducible to 2-metric k-median in the space $(\mathcal{X}, D)$ using $D(\cdot, \cdot) = d(\cdot, \cdot)^2$. See Table 1 for more examples. We work in this slightly abstract language since it allows a single proof to naturally generalize our results to any M-estimator.

A weighted set (P, w) is a set $P \subset \mathcal{X}$ along with a weight function $w : P \to [0, \infty)$. As usual, we define the distance between a point and a set $D(p, Z) = \min_{z \in Z} D(p, Z)$. Let

$$\mathsf{COST}(P, w, Z) = \sum_{p \in P} w(p) D(p, Z)$$

The ρ-metric k-median problem is, given input (P, w) and an integer $k \geq 1$, to find a set C of k points in $\mathcal{X}$ that minimizes $\mathsf{COST}(P, w, C)$. We use $\mathsf{OPT}_k(P)$ to denote the minimal value. Other works have shown that for small enough ϵ, it is NP-Hard to even compute a $(1 + \epsilon)$ approximation for k-means or k-median when k is part of the input (see the Related Work section of [4] for a survey of hardness results). Therefore weaker definitions of approximation have been introduced. The notion of a bicriterion approximation is well-known; we state a slightly more verbose definition that suits our needs. The difference is that usually the map f is implicit, simply mapping a point to a nearest center.

▶ **Definition 1** ((α,β)-approximation). *For $\alpha,\beta \geq 1$, an (α,β)-approximation for the k-median clustering of a weighted set (P,w) is a weighted set (B,v) along with a map $f : P \to B$ such that:*

1. $\sum_{p\in P} w(p)D(p,f(p)) \leq \alpha \mathsf{OPT}_k(P)$
2. $|B| \leq \beta k$
3. $v(b) = \sum_{p\in f^{-1}(b)} w(p)$ *for every* $b \in B$

Observe that a $(1,1)$-approximation is an optimal solution. We now define a coreset, the datastructure that we compute in this paper.

▶ **Definition 2** ((k,ϵ)-coreset). *For $k \in \mathbb{N}$ and $\epsilon \in (0,1)$, a (k,ϵ)-coreset for a weighted set (P,w) is a weighted set (Q,v) such that for every $Z \in \mathcal{X}^k$ we have $(1-\epsilon)\mathsf{COST}(P,w,Z) \leq \mathsf{COST}(Q,v,Z) \leq (1+\epsilon)\mathsf{COST}(P,w,Z)$.*

Coresets with possibly negative weight functions have been considered [13]. However, computing approximate solutions on these coresets in polynomial-time remains an open problem, so we restrict our definition to non-negative weight functions to ensure that an approximate solution can be quickly produced [16, 8]. This implies a PTAS for Euclidean space and a polynomial-time $2\tau(1+\epsilon)$-approximation for general metric spaces (where τ is the best polynomial-time approximation factor for the problem in the offline setting). This factor of $2\tau(1+\epsilon)$ is well-known in the literature, see [9, 7, 16] for details.

3 Our Techniques

In this work, we provide an alternative technique for constructing a ρ-metric k-median coreset in the streaming setting. Instead of using the merge-and-reduce tree (see Section 4.1) where each node of the tree uses the offline construction, we perform a single offline construction in the streaming setting. This reduces $O(\log^{a+1} n)$ multiplicative overhead[1] of merge-and-reduce to $O(1)$ overhead.

The offline coreset construction of [6] has the following structure: first, a bicriterion approximation (see Definition 1) is computed over the entire input set P. The bicriterion is used to estimate for each point p its "sensitivity" $s_P(p)$ which, intuitively speaking, measures how important p is relative to the entire set P. Then, $m = m(n,k,\epsilon)$ points are sampled i.i.d. according to the distribution of sensitivities. This suggests a two-pass streaming algorithm with no overhead: in the first pass, construct a bicriterion using an algorithm such as [7]. In the second pass, sample according to sensitivity distribution, computed using the bicriterion found in the first pass. Our contribution is showing how these two passes can be combined into a single-pass.

How do we accomplish both these tasks in parallel? If the sensitivities stayed constant, we could use weighted reservoir sampling to maintain an i.i.d. sample. However, we cannot do this for a changing distribution. The sensitivities may decrease because when a new point is added, all existing points may become less important to the overall stream.

Inductively assume that we have a coreset. Upon receiving the next point, we generate a new bicriterion which we use to update the sensitivities of the points seen so far. The first idea is that instead of sampling m points from a distribution such that point p is sampled with probability $s_P(p)$, we built a set which contains point p with probability $s_P(p)$ as follows: let $u(p)$ be a uniform random number in $[0,1)$, and store p as long as $u(p) < ms_P(p)$ (recall

[1] Recall that a is the exponent in the offline construction's dependence on ϵ^{-1}, and n is the length of the stream.

that $s_P(p)$ decreases as more points are added to P). With high probability, we return a set of $\Theta(m)$ points. The problem is that this set is not an i.i.d sample. To see this, consider the fact that unlike an i.i.d. sample, this set cannot have repeated points.

To solve this problem, we repeat the above process independently in parallel m times, where each process should sample *exactly* 1 point. The first issue is that the total probability $t = \sum_{p \in P} s_P(p)$, which must be scaled to 1, is only known up to a factor of k. Even after solving this, and assuming we have at least m trials that return exactly one point, they do not follow the original distribution. Indeed, the probability of a trial returning only one point p is the probability of drawing point p multiplied the probabilities of not drawing all other points. The rough idea to overcome this is that we distort the probabilities by computing the inverse to this transformation such that the final probabilities follow the desired distribution.

We prove our result for ρ-metric k-median. As stated before, it is well-known that most M-estimators can be recast as a ρ-metric k-median problem for a low value of ρ [14]. See Table 1 for a list of several common M-estimators along with the ρ-metric they satisfy. For many M-estimators, ours is the first coreset result over data streams besides merge-and-reduce.

Table 1 List of several examples to which our result applies. An M-estimator with ψ-function $\psi(x)$ induces a clustering problem with cost $\psi(d(p,c))$ for a point p with nearest center c. The ρ-value is calculated from the ψ-function, making the column redundant but non-trivial to compute.

Estimator	ψ-function	ρ
k-median	x	1
k-means	x^2	2
Huber	$\psi(x) = \begin{cases} \frac{x^2}{2} & \text{if } x < 1 \\ x - \frac{1}{2} & \text{if x} \geq 1 \end{cases}$	2
Cauchy	$\psi(x) = \log(1 + x^2)$	2
Tukey	$\psi(x) = \begin{cases} \frac{1}{6}(1 - (1 - x^2)^3) & \text{if } x < 1 \\ \frac{1}{6} & \text{if x} \geq 1 \end{cases}$	3

4 Related Work

Table 2 summarizes previous work along with our current results. By far, the most widely-studied problems in this class have been the k-median and k-means functions. In general, the extension to arbitrary M-estimators is non-trivial; the first such result was [14]. Our approach naturally lends itself to this extension. M-estimators are highly important for noisy data or data with outliers. As one example, Huber's estimator is widely used in the statistics community [17, 20]. It was written that "this estimator is so satisfactory that it has been recommended for almost all situations" [22]. Our results work not only for Huber's estimator but for all M-estimators, such as the Cauchy and Tukey biweight functions which are also widely-used functions.

Note that in the below table, $\tilde{O}$ notation is used to write in terms of d, ϵ, k, and $\log n$ (therefore hiding factors of $\log \log n$ but not $\log n$).

k-means

In the k-means problem we wish to compute a set k of centers (points) in some metric space, such that the sum of squared distances to the input points is minimized, where each input point is assigned to its nearest center. The corresponding coreset is a positively weighted subset of points that approximates this cost to every given set of k centers. Deterministic

Table 2 Summary of Related Work. Note that Metric M-estimators are the most general, and these results apply to all other categories.

Problem	Streaming Size	Paper
Euclidean k-means	$O(k\epsilon^{-d}\log^{d+2} n)$	[19]
Euclidean k-means	$O(k^3\epsilon^{-(d+1)}\log^{d+2} n)$	[18]
Euclidean k-means	$O(dk^2\epsilon^{-2}\log^8 n)$	[10]
Euclidean k-means	$O(dk\log k\epsilon^{-4}\log^5 n)$	[13]
Euclidean k-means	$\tilde{O}((d/\epsilon)^{O(d)}k\log^{O(d)} n)$	[1]
Metric k-means	$O(\epsilon^{-2}k^2\log^8 n)$	[11]
Metric k-means	$O(\epsilon^{-4}k\log k\log^6 n)$	[13]
Euclidean k-median	$O(dk^2\epsilon^{-2}\log^8 n)$	[10]
Euclidean k-median	$O(k\epsilon^{-d}\log^{d+2} n)$	[19]
Euclidean k-median	$O(k^2\epsilon^{-O(d)}\log^{d+1} n)$	[18]
Euclidean k-median	$O(d\epsilon^{-2}k\log k\log^3 n)$	[13]
Metric k-median	$O(k^2\epsilon^{-2}\log^8 n)$	[10]
Metric k-median	$O(\epsilon^{-2}k\log k\log^4 n)$	[13]
Euclidean M-estimators	$O(\epsilon^{-2}k^{O(k)}d^2\log^5 n)$	[14]
Metric M-estimators	$O(\epsilon^{-2}k^{O(k)}\log^7 n)$	[14]
Metric M-estimators	$O(\epsilon^{-2}k\log k\log n)$	This paper

coresets of size exponential in d were first suggested by Har-Peled and Mazumdar in [19]. The first coreset construction of size polynomial in d was suggested by Ke-Chen in [10] using several sets of uniform sampling. Other high-dimensional results (e.g. [2]) are also known in the streaming setting.

Streaming

The metric results of [10, 13] and Euclidean results of [10, 19, 18, 13] that rely on merge-and-reduce appear in Table 2. In Euclidean space, a more diverse set of stronger results is known. In particular, coreset constructions are known that do not begin with a bicriterion solution, and whose streaming variant does not rely on merge-and-reduce [1]. Sketches have been given in [12] for M-estimators in Euclidean space. With the additional assumption that points lie on a discrete Euclidean grid $\{1,\ldots,\Delta\}^d$, alternative techniques are known for k-means and other problems, even when the stream allows the deletion of points [15].

4.1 Merge and Reduce Tree

We briefly summarize the previous technique for maintaining coresets in the streaming setting due to Har-Peled and Mazumdar [19] and Bentley and Sax [5]. In this method, a merge-and-reduce tree is built by using an offline coreset construction as a blackbox. Previously, this was the only known technique for building a streaming coreset for many metric problems. It relies solely on the following two properties which can be easily verified:

1. Merge: The union of (k,ϵ)-coresets is a (k,ϵ)-coreset.
2. Reduce: A (k,ϵ)-coreset of a (k,ϵ')-coreset is a $(k,\epsilon+\epsilon'+\epsilon\epsilon')$-coreset.

The merge-and-reduce tree works as follows. There are buckets T_i for $i\geq 0$. In each step, the bucket T_0 takes in a segment of $O(1)$ points from the stream. Then the tree works like counting in binary: whenever buckets T_0 to T_{i-1} are full, these i buckets are merged and then reduced by taking a $(k,\frac{\epsilon}{\log n})$-coreset of their union and storing the result in T_i.

Let s be the space of offline construction, which depends on ϵ as ϵ^{-a}. At the end of the stream, $O(\log n)$ buckets have been used and each bucket uses $O(s \log^a n)$ space; this incurs a multiplicative overhead of $\Theta(\log^{a+1} n)$ in the storage requirement. The second factor comes from using the accuracy parameter $\frac{\epsilon}{\log n}$, which is necessary by Property 2 since the construction will be compounded $O(\log n)$ times. Due to this compounding, the runtime is multiplied by a factor of $O(\log n)$.

5 Streaming Algorithm

We present a streaming algorithm to construct a coreset for ρ-metric k-median. Our method combines a streaming bicriterion algorithm [7, 21] and a batch coreset construction [13] to create a streaming coreset algorithm. The space requirements are combined addivitely, therefore ensuring no overhead.

We now state our main theorem. The probability of success $1-\delta$ typically has one of two meanings: that the construction succeeds at the end of the stream (a weaker result), or that the construction succeeds at every intermediate point of the stream (a stronger result). Our theorem gives the stronger result, maintaining a valid coreset at every point of the stream. Our space and time bounds follow the convention that a point can be stored in $O(1)$ space.

▶ **Theorem 3** (Main Theorem). *Let $\epsilon, \delta \in (0,1)$. Given the problem of k-median clustering in a ρ-metric space for $\rho = O(1)$, there exists an insertion-only streaming algorithm that maintains a (k,ϵ)-coreset on a stream of n points while requiring $O(\epsilon^{-2}k(\log k \log n + \log \frac{1}{\delta}))$ space and worst-case update time, and succeeds at every point of the stream with probability at least $1-\delta$.*

In Section 5.1 we introduce the streaming bicriterion algorithm. Then in Section 5.2 we review the offline coreset construction we will be adapting to the streaming setting. We prove in Section 5.3 how to use the bicriterion to bound the importance of points. Finally we present the streaming algorithm in Section 5.4.

5.1 Streaming Bicriterion Algorithm

Let P_i denote the prefix of the stream $\{p_1, \ldots, p_i\}$. The entire stream P is then P_n. Recall that in the streaming setting, we receive each point sequentially in the order $(p_1, p_2, \ldots)$. We use the function $\mathbb{1} : P \to \{1\}$ to map every point to unit weight. For ease of exposition we assume the input set is weighted as $(P, \mathbb{1})$ and that all points of P are distinct. Consider the moment when the first i points have arrived, meaning that the prefix P_i is the current set of arrived points. The algorithm $\mathcal{A}$ of [7] provides an $(O(1), O(\log n))$-approximation of P_i. We now restate their general result, adding several details (such as the outputs L_i and π_i) that were merely internal details for them but will be crucial for us.

▶ **Theorem 4** ([7], restated). *Let $\alpha, \gamma > 1$ be absolute constants. Define $B_0 = \emptyset$. Let $(p_1, \ldots, p_n)$ be a stream of at most n points in a ρ-metric space for $\rho = O(1)$. Let $n, k \geq 1$, $\delta \in (0,1)$ be input parameters. Upon receiving point p_i, algorithm $\mathcal{A}(k, n, \delta)$ returns a weighted set (B_i, w_i), a value $L_i > 0$, and a map $\pi_i : B_{i-1} \cup \{p_i\} \to B_i$. Define $f_i(p_j) = \pi_i(\pi_{i-1}(\ldots \pi_j(p_j)\ldots))$. For any integer $i \in [n]$ we have with probability at least $1-\delta$ that the following three statements hold.*

1. $L_i \leq \mathsf{OPT}_k(P_i)$
2. $\sum_{p \in P_i} D(p, f_i(p)) \leq \alpha L_i$
3. *(B_i, w_i) along with the map $f_i : P_i \to B_i$ is an $(\alpha, \gamma(\log n + \log \frac{1}{\delta}))$-approximation for $(P, \mathbb{1})$.*

The algorithm requires $O(\gamma(\log n + \log \frac{1}{\delta}))$ space and update time.

Algorithm $\mathcal{A}$ reduces the number of distinct points by combining nearby points into a single point of higher weight.

5.2 Offline Coreset Construction

In the offline coreset construction of [6], the sensitivity of a point $p \in P$ in a ρ-metric space $(\mathcal{X}, D)$ is defined as:

$$s_P(p) = \max_{Z \in \mathcal{X}^k} \frac{D(p,Z)}{\sum_{q \in P} D(q,Z)}$$

Notice that $0 \leq s_P(p) \leq 1$ for every point $p \in P$, and that the sensitivity of a point p is relative to the set P it belongs to. When context is clear, we omit the subscript and write $s(p)$. Computing $s(p)$ may be difficult, but we can give an upper bound $s'(p) \in [s(p), 1]$. Define the total sensitivity $t' = \sum_{p \in P} s'(p)$. We will apply the following theorem:

▶ **Theorem 5** (proven in [6]). *Let P be a set of n points, and define $s' : P \to [0,1]$ and t' as above. Let $\delta, \epsilon \in (0,1)$ be input parameters. Consider a distribution $\mathcal{S}$ supported on P where p has weight $s'(p)/t'$. Define $m' = \lceil 3t'\epsilon^{-2}(\log n \log t' + \log(1/\delta))\rceil$. Let Q be an i.i.d. sample of at least m' points from $\mathcal{S}$. Define a weight function $v : Q \to [0,\infty)$ as $v(q) = (|Q|s'(q))^{-1}$. With probability at least $1-\delta$, (Q,v) is a (k,ϵ)-coreset for P.*

One may be skeptical why only an upper bound is necessary, wondering why not simply set $s'(p) = 1$. This does indeed work, but has the undesirable effect of setting $t' = n$ and therefore results in a coreset of $\Omega(n)$ points. More generally, the coreset is useless if t' is large since the size of Q may be comparable to the size of P. In the next subsection, we show how to bound $t' = O(\rho^2 k)$. Observe that neither k nor ρ appear explicitly in the sample size, but they both appear implicitly through the value of t'.

5.3 Bounding the Sensitivity

Let the map $p \mapsto p'$ be an (σ,λ)-approximation of P for some constants σ and λ. Define $P(p) = \{q \in P : q' = p'\}$ to be the cluster containing p.

▶ **Lemma 6.** *Let the map $p \mapsto p'$ define an (σ,λ)-approximation for the k-median clustering of P. For every point $p \in P$:*

$$s(p) \leq \frac{\rho\sigma D(p,p')}{\sum_{q \in P} D(q,q')} + \frac{\rho^2(\sigma+1)}{|P(p)|}$$

Proof. For an arbitrary $Z \in \mathcal{X}^k$ we need to provide a uniform bound for

$$\begin{aligned} \frac{D(p,Z)}{\sum_{q \in P} D(q,Z)} &\leq \frac{\rho D(p,p')}{\sum_{q \in P} D(q,Z)} + \frac{\rho D(p',Z)}{\sum_{q \in P} D(q,Z)} \\ &\leq \frac{\sigma\rho D(p,p')}{\sum_{q \in P} D(q,q')} + \frac{\rho D(p',Z)}{\sum_{q \in P} D(q,Z)} \end{aligned} \tag{1}$$

where the second inequality holds because $\sum_{q \in P} D(q,q') \leq \sigma \mathsf{OPT}(P) \leq \sigma \sum_{q \in P} D(q,Z)$. To

bound the last term, recall that $q' = p'$ for all $q \in P(p)$ so:

$$\begin{aligned} D(p', Z)|P(p)| &= \sum_{q \in P(p)} D(p', Z) = \sum_{q \in P(p)} D(q', Z) \\ &\leq \rho \sum_{q \in P(p)} (D(q', q) + D(q, Z)) \\ &\leq \rho \sum_{q \in P} D(q', q) + \rho \sum_{q \in P(p)} D(q, Z) \\ &\leq \rho\sigma \sum_{q \in P} D(q, Z) + \rho \sum_{q \in P(p)} D(q, Z) \\ &\leq \rho(\sigma + 1) \sum_{q \in P} D(q, Z) \end{aligned}$$

Dividing by $|P(p)| \sum_{q \in P} D(q, Z)$ gives

$$\frac{D(p', Z)}{\sum_{q \in P} D(q, Z)} \leq \frac{\rho(\sigma + 1)}{|P(p)|}$$

Substituting this in (1) yields the desired result. ◀

We will use Lemma 6 to define our upper bound $s'(p)$. An immediate but extremely important consequence of Lemma 6 is that $t' = \sum_{p \in P} s'(p) = \rho\sigma + \rho^2(\sigma + 1)\lambda k$. This can be seen by directly summing the formula given by the lemma.

5.4 Streaming Algorithm

Consider the prefix P_i which is the input after the first i points have arrived. For ease of notation, we write $s'_i(p)$ to refer to $s'_{P_i}(p)$, an upper bound on the sensitivity of p with respect to the first i points. After processing first i points of the stream, Algorithm 1 constructs a set (Q_i, v_i). With probability at least $1 - \delta$, (Q_i, v_i) is a (k, ϵ)-coreset for P_i for every $i \in [n]$. This algorithm will satisfy the claims of Theorem 3.

5.4.1 Overview of the algorithm

The algorithm initializes on Lines 1-9. The main loop of Lines 10-38 processes the stream, accepting one point per iteration. For each iteration, the first step is to process the point with algorithm $\mathcal{A}$ (Line 11) then compute a $(O(1), O(1))$-bicriterion approximation on its output (Line 18). On Line 23 we use this to maintain an upper bound $s'_i(p)$ on the sensitivity of a point p with respect to P_i. In the analysis, let $t'_i = \sum_{\ell=1}^{i} s'_i(p_\ell)$ be the upper bound on the total sensitivity of P_i. We now define $\mathcal{S}_i$, the distribution from which we will draw our sample. Note that $\mathcal{S}_i$ is non-deterministic, since the values of $s'_i(p)$ (and therefore t'_i) depend on the randomness of Algorithm $\mathcal{A}$ as well as any possible randomness used in the bicriterion approximation.

▶ **Definition 7.** *The probability distribution $\mathcal{S}_i$ is supported on P_i and assigns probability $s'_i(p)/t'_i$ to point p.*

By Theorem 5, it suffices to sample $m'_i = \lceil 3t'_i\epsilon^{-2}(\log n \log t'_i + \log(n/\delta)) \rceil$ points i.i.d. from $\mathcal{S}_i$ to construct a coreset for $(P_i, \mathbb{1})$ with probability at least $1 - \delta/n$. On Line 2 we define $t_\circ$ to upper bound the maximal value of t'_i over any set of points in $\mathcal{X}$. Likewise on Line 3 we set $m_\circ = \lceil 3t_\circ\epsilon^{-2}(\log n \log t_\circ + \log(n/\delta)) \rceil$ which is an upper bound on the required sample size.

We maintain $\Theta(k(\log n \log k + \log \frac{n}{\delta}))$ sets M_y, each containing a sample of the stream. We hope that many of these samples contain only a single point, because if so then by Lemma 6 the point follows the distribution $\mathcal{S}_i$. With high probability, at least m'_i sets will be singletons, and so we take their union to construct the coreset (Q_i, v_i).

Memory considerations

Due to memory constraints, we only store the maps g_i, f_i, u_y, z_i, and s'_i for points in $\cup_{y \in Y} M_y$. In the analysis only, we use f_i, z_i, and $s'_i(p)$ for points that have been deleted. This refers to the value that would have been set if we continued executing Lines 21-23 for p.

5.4.2 Proof of correctness

Using Algorithm $\mathcal{A}$ we obtain an $(\alpha, \gamma(\log n + \log \frac{n}{\delta}))$-approximation (B_i, w_i) with the map $\pi_i : B_{i-1} \cup \{p_i\} \to B_i$ which we use to obtain the map $f_i : P_i \to B_i$. Line 23 runs an offline (γ, λ)-approximation algorithm on B_i, and we obtain weighted set (C_i, v_i). The following lemma shows that (C_i, v_i) is a (σ, λ)-approximation for P_i, where $\sigma = \rho\alpha + 2\rho^2\gamma(\alpha + 1)$ as defined on Line 1. The clustering map will be $g_i \circ f_i : P_i \to B_i \to C_i$.

▶ **Lemma 8.** *Assume that algorithm $\mathcal{A}$ has not failed. Then (C_i, w_i^C) with $g_i \circ f_i : P_i \to C_i$ is a (σ, λ)-approximation of P_i.*

Proof. As the context is clear, we drop the subscript i. By Theorem 4, (B, w^B) with $f : P \to B$ is an (α, β)-approximation of P where $\beta = \gamma(\log n + \log \frac{n}{\delta})$. Also, (C, w^c) with $g : B \to C$ is the (γ, λ)-approximation of B. In the following, all sums will be taken over all $p \in P$. The hypotheses state that $\sum D(p, f(p)) \leq \alpha \mathsf{OPT}_k(P)$ and $\sum D(f(p), g(f(p))) \leq \gamma \mathsf{OPT}_k(B)$. Let P^* be an optimal clustering of P, that is $\sum D(p, P^*) = \mathsf{OPT}_k(P)$. Then $\frac{1}{2}\mathsf{OPT}_k(B) \leq \sum D(f(p), P^*) \leq \rho \sum (D(f(p), p) + D(p, P^*)) \leq \rho(\alpha+1)\mathsf{OPT}(P)$. The factor of $\frac{1}{2}$ comes from the fact that $\mathsf{OPT}(B)$ is defined using centers restricted to B (see [16] for details). We now write $\sum D(p, g(f(p))) \leq \rho \sum (D(p, f(p)) + D(f(p), g(f(p)))) \leq (\rho\alpha + 2\rho^2\gamma(\alpha + 1))\mathsf{OPT}_k(P)$ as desired. ◀

To use Lemma 6 to determine $s'_i(p)$, we will compute the cluster sizes $|P_i(p)|$ and estimate the clustering cost $\sum_{q \in P} D(q, q')$ by L_i. We must bound the clustering cost from below because we require $s'_i(p)$ to be an upper-bound of $s_i(p)$.

▶ **Lemma 9.** *Assume that algorithm $\mathcal{A}$ has not failed and that (Q_{i-1}, v_{i-1}) is a (k, ϵ)-coreset for P_{i-1}. Then $z_i(p) \geq s_i(p)$ for every $p \in P_i$.*

Proof. For the first claim, consider Lemma 6 applied with the clustering map $g_i \circ f_i : P_i \to C_i$. We write $p' = g_i \circ f_i(p)$. By Lemma 8, this map is a (σ, λ)-approximation of P_i. Observe that $|P_i(p)|$ (from Lemma 6) is precisely $w_i^C(g_i \circ f_i(p))$ since the weight of a point c is determined by how many points in P_i were clustered to c. By Theorem 4, $L_i \leq \mathsf{OPT}_k(P_i) \leq \mathsf{COST}(P_i, \mathbb{1}, C_i)$. We may then write:

$$\begin{aligned} z_i(r) &= \frac{\rho\sigma D(r, g_i \circ f_i(r))}{L_i} + \frac{\rho^2(\sigma+1)}{w_i^C(g_i \circ f_i(r))} \\ &\geq \frac{\rho\sigma D(r, r')}{\sum_{p \in P_i} D(p, p')} + \frac{\rho^2(\sigma+1)}{|P_i(r)|} \\ &\geq s_i(r) \end{aligned}$$

where the last inequality follows by Lemma 6. ◀

Algorithm 1 Input: parameters $\epsilon, \delta \in (0,1)$ and $n, k \in \mathbb{N}$. A stream of n points in a ρ-metric space. Notes: $\mathcal{A}(\cdot,\cdot,\cdot)$ denotes the blackbox algorithm from Theorem 4 along with its universal constants α and γ. Line 18 uses any RAM-model $(O(1), O(1))$-approximation such as [3].

1: $\sigma \leftarrow \rho\alpha + 2\rho^2\gamma(\alpha+1)$
2: $t_\circ \leftarrow \rho\sigma\alpha + \rho^2(\sigma+1)\lambda k$
3: $m_\circ \leftarrow \lceil 3t_\circ\epsilon^{-2}(\log n \log t_\circ + \log(n/\delta))\rceil$
4: $Q_0 \leftarrow \emptyset$
5: Initialize $\mathcal{A}(k, n, \delta/n)$
6: $Y \leftarrow \{1, \ldots, 8m_\circ\}$
7: **for** each $y \in Y$ **do**
8: $\quad M_y \leftarrow \emptyset$
9: **end for**
10: **for** the next point p_i from the stream **do**
11: $\quad (B_i, w_i^B, \pi_i, L_i) \leftarrow$ update $\mathcal{A}$ with point p_i
12: $\quad$ **for** each $y \in Y$ **do**
13: $\quad\quad u_y(p_i) \leftarrow$ uniform random number from $[0,1)$
14: $\quad\quad M_y \leftarrow M_y \cup \{p_i\}$
15: $\quad$ **end for**
16: $\quad f_{i-1}(p_i) \leftarrow p_i$
17: $\quad s'_{i-1}(p_i) \leftarrow 1$
18: $\quad (C_i, w_i^C, g_i) \leftarrow (\gamma,\lambda)$-approximation of (B_i, w_i^B)
19: $\quad R \leftarrow \cup_y M_y$
20: $\quad$ **for** each $r \in R$ **do**
21: $\quad\quad f_i(r) \leftarrow \pi_i \circ f_{i-1}(r)$
22: $\quad\quad z_i(r) \leftarrow \frac{\rho\sigma D(r, g_i \circ f_i(r))}{L_i} + \frac{\rho^2(\sigma+1)}{w_i^C(g_i \circ f_i(r))}$
23: $\quad\quad s'_i(r) \leftarrow \min(s'_{i-1}(r), z_i(r))$
24: $\quad$ **end for**
25: $\quad$ **for** each $y \in Y$ **do**
26: $\quad\quad$ **for** each $q \in M_y$ **do**
27: $\quad\quad\quad$ **if** $u_y(q) > \frac{s'_i(q)}{s'_i(q)+t_\circ}$ **then**
28: $\quad\quad\quad\quad$ Delete q from M_y
29: $\quad\quad\quad$ **end if**
30: $\quad\quad$ **end for**
31: $\quad$ **end for**
32: $\quad \Gamma_i \leftarrow \{y \in Y : |M_y| = 1\}$
33: $\quad Q_i \leftarrow \cup_{y \in \Gamma_i} M_y$
34: $\quad$ **for** each $q \in Q_i$ **do**
35: $\quad\quad v_i(q) \leftarrow (|\Gamma_i| s'_i(q))^{-1}$
36: $\quad$ **end for**
37: $\quad$ **return** (Q_i, v_i)
38: **end for**

The implication is that our upper bound on sensitivity is valid, as we now prove formally:

▶ **Lemma 10.** *Assume that algorithm $\mathcal{A}$ has not failed and that (Q_{i-1}, v_{i-1}) is a (k,ϵ)-coreset for P_{i-1}. Then $s'_i(p) \geq s_i(p)$ for every $p \in P_i$.*

Proof. Fix a point $p \in P_i$. We see from Line 23 that $s'_i(p) = z_j(p)$ for some $j \leq i$. Lemma 11 shows that $z_j(p) \geq s_j(p)$. Observe directly from the definition of sensitivity that $s_i(p) \leq s_j(p)$ for any $j \leq i$. Combining these shows that $s'_i(p) = z_j(p) \geq s_j(p) \geq s_i(p)$. We conclude that $s'_i(p) \geq s_i(p)$ for all $p \in P_i$. ◀

For each point p, the value of $s'_i(p)$ is non-increasing in i. This is because $s'_i(p)$ is defined as the minimum of itself and a new value on Line 23. It follows from the monotonicity of $f(x) = \frac{x}{x+1}$ that $\frac{s'_i(r)}{s'_i(r)+t_\circ}$ is also non-increasing in i. Therefore once the deletion condition on Line 27 becomes satisfied, it remains satisfied forever. This is essential because after deleting a point from memory, it can never be retrieved again. We can characterize $M_y^{(i)}$ without reference to the streaming setting: $M_y^{(i)} = \{p \in P_i : u_y(p) \leq \frac{s'_i(p)}{s'_i(p)+t_\circ}\}$. This has the important implication that $Pr(p \in M_y^{(i)}) = \frac{s'_i(p)}{s'_i(p)+t_\circ}$.

▶ **Lemma 11.** *Assume that algorithm $\mathcal{A}$ has not failed and that (Q_{i-1}, v_{i-1}) is a (k,ϵ)-coreset for P_{i-1}. Then $\sum_{p\in P_i} z_i(p) < t_\circ$.*

Proof. The value of $t_\circ$ is defined on Line 2 as $\rho\sigma\alpha + \rho^2(\sigma+1)\lambda k$.

$$\begin{aligned}\sum_{p\in P_i} z_i(p) &= \sum_{p\in P_i} \frac{\rho\sigma D(p, g_i \circ f_i(p))}{L_i} + \frac{\rho^2(\sigma+1)}{w_i^C(g_i \circ f_i(p))} \\ &\leq \frac{\rho\sigma\alpha L_i}{L_i} + \rho^2(\sigma+1)\lambda k \\ &\leq \rho\sigma\alpha + \rho^2(\sigma+1)\lambda k \\ &= t_\circ\end{aligned}$$

where the first inequality comes from Lemma 8 and Theorem 4. Note that we have summed the second term using the fact that a center $c \in C_i$ with weight $w_i^C(c)$ has exactly $w_i^C(c)$ points of P_i clustered to it. Therefore we may re-write the sum:

$$\sum_{p\in P_i} \frac{1}{w_i^C(g_i \circ f_i(p))} = \sum_{c\in C_i} 1 = \lambda k$$ ◀

To construct a coreset by sampling from $\mathcal{S}_i$, the algorithm take the union of those M_y for $y \in Y$ that are singletons. Any set M_y that is either empty or contains more than one point will be ignored, but still kept track of since it may later become a singleton. We now show that if a sample M_y contains a single point, then it follows the distribution $\mathcal{S}_i$. Let $M_y^{(i)}$ denote the state of M_y after the prefix P_i has been processed, and let $Pr(A : B)$ denote the probability of event A conditioned on event B.

▶ **Lemma 12.** *For any $p \in P_i$, $Pr(M_y^{(i)} = \{p\} : |M_y^{(i)}| = 1) = s'_i(p)/\sum_{\ell=1}^{i} s'_i(p_\ell)$.*

Proof. Define $\alpha_\ell = \frac{s'_i(p_\ell)}{s'_i(p_\ell)+t_\circ}$ and $\Psi = \prod_{\ell=1}^{i}(1-\alpha_\ell)$. For $z \in [i]$ let E_z denote the event that $M_y^{(i)} = \{p_z\}$. The probability that the sampler contains only p_z means that it failed to sample all p_ℓ for $\ell \neq z$, meaning that $Pr(E_z) = \alpha_z\Psi/(1-\alpha_z) = s'_i(p_z)\Psi/t_\circ$. The result is obtained since:

$$\begin{aligned} Pr(E_z : |M_y^{(i)}| = 1) &= Pr(E_z)/Pr(|M_y^{(i)}| = 1) \\ &= Pr(E_z)/Pr(\cup_{\ell=1}^{i} E_\ell) \\ &= Pr(E_z)/\sum_{\ell=1}^{i} Pr(E_\ell) \\ &= s_i'(p_z)/\sum_{\ell=1}^{i} s_i'(p_\ell) \end{aligned}$$ ◀

The significance of Lemma 12 is tantamount. If a sample M_y contains a singleton, then it is equivalent to a random draw from $\mathcal{S}_i$. Therefore if at least m_i' samplers contain a singleton, taking their union gives us an i.i.d. sample of at least m_i' points from $\mathcal{S}_i$. This is precisely what we need to construct a coreset using Theorem 5. However, it remains to show that we will have enough singleton samplers with high probability. The next lemma begins to establish this fact.

▶ **Lemma 13.** *Fix any $y \in Y$ and $i \in [n]$. Then $P(|M_y^{(i)}| = 1) \geq \frac{t'}{4t_\circ}$.*

Proof. Let p_i be the most recently arrived point, and define $\alpha_\ell = \frac{s_i'(p_\ell)}{s_i'(p_\ell)+t_\circ}$. Observe that $s_i'(p_\ell) \geq 0$ implies $\alpha_\ell \leq s_i'(p_\ell)/t_\circ$. It follows from Line 27 that $Pr(p_\ell \in M_y) = \alpha_\ell \leq s_i'(p_\ell)/t_\circ$. The expected value of $|M_y|$ is therefore at most $\sum_{\ell=1}^{i} \frac{1}{t_\circ} s_i'(p_\ell) = \frac{t'}{t_\circ}$. Markov's inequality yields $Pr(|M_y| \geq 2) = Pr(|M_y| \geq \frac{2t_\circ}{t'}\frac{t'}{t_\circ}) \leq \frac{t'}{2t_\circ} \leq \frac{1}{2}$.

$$\begin{aligned} P(|M_y| = 1) &= \sum_{\ell=1}^{i} P(|M_y| = \{p_\ell\}) \\ &= \sum_{\ell=1}^{i} \alpha_\ell \prod_{z \neq \ell} (1 - \alpha_z) \\ &= \sum_{\ell=1}^{i} \frac{\alpha_\ell}{1-\alpha_\ell} \prod_{z=1}^{i} (1 - \alpha_z) \\ &= \frac{1}{t_\circ} \left(\sum_{\ell=1}^{i} s_i'(p_\ell) \right) \left(\prod_{z=1}^{i} (1 - \alpha_z) \right) \\ &= \frac{t'}{t_\circ} Pr(|M_y| = 0) \end{aligned}$$

We know that $Pr(|M_y| = 0) + Pr(|M_y| = 1) + Pr(|M_y| \geq 2) = 1$, so substitution gives $(\frac{t_\circ}{t'} + 1)Pr(|M_y| = 1) + \frac{1}{2} \geq 1$. Rearranging this, we obtain $Pr(|M_y| = 1) \geq \frac{1}{2}\frac{1}{t_\circ/t'+1} \geq \frac{t'}{4t_\circ}$ as desired. ◀

Now that we have provided a lower bound on the probability of any sample M_y being a singleton, we move on to lower bound the probability of at least m_i' of the samples $\{M_y\}_{y \in Y}$ being singletons. Observe that the $\{M_y\}$ are entirely independent. We use a Chernoff bound to lower bound the size of Γ_i, defined on Line 32, which is the set of all singleton samples.

▶ **Lemma 14.** *$|\Gamma_i| \geq m_i'$ with probability at least $1 - \delta/n$.*

Proof. We see directly from Line 32 that $|\Gamma_i|$ is a sum of $|Y|$ independent Bernoulli trials, each which succeeds with probability at least $\frac{t'}{4t_\circ}$ by Lemma 13. By a Chernoff bound, $Pr(|S| \leq \frac{1}{2}|Y|\frac{t'}{4t_\circ}) \leq e^{-|Y|t'/32t_\circ}$. We note that $|Y|t'/32t_\circ = 8m_\circ t'/32t_\circ \geq \ln(n/\delta)$. Plugging this into the Chernoff bound yields that $|S| \geq m_\circ t'/t_\circ$ with probability at least $1 - \delta/n$. We conclude by noting that $m_\circ t'/t_\circ = 3t'(\log n \log t_\circ + \log \frac{\delta}{n}) \geq m'$. ◀

We now have all the tools to proceed with the proof of Theorem 1. We begin with the space requirement, then prove correctness.

▶ **Lemma 15.** *After processing P_i, Algorithm 1 stores $O(\epsilon^{-2}k(\log k \log n + \log \frac{n}{\delta}))$ points with probability at least $1 - \delta/n$.*

Proof. First note that $O(\log n + \log \frac{n}{\delta}) = O(\log n + \log \frac{1}{\delta})$. By Theorem 4 we know that $\mathcal{A}$ stores $O(k(\log n + \log \frac{n}{\delta}))$ points deterministically. In addition to the blackbox $\mathcal{A}$, the algorithm stores the sets M_y along with a constant amount of satellite data per point.

We showed that $E[|M_y|] \leq t'/t_\circ$ in the proof of Lemma 13. Directly from Lemma 13, we lower bound $E[|M_y|] \geq Pr(|M_y| = 1) \geq t'/4t_\circ$. Combining these upper and lower bounds permits us to write $2m_\circ t'/t_\circ \leq E[\sum_{y \in Y} |M_y|] \leq 8m_\circ t'/t_\circ$.

A Chernoff bound can be applied for a high-probability guarantee. The random variable $X = \sum_{y \in Y} |M_y|$ is a sum of $|Y|i$ independent Bernoulli trials, each event being $p_\ell \in M_y$ for some $1 \leq \ell \leq i$ and $y \in Y$. We have the Chernoff bound that $\text{Prob}[X \geq (1+\eta)\mu] \leq e^{-\eta^2\mu/(2+\eta)}$ for any $\eta \geq 1$ where $\mu = E[X]$. Using $\eta = 1$, this yields that $X < 16m_\circ$ with probability at least $1 - e^{-2m_\circ t'/3t_\circ} < e^{-2m'/3} < \delta/n$. ◀

We now prove correctness, using the following lemma as a tool for our final claim.

▶ **Lemma 16.** *Assume that algorithm $\mathcal{A}$ has not failed and that (Q_{i-1}, v_{i-1}) is a (k, ϵ)-coreset for P_{i-1}. Then (Q_i, v_i) is a (k, ϵ) coreset of $(P_i, \mathbb{1})$ with probability at least $1 - 3\delta/n$.*

Proof. Lemma 10 shows that $\mathcal{S}_i$ meets the criteria of Theorem 5 with probability at least $1 - \delta/n$. Lemmas 12 and 14 show that Q_i is an i.i.d. sample of at least m'_i points from $\mathcal{S}_i$ with probability at least $1 - \delta/n$. By Theorem 5, conditioning on the success of the previous two statements, (Q_i, v_i) is a (k, ϵ)-coreset for $(P_i, \mathbb{1})$ with probability at least $1 - \delta/n$. We arrive at the desired result by applying the union bound. ◀

We complete the proof of Theorem 3 by induction over each prefix P_i on the following events: (1) the success of $\mathcal{A}$; (2) that (Q_i, v_i) is a (k, ϵ) coreset of $(P_i, \mathbb{1})$; and (3) the storage requirement holding from Lemma 15. The base cases hold trivially.

By Theorem 4, Algorithm $\mathcal{A}(n, k, \delta/n)$ will succeed on P_i with probability at least $1 - \delta/n$. By Lemma 15, the space requirement is maintained with probability at least $1 - \delta/n$. By Lemma 16, (Q_i, v_i) is a (k, ϵ) coreset of $(P_i, \mathbb{1})$ with probability at least $1 - 3\delta/n$. Combining these pieces, we succeed inductively after processing a single point with probability at least $1 - 5\delta/n$. Therefore we succeed at every step of the entire stream with probability at least $1 - 5\delta$. Theorem 3 follows by scaling δ appropriately.

References

1 Marcel R. Ackermann, Marcus Märtens, Christoph Raupach, Kamil Swierkot, Christiane Lammersen, and Christian Sohler. StreamKM++: A Clustering Algorithm for Data Streams. *J. Exp. Algorithmics*, 17:2.4:2.1–2.4:2.30, May 2012. doi:10.1145/2133803.2184450.

2 Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A Framework for Projected Clustering of High Dimensional Data Streams. In Mario A. Nascimento, M. Tamer Özsu, Donald Kossmann, Renée J. Miller, José A. Blakeley, and K. Bernhard Schiefer, editors, *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, August 31 - September 3 2004*, pages 852–863. Morgan Kaufmann, 2004. URL: http://www.vldb.org/conf/2004/RS21P7.PDF, doi:10.1016/B978-012088469-8.50075-9.

3 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local Search Heuristic for K-median and Facility Location Problems. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 21–29, New York, NY, USA, 2001. ACM. doi:10.1145/380752.380755.

4 Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The Hardness of Approximation of Euclidean k-Means. In *31st International Symposium on Computational Geometry, SoCG 2015, June 22-25, 2015, Eindhoven, The Netherlands*, pages 754–767, 2015.

5 Jon Louis Bentley and James B Saxe. Decomposable searching problems I. Static-to-dynamic transformation. *Journal of Algorithms*, 1(4):301–358, 1980.

6 Vladimir Braverman, Dan Feldman, and Harry Lang. New Frameworks for Offline and Streaming Coreset Constructions. *CoRR*, abs/1612.00889, 2016. arXiv:1612.00889.

7 Vladimir Braverman, Adam Meyerson, Rafail Ostrovsky, Alan Roytman, Michael Shindler, and Brian Tagiku. Streaming K-means on Well-clusterable Data. In *Proceedings of the Twenty-second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 26–40. SIAM, 2011. URL: http://dl.acm.org/citation.cfm?id=2133036.2133039.

8 Jaroslaw Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An Improved Approximation for K-median, and Positive Correlation in Budgeted Optimization. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 737–756. SIAM, 2015. URL: http://dl.acm.org/citation.cfm?id=2722129.2722179.

9 Moses Charikar, Liadan O'Callaghan, and Rina Panigrahy. Better Streaming Algorithms for Clustering Problems. In *Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 30–39, New York, NY, USA, 2003. ACM. doi:10.1145/780542.780548.

10 Ke Chen. On coresets for k-median and k-means clustering in metric and euclidean spaces and their applications. *SIAM Journal on Computing*, 39(3):923–947, 2009.

11 Ke Chen. On Coresets for K-Median and K-Means Clustering in Metric and Euclidean Spaces and Their Applications. *SIAM J. Comput.*, 39(3):923–947, August 2009. doi:10.1137/070699007.

12 Kenneth L. Clarkson and David P. Woodruff. Sketching for M-estimators: A Unified Approach to Robust Regression. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 921–939, Philadelphia, PA, USA, 2015. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2722129.2722192.

13 Dan Feldman and Michael Langberg. A Unified Framework for Approximating and Clustering Data. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 569–578, New York, NY, USA, 2011. ACM. doi:10.1145/1993636.1993712.

14 Dan Feldman and Leonard J Schulman. Data reduction for weighted and outlier-resistant clustering. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1343–1354. SIAM, 2012.

15 G. Frahling and C. Sohler. Coresets in dynamic geometric data streams. In *Proc. 37th Annu. ACM Symp. on Theory of Computing (STOC)*, pages 209–217, 2005.

16 Sudipto Guha, Adam Meyerson, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering Data Streams: Theory and Practice. *IEEE Trans. on Knowl. and Data Eng.*, 15(3):515–528, March 2003. doi:10.1109/TKDE.2003.1198387.

17 Frank Hampel, Christian Hennig, and Elvezio Ronchetti. A smoothing principle for the Huber and other location M-estimators. *Computational Statistics & Data Analysis*, 55(1):324–337, 2011. doi:10.1016/j.csda.2010.05.001.

18 S. Har-Peled and A. Kushal. Smaller coresets for k-median and k-means clustering. *Discrete Comput. Geom.*, 37(1):3–19, 2007. doi:10.1007/s00454-006-1271-x.

19 S. Har-Peled and S. Mazumdar. On coresets for k-means and k-median clustering. In *STOC*, 2004.

20 P. J. Huber. Robust Statistics. *Wiley*, 1981.

21 Harry Lang. Online Facility Location Against a t-bounded Adversary. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1002–1014, Philadelphia, PA, USA, 2018. Society for Industrial and Applied Mathematics.

22 Z. Zhang. M-estimators. *http://research.microsoft.com/en-us/um/people/zhang/INRIA/Publis/Tutorial-Estim/node20.html*, [accessed July 2011].

Pairwise Independent Random Walks Can Be Slightly Unbounded

Shyam Narayanan
Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
shyam.s.narayanan@gmail.com

Abstract

A family of problems that have been studied in the context of various streaming algorithms are generalizations of the fact that the expected maximum distance of a 4-wise independent random walk on a line over n steps is $O(\sqrt{n})$. For small values of k, there exist k-wise independent random walks that can be stored in much less space than storing n random bits, so these properties are often useful for lowering space bounds. In this paper, we show that for all of these examples, 4-wise independence is required by demonstrating a pairwise independent random walk with steps uniform in ± 1 and expected maximum distance $\Omega(\sqrt{n} \lg n)$ from the origin. We also show that this bound is tight for the first and second moment, i.e. the expected maximum square distance of a 2-wise independent random walk is always $O(n \lg^2 n)$. Also, for any even $k \geq 4$, we show that the kth moment of the maximum distance of any k-wise independent random walk is $O(n^{k/2})$. The previous two results generalize to random walks tracking insertion-only streams, and provide higher moment bounds than currently known. We also prove a generalization of Kolmogorov's maximal inequality by showing an asymptotically equivalent statement that requires only 4-wise independent random variables with bounded second moments, which also generalizes a result of Błasiok.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains; Mathematics of computing → Probabilistic algorithms

Keywords and phrases k-wise Independence, Random Walks, Moments, Chaining

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.63

Category RANDOM

Related Version `https://arxiv.org/abs/1807.04910`

Funding This research was funded by the PRISE program at Harvard University and by Harvard's Herchel Smith Fellowship.

Acknowledgements I would like to thank Prof. Jelani Nelson for advising this work, as well as for problem suggestions, forwarding me many papers from the literature, and providing helpful feedback on my writeup.

1 Introduction

Random walks are well-studied stochastic processes with numerous applications in physics [14], math [17], computer science [2], economics [13], and biology [4]. A commonly studied random walk on $\mathbb{Z}$ is a process that starts at 0 and at each step independently moves either $+1$ or -1 with equal probability. In this paper, we do not study this random walk but instead study k-wise independent random walks, meaning that steps are not totally independent but that any k steps are completely independent. In many low-space randomized algorithms, information is tracked with processes similar to random walks, but simulating a totally random walk of n steps is known to require $O(n)$ bits while there exist k-wise independent families which can be simulated with $O(k \lg n)$ bits [10]. As a result, understanding properties of k-wise independent random walks have applications to streaming algorithms, such as heavy-hitters [8, 9], distinct elements [5], and ℓ_p tracking [6].


© Shyam Narayanan;
licensed under Creative Commons License CC-BY

For any k-wise independent random walk, where $k \geq 2$, it is well-known that after n steps, the expected squared distance from the origin is exactly n, since $\mathbb{E}_{h\in\mathcal{H}}(h(1)+\cdots+h(n))^2 = n$ for any 2-wise independent hash family $\mathcal{H}$. One can see this by expanding and applying linearity of expectation. This property provides good bounds for the distribution of the final position of a 2-wise independent random walk. However, we study the problem of bounding the position throughout the random walk, by providing comparable moment bounds for $\sup_{1\leq i\leq n} |h(1)+\cdots+h(i)|$ rather than just for $|h(1)+\cdots+h(n)|$ and determining an example of a 2-wise independent random walk where the expected bounds do not hold, even though very strong bounds for even 4-wise independent random walks can be established.

Two more general questions that have been studied in the context of certain streaming algorithms are random walks corresponding to insertion-only streams, and random walks with step sizes corresponding to random variables. These are useful generalizations as the first proves useful in certain algorithms with insertion stream inputs, and the second allows for a setup similar to Kolmogorov's inequality [16], which we will generalize to 4-wise independent random variables. To understand these two generalizations, consider a k-wise independent family of random variables $X_1, \ldots, X_n$ and an insertion stream $p_1, \ldots, p_m \in [n]$, where now seeing p_j means that our random walk moves by X_{p_j} on the jth step. The insertion stream can be thought of as keeping track of a vector z in $\mathbb{R}^n$ where seeing p_j increments the p_jth component of z by 1, and $\vec{X}$ can be thought of as a vector in $\mathbb{R}^n$ with ith component X_i. Then, one goal is to bound for appropriate values of k'

$$\mathbb{E}_{h\in\mathcal{H}}\left[\sup_{1\leq t\leq m}\left|\langle\vec{X}, z^{(t)}\rangle\right|^{k'}\right],$$

where $z^{(t)}$ is the vector z after seeing only the first t elements of the insertion stream. Notice that bounding the k'th moment of the furthest distance from the origin in a k-wise independent random walk is the special case of $m = n$, $p_j = j$ for all $1 \leq j \leq n$, and the X_i's are uniform random signs.

1.1 Main Results

Intuitively, even in a pairwise independent random walk, since the positions at various times have strong correlations with each other, the expectation of the furthest we ever get from the origin should not be much more than the expectation of than our distance from the origin after n steps. But surprisingly, we show in Section 2 that there is a pairwise independent family $\mathcal{H}$ such that

$$\mathbb{E}_{h\in\mathcal{H}}\left[\sup_{1\leq t\leq n}|h_1+\cdots+h_t|\right] = \Omega\left(\sqrt{n}\lg n\right), \tag{1}$$

meaning there is a uniform pairwise independent ±1-valued random walk which is not continuously bounded in expectation by $O(\sqrt{n})$. Furthermore, this bound of $\sqrt{n}\lg n$ is tight up to the first and second moments, because in Section 3 we prove that for any pairwise independent family $\mathcal{H}$ from $[n]$ to $\{-1, 1\}$ with $\mathbb{E}[h_i] = 0$ for all i,

$$\mathbb{E}_{h\in\mathcal{H}}\left[\sup_{1\leq t\leq n}(h_1+\cdots+h_t)^2\right] = O\left(n\lg^2 n\right). \tag{2}$$

In Section 4, we uniformly bound random walks corresponding to insertion-only streams and random walks with step sizes not necessarily uniform ±1 variables. We first generalize Kolmogorov's inequality [16] by proving that for any 4-wise independent random variables $X_1, \ldots, X_n$ with mean 0 and finite variance,

$$\mathbb{P}\left(\sup_{1\leq i\leq n}|X_1+\cdots+X_i|\geq\lambda\right) \leq \frac{\sum\mathbb{E}[X_i^2]}{\lambda^2} \tag{3}$$

for all $\lambda > 0$. In Appendix A, we generalize Equation (2) by proving for any family $X_1, \ldots, X_n$ of pairwise independent variables such that $\mathbb{E}[X_i] = 0, \mathbb{E}[X_i^2] \leq 1$, and for any insertion stream $p_1, \ldots, p_m \in [n]$,

$$\mathbb{E}\left[\sup_{1\leq t\leq m}\left|\langle \vec{X}, z^{(t)}\rangle\right|^2\right] = O\left(||z||_2^2 \lg^2 m\right) \tag{4}$$

where $z = z^{(m)}$ is the final position of the vector. Finally, we show that for any even $k \geq 4$, any k-wise independent family $X_1, \ldots, X_n$ such that $\mathbb{E}[X_i] = 0, \mathbb{E}[X_i^k] \leq 1$, and any insertion stream $p_1, \ldots, p_m \in [n]$,

$$\mathbb{E}\left[\sup_{1\leq t\leq m}\left|\langle \vec{X}, z^{(t)}\rangle\right|^k\right] = O\left(||z||_2^k\right). \tag{5}$$

Equations (3), (4), and (5) are interesting together as they provide various bounds on the supremum of generalized random walks under differing moment bounds and degrees of independence.

Finally, we note that to prove Equation (1), we create a complicated pairwise independent hash function, which suggests that standard pairwise independent hash functions do not have this property. Indeed, for many such families, such as some types of codes constructed from Hadamard matrices or random linear functions from $\mathbb{Z}/p\mathbb{Z} \to \mathbb{Z}/p\mathbb{Z}$, we have $\mathbb{E}\sup_i |h_1 + \cdots + h_i| = o(n \lg n)$. (See the appendices in the arXiv version of this paper.) However, we note that for some standard pairwise independent hash functions, it is difficult to provide either an upper or lower bound for $\mathbb{E}\sup_i |h_1 + \cdots + h_i|$. Therefore, even if some simpler pairwise independent hash function satisfies Equation 1, our hash family has the advantage that the analysis is simpler, even if the construction of the family is not.

1.2 Motivation and Relation to Previous Work

The primary motivation of this paper comes from certain theorems that provide strong bounds for certain variants of 4-wise independent random walks, which raised the question of whether any of these bounds can be extended to 2-wise independence. For example, Theorem 1 in [8] proves for any family $\mathcal{H}$ of $h \in \{-1, 1\}^n$ with 4-wise independent coordinates, $\mathbb{E}_{h\in\mathcal{H}}\left(\sup_t \langle h, z^{(t)}\rangle\right) = O(||z||_2)$. This result generalizes a result from [9] which proves the same but only if h is uniformly chosen from $\{-1, 1\}^n$. [8] provides an algorithm that successfully finds all ℓ_2 ε-heavy hitters in an insertion-only stream in $O(\varepsilon^{-2} \log \varepsilon^{-1})$ space, in which the above result was crucial for analysis of a subroutine which attempts to find bit-by-bit the index of a single "super-heavy" heavy hitter if one exists. Theorem 1 in [8] also proved valuable for an algorithm for continuous monitoring of ℓ_p norms in insertion-only data streams [6]. Lemma 18 in [5] shows that even without bounded fourth moments, given 4-wise independent random variables $X_1, \ldots, X_n$, each with mean 0 and finite variance,

$$\mathbb{P}\left(\max_{1\leq i\leq n} |X_1 + \cdots + X_i| \geq \lambda\right) = O\left(\frac{n \cdot \max_i \mathbb{E}[X_i^2]}{\lambda^2}\right).$$

This theorem was crucial in analyzing an algorithm tracking distinct elements that provides a $(1+\varepsilon)$-approximation with failure probability δ in $O(\varepsilon^{-2} \lg \delta^{-1} + \lg n)$ bits of space. Notice that our Equation (3) is stronger than the above equation and is asymptotically equivalent to Kolmogorov's inequality, though under much weaker assumptions.

A natural follow-up question to the above theorems is whether 4-wise independence is necessary, or whether lesser levels of independence such as 2-wise or 3-wise are required. Equation (1) shows that 2-wise independence does not suffice for any of the above results,

because the random walk on a line case is strictly weaker than all of the above results, though the case of 3-wise independence is still unknown. As a result, we know that the tracking sketches in [8, 6, 5] cannot be extended to 2-wise independent sketches.

However, the results given still have interesting extensions, such as to higher moments. Equation (5) shows a stronger result than the one established in [8], since it not only bounds the first moment of $\sup_t \langle h, z^{(t)} \rangle$ for a 4-wise independent family of uniform ± 1 variables but also bounds the 4th moment equally (as they have mean 0 and kth moment 1). The main methods used for proving most of our upper bounds are based on chaining methods, specifically Dudley chaining, with slight modifications, although the bounds in Section 3 are proved differently from standard chaining methods but are still motivated by similar ideas. Dudley chaining was introduced in [11], and Dudley chaining and other chaining techniques, along with applications, are summarized in [18].

k-wise independence for hash functions was first introduced in [10]. Bounding the amount of independence required for analysis of algorithms has been studied in various contexts, often since k-wise independent hash families can be stored in low space but may provide equally adequate bounds as totally independent families. As further examples, the well-known AMS sketch [1] is a streaming algorithm to estimate the ℓ_2 norm of a vector z to a factor of $1 \pm \varepsilon$ with high probability by multiplying the vector by a sketch matrix $\Pi \in \mathbb{R}^{n \times (1/\varepsilon^2)}$ of 4-wise independent random signs and using $||\Pi z||_2$ as an estimate for $||z||_2$. It is known from [20, 22] that the accuracy of the AMS sketch can be much worse if 3-wise independent random signs are used instead of 4-wise independent random signs. If z is given as an insertion stream, it is known that the AMS sketch with 8-wise independent random signs can provide weak tracking [8], meaning that $\mathbb{E} \sup_t \left| ||\Pi z^{(t)}||_2^2 - ||z^{(t)}||_2^2 \right| \leq \varepsilon ||z||_2^2$. This implies that the approximation of the ℓ_2 norm with the 8-wise independent AMS sketch is quite accurate at all times t. While one cannot perform weak tracking with 3-wise independence of the AMS sketch, it is unknown for 4-wise independence through 7-wise independence whether the AMS sketch provides weak tracking. Finally, linear probing, a well-known implementation of hash tables, was shown to take $O(1)$ expected update time with any 5-wise independent hash function [19] but was shown to take $\Theta(\lg n)$ expected update time for certain 4-wise independent hash functions and $\Theta(\sqrt{n})$ expected update time for certain 2-wise independent hash functions [20].

Bounding the maximum distance traveled of a random walk has also been studied in probability theory independent of computer science applications, both when the steps are totally independent or k-wise independent. For example, Kolmogorov's inequality [16] provides bounds for $\sup_t(X_1 + \cdots + X_t)$ for independent random variables $X_1, \ldots, X_t$ even if only the second moments of $X_1, \ldots, X_t$ are finite. [3] constructed an infinite sequence $\{X_1, X_2, \ldots\}$ of pairwise independent random variables taking on the values ± 1 such that $\sup_t(X_1 + \cdots + X_t)$ is bounded almost surely, though the paper also proved that this phenomenon can never occur for 4-wise independent variables taking on the values ± 1. Finally, the supremum of a random walk with i.i.d. bounded random variable steps was studied in [12], which provided comparisons with the supremum of a Brownian motion random walk regardless of the random variable chosen for step size.

1.3 Notation

We define $[n] := \{1, \ldots, n\}$, and treat $p_1, \ldots, p_m \in [n]$ as an insertion-only stream that keeps track of a vector z that starts at the origin and increments its p_jth component by 1 after we see p_j.

A *k-wise independent family* from $[n]$ to $\{-1,1\}$ is a family $\mathcal{H}$ of functions $h:[n]\to\{-1,1\}$ such that for any k distinct indices, their values are independent *Rademachers*, where Rademachers are random variables uniformly selected from $\{-1,1\}$. A *k-wise independent random walk* is a random walk where one's position after t steps is $h(1)+\cdots+h(t)$, with h chosen from $\mathcal{H}$. We may also denote a k-wise independent random walk as a random walk where the ith step is a random variable X_i, assuming $X_1,\ldots,X_n$ are random variables such that any k distinct X_i's are totally independent.

In this paper, we think of a hash function $h:[n]\to\{-1,1\}$ as a vector in $\mathbb{R}^n$, where $h_i=h(i)$, for the purpose of denoting inner products. Similarly, treat $\vec{X}$ as the vector $(X_1,\ldots,X_n)$.

Finally, in Section 2, we assume that n is a power of 4, in Section 3, we assume n is a power of 2 and is at least 4, and in Section 4, we assume m is a sufficiently large power of 2. We note that these assumptions can be removed by replacing n with the largest power of 4 less than n or the smallest power of 2 greater than n or m, respectively.

1.4 Overview of Proof Ideas

Here, we briefly outline some of the main ideas behind the proofs of Equations (1) through (5).

The main goal in Section 2 is to establish Equation (1), i.e. construct a pairwise independent $\mathcal{H}$ such that $\mathbb{E}[h_ih_j]=0$ for all $i\neq j$,. In other words, we wish for the covariance matrix $\mathcal{M}=\mathbb{E}[h^Th]$ to be the identity matrix I_n. We also want $\sup_{1\le i\le n}|h_1+\cdots+h_i|$ to be $\Omega(\sqrt{n}\lg n)$ in expectation. The construction has two major steps.

1. Create a hash function such that $\mathbb{E}\sup_{1\le i\le n}|h_1+\cdots+h_i|=\Omega(\sqrt{n}\lg n)$ but rather than have $\mathbb{E}[h_ih_j]=0$ for all $i\neq j$, have $\sum_{i\neq j}|\mathbb{E}[h_ih_j]|=O(n)$, i.e. the cross terms in total aren't very large in absolute value (this hash function will be $\mathcal{H}_2$ in our proof). To do this, we first created $\mathcal{H}_1$, which certain properties, most notably that $\mathbb{E}[h_1+\cdots+h_n]=0$ but $\mathbb{E}[h_1+\cdots+h_{n/2}]=\Theta(\sqrt{n}\lg n)$, and rotated the hash family by a uniform index. The rotation allows many of the cross terms to average out, reducing the sum of their absolute values.
2. Remove the cross terms. To do this, we make $\mathcal{H}$ a hash family where with some constant probability, we choose from $\mathcal{H}_2$ and with some probability, we choose some set of indices and pick a hash function such that $\mathbb{E}[h_ih_j]$ will be the opposite sign of $\mathbb{E}_{h\in\mathcal{H}_2}[h_ih_j]$ for certain indices i,j, so that overall, $\mathbb{E}[h_ih_j]$ will be 0. Certain symmetry properties and most importantly the fact that $\sum_{i\neq j}|\mathbb{E}_{h\in\mathcal{H}_2}[h_ih_j]|=O(n)$ will allow for us to choose from $\mathcal{H}_2$ with constant probability, which means even for our final hash function, $\mathbb{E}\sup_{1\le i\le n}|h_1+\cdots+h_i|=\Omega(\sqrt{n}\lg n)$.

The goal of Section 3 is to establish Equation (2), i.e. to show that if $\mathcal{M}=\mathbb{E}[h^Th]=I_n$, which is true for any pairwise independent hash function, then $\sup_{1\le i\le n}|h_1+\cdots+h_i|^2=O(n\lg^2 n)$. To do this, we apply probabilistic method ideas. We notice that for any matrix A, $\mathbb{E}[h^TAh]=Tr(A)$, and thus, if we can find a matrix such that the trace of the matrix is small, but h^TAh is reasonably large in comparison to $\sup_{1\le i\le n}|h_1+\cdots+h_i|^2$, then $\mathbb{E}[h^TAh]$ is small but is large in comparison to $\mathbb{E}\left[\sup_{1\le i\le n}|h_1+\cdots+h_i|^2\right]$. If we assume that n is a power of 2, then the matrix that corresponds to the quadratic form

$$h^TAh=\sum_{r=0}^{\lg n}\sum_{i=0}^{(n/2^r)-1}(h_{i\cdot 2^r+1}+\cdots+h_{(i+1)\cdot 2^r})^2,$$

i.e. $h^TAh=h_1^2+\cdots+h_n^2+(h_1+h_2)^2+\cdots+(h_{n-1}+h_n)^2+\cdots+(h_1+\cdots+h_n)^2$ can be shown to satisfy $Tr(A)=n\lg n$ and for any vector x, $x^TAx\ge\frac{1}{\lg n}\cdot(x_1+\cdots+x_i)^2$ for all $1\le i\le n$, not just in expectation. These conditions will happen to be sufficient for our goals.

This method, in combination with Equation (1), will also allow us to prove an interesting matrix inequality, proven at the end of Section 3. The method above actually generalizes to looking at kth moments of k-wise independent hash functions, as well as random walks corresponding to tracking insertion-only streams, and will allow us to prove Equations (4) and (5). However, these generalizations will also need the construction of ε-nets, which are explained in Appendix A, or in [18].

We finally explain the ideas behind Equation (3), the generalization of Kolmogorov's inequality and Lemma 18 of [5]. We use ideas of chaining, such as in [18], and an idea of [5] that allows us to bound the minimum of $X_{i+1} + \cdots + X_j$ and $X_{j+1} + \cdots + X_k$ where $i < j < k$, given 4-wise independent functions $X_1, \ldots, X_n$ with only bounded second moments. We combine these with another idea, that we can consider distances between i and j for $1 \le i < j \le n$ as $\mathbb{E}[X_{i+1}^2 + \cdots + X_j^2]$ and that for any $i < j < k$, either $\mathbb{E}[X_{i+1}^2 + \cdots + X_j^2]$ is very small and we can bound $X_{i+1} + \cdots + X_j$, $\mathbb{E}[X_{j+1}^2 + \cdots + X_k^2]$ is very small and we can bound $X_{j+1} + \cdots + X_k$, or we can bound $\min(|X_{i+1} + \cdots + X_j|, |X_{j+1} + \cdots + X_k|)$ with the idea of [5]. These ideas allow for our chaining method to be quite effective, even if the X_i's do not have bounded 4th moments or if the X_i's wildly differ in variance.

2 Lower Bounds for Pairwise Independence

In this section, we construct a 2-wise independent family $\mathcal{H}$ such that the furthest distance traveled by the random walk is $\Omega(\sqrt{n} \lg n)$ in expected value. In other words, we prove the following:

▶ **Theorem 1.** *There exists a 2-wise independent hash family $\mathcal{H}$ from $[n] \to \{-1, 1\}$ such that*

$$\mathbb{E}_{h \in \mathcal{H}} \left[\sup_{1 \le t \le n} \left| \sum_{1 \le j \le t} h_j \right| \right] = \Omega(\sqrt{n} \lg n).$$

To actually construct this counterexample, we proceed by a series of families and tweak each family accordingly to get to the next one, until we get the desired $\mathcal{H}$.

We start by creating $\mathcal{H}_1$. First, split $[n]$ into blocks of size $\sqrt{n}$ so that $\{(c-1)\sqrt{n} + 1, \ldots, c\sqrt{n}\}$ form the cth block for each $1 \le c \le \sqrt{n}$. Also, define $\ell = \frac{\sqrt{n}}{2}$. Now, to pick a function h from $\mathcal{H}_1$, choose the value of h_i for each $1 \le i \le n$ independently, but if i is in the cth block for some $1 \le c \le \ell$, make $\mathbb{P}[h_i = 1] = \frac{1}{2} + \frac{1}{2(\ell+1-c)}$ and if i is in the cth block for some $\ell + 1 \le c \le \sqrt{n}$, make $\mathbb{P}[h_i = 1] = \frac{1}{2} - \frac{1}{2(c-\ell)}$. This way, $\mathbb{E}[h_i] = \frac{1}{\ell+1-c}$ if i is in the cth block for $c \le \ell$ and $\mathbb{E}[h_i] = -\frac{1}{c-\ell}$ if i is in the cth block for $c > \ell$.

From now on, assume that h_i is periodic modulo n, i.e. $h_{i+n} = h_i$ for all integers i. We first prove the following about $\mathcal{H}_1$:

▶ **Lemma 2.** *Suppose that $1 \le i < j \le n$. Suppose that i is in block c_1 and j is in block c_2, where c_1 and c_2 are not necessarily distinct. Define $r = \min(c_2 - c_1, \sqrt{n} - (c_2 - c_1))$. Then,*

$$\sum_{d=0}^{\sqrt{n}-1} \mathbb{E}_{h \in \mathcal{H}_1}(h_{i+d\sqrt{n}} h_{j+d\sqrt{n}}) = O\left(\frac{\lg(r+2)}{(r+1)^2}\right).$$

Proof. For $1 \leq c \leq \sqrt{n}$, define f_c to equal $\frac{1}{\ell+1-c}$ if $1 \leq c \leq \ell$ and to equal $-\frac{1}{c-\ell}$ if $\ell+1 \leq c \leq \sqrt{n}$. In other words, $f_c = \mathbb{E}[h_i]$ if i is in the cth block. Furthermore, assume that f is periodic modulo $\sqrt{n}$, i.e. $f_c = f_{c+\sqrt{n}}$ for all integers c. Then,

$$\sum_{d=0}^{\sqrt{n}-1} \mathbb{E}_{h\in\mathcal{H}_1}(h_{i+d\sqrt{n}}h_{j+d\sqrt{n}}) = \sum_{d=0}^{\sqrt{n}-1} \mathbb{E}_{h\in\mathcal{H}_1}(h_{i+d\sqrt{n}})\mathbb{E}_{h\in\mathcal{H}_1}(h_{j+d\sqrt{n}})$$
$$= \sum_{d=0}^{\sqrt{n}-1} f_{c_1+d}f_{c_2+d} = \sum_{d=1}^{\sqrt{n}} f_d f_{r+d}.$$

Now, since $r \leq \ell$, if we assume $r \geq 1$, this sum can be explicitly written as

$$2\cdot\sum_{d=1}^{\ell-r}\frac{1}{d(d+r)} - \sum_{d=1}^{r}\frac{1}{d(r+1-d)} - \sum_{d=1}^{r}\frac{1}{(n+1-d)(n+1-(r+1-d))}$$
$$\leq 2\sum_{d=1}^{\infty}\frac{1}{d(d+r)} - \sum_{d=1}^{r}\frac{1}{d(r+1-d)}$$
$$= \frac{2}{r}\sum_{d=1}^{\infty}\left(\frac{1}{d}-\frac{1}{d+r}\right) - \frac{1}{r+1}\sum_{d=1}^{r}\left(\frac{1}{d}+\frac{1}{r+1-d}\right)$$
$$= \frac{2}{r}\left(\sum_{d=1}^{r}\frac{1}{d}\right) - \frac{2}{r+1}\left(\sum_{d=1}^{r}\frac{1}{d}\right)$$
$$= \frac{2}{r(r+1)}\left(\sum_{d=1}^{r}\frac{1}{d}\right) \leq \frac{C_1\lg(r+2)}{(r+1)^2}$$

for some constant C_1. If we assume $r = 0$, then this sum can be explicitly written as

$$2\cdot\sum_{d=1}^{\ell}\frac{1}{d^2} \leq C_2 = \frac{(C_2)\cdot\lg(0+2)}{(0+1)^2}$$

for some constant C_2. Therefore, setting $C_3 = \max(C_1, C_2)$ as our constant, we are done. ◀

To construct $\mathcal{H}_2$, first choose $h \in \mathcal{H}_1$ at random, and then choose an index d between 0 and $\sqrt{n}-1$ uniformly at random. Our chosen function h' will then be the function that satisfies $h'_i = h_{i+d\cdot\sqrt{n}}$ for all i. We show the following about $\mathcal{H}_2$:

▶ **Lemma 3.** *The following three statements are true:*

a) *For all $i, j \in \mathbb{Z}$, $\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = \mathbb{E}_{h\in\mathcal{H}_2}(h_{i+\sqrt{n}}h_{j+\sqrt{n}})$.*

b) *Suppose that $1 \leq i, i', j, j' \leq n$, where i, i' are in blocks c_1, j, j' are in blocks c_2, and $i \neq j, i' \neq j'$. Then, $\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = \mathbb{E}_{h\in\mathcal{H}_2}(h_{i'}h_{j'})$.*

c) *$\sum_{i\neq j}|\mathbb{E}_{h\in\mathcal{H}_2}h_ih_j| = O(n)$.*

Proof. Part a) is quite straightforward, since

$$\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = \frac{1}{\sqrt{n}}\sum_{d=0}^{\sqrt{n}-1}\mathbb{E}_{h\in\mathcal{H}_1}(h_{i+d\sqrt{n}}h_{j+d\sqrt{n}})$$

$$= \frac{1}{\sqrt{n}}\sum_{d=0}^{\sqrt{n}-1}\mathbb{E}_{h\in\mathcal{H}_1}(h_{i+(d+1)\sqrt{n}}h_{j+(d+1)\sqrt{n}}) = \mathbb{E}_{h\in\mathcal{H}_2}(h_{i+\sqrt{n}}h_{j+\sqrt{n}})$$

by periodicity of h modulo n.

For part b), for all $d \in \mathbb{Z}$, note that $i + d\sqrt{n}$ and $i' + d\sqrt{n}$ are in the same blocks, $j + d\sqrt{n}$ and $j' + d\sqrt{n}$ are in the same blocks, $i + d\sqrt{n} \neq j + d\sqrt{n}$ and thus $h_{i+d\sqrt{n}}, h_{j+d\sqrt{n}}$ are independent, and $i' + d\sqrt{n} \neq j' + d\sqrt{n}$ and thus $h_{i'+d\sqrt{n}}, h_{j'+d\sqrt{n}}$ are independent. Therefore, $\mathbb{E}_{h\in\mathcal{H}_1}(h_{i+d\sqrt{n}}h_{j+d\sqrt{n}}) = \mathbb{E}_{h\in\mathcal{H}_1}(h_{i'+d\sqrt{n}}h_{j'+d\sqrt{n}})$ for all d. Because of the way we constructed $\mathcal{H}_2$, part b) is immediate from these observations.

We use Lemma 2 to prove part c). First note that for all $i \neq j$,

$$\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = \frac{1}{\sqrt{n}}\sum_{d=0}^{\sqrt{n}-1}\mathbb{E}_{h\in\mathcal{H}_1}(h_{i+d\sqrt{n}}h_{j+d\sqrt{n}}) \leq \frac{C_3\lg(r+2)}{\sqrt{n}\cdot(r+1)^2},$$

where i is in block c_1, j is in block c_2, and $r = \min(|c_1 - c_2|, \sqrt{n} - |c_1 - c_2|)$. Now, there are exactly $n(\sqrt{n}-1)$ pairs (i,j) where $1 \leq i,j \leq n$, $i \neq j$, and $r = 0$. This is because we can choose from $\sqrt{n}$ blocks for the value of $c_1 = c_2$, and then choose from $\sqrt{n}(\sqrt{n}-1)$ possible pairs (i,j) in each block. For a fixed $0 < r < \ell$, there are exactly $2n^{3/2}$ pairs (i,j), since there are $2\sqrt{n}$ choices for blocks c_1 and c_2 and $\sqrt{n}$ choices for each of i and j after that, for $r = \ell$, there are exactly $n^{3/2}$ such pairs, since there are $2\sqrt{n}$ choices for blocks c_1 and c_2 and $\sqrt{n}$ choices for each of i and j after that, and finally we cannot have $r > \ell$. Therefore,

$$\sum_{i\neq j}\max\left(0, \mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j)\right) \leq 2n^{3/2}\cdot\sum_{r=0}^{\ell}\frac{C_3\lg(r+2)}{\sqrt{n}(r+1)^2} \leq C_4 n$$

for some constant C_4, since $\sum \frac{\lg(r+2)}{(r+1)^2}$ is a convergent series.

To finish, note that $|x| = 2\cdot\max(0,x) - x$, so

$$\sum_{i\neq j}|\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j)| \leq 2\cdot C_4 n - \sum_{i\neq j}\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) \leq (2C_4+1)n,$$

since

$$\sum_{i\neq j}\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = \sum_{i,j}\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) - \sum_i\mathbb{E}_{h\in\mathcal{H}_2}h_i^2 = \mathbb{E}_{h\in\mathcal{H}_2}(h_1+\cdots+h_n)^2 - n \geq -n.$$

Thus, setting $C_5 = 2C_4 + 1$ gets us our desired result. ◀

Next, we tweak $\mathcal{H}_2$ to create a new family $\mathcal{H}_3$. First, notice that we can define $g_{c_1c_2}$ for $1 \leq c_1, c_2 \leq \sqrt{n}$ to equal $\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j)$ for some i in the c_1th block and j in the c_2th block such that $i \neq j$. This is well defined by Lemma 3 b), and as $1 \leq c_1, c_2 \leq \sqrt{n}$, there always exist $i \neq j$ with i in the c_1th block and j in the c_2th block, as long as $n \geq 4$. Now, to create $\mathcal{H}_3$, define $g = 1 + \sum_{c_1<c_2}|g_{c_1c_2}|$. Then, with probability $\frac{1}{g}$, we choose a hash function from $\mathcal{H}_2$. With probability $\frac{|g_{c_1c_2}|}{g}$ for each $1 \leq c_1 < c_2 \leq \sqrt{n}$, we choose $h_i = 1$ for all i in the c_1th bucket, if $g_{c_1c_2} \geq 0$, we make $h_i = -1$ for all i in the c_2th bucket and if $g_{c_1c_2} < 0$, we make $h_i = 1$ for all i in the c_2th bucket, and if i is not in either the c_1th or the c_2th bucket, we let h_i be an independent Rademacher. We prove the following about $\mathcal{H}_3$:

▶ **Lemma 4.** *If i and j are in different buckets, then $\mathbb{E}_{h\in\mathcal{H}_3}(h_ih_j) = 0$. If i,j are in the same bucket but $i \neq j$, then there is some constant $0 \leq C_6 \leq C_5$ such that $\mathbb{E}_{h\in\mathcal{H}_3}(h_ih_j) = \frac{C_6}{\sqrt{n}}$.*

Proof. Assume WLOG that $i < j$. If i,j are in different buckets, then we compute $\mathbb{E}_{h\in\mathcal{H}_3}(h_ih_j)$ as follows. With probability $\frac{1}{g}$, we are choosing h from $\mathcal{H}_2$, and if i is in the c_1th bucket and j is in the c_2th bucket, then $\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = g_{c_1c_2}$. With probability $\frac{|g_{c_1c_2}|}{g}$ we have $h_ih_j = 1$ with probability 1 if $g_{c_1c_2} < 0$ and $h_ih_j = -1$ with probability 1 if

$g_{c_1c_2} \geq 0$. In all other scenarios, either h_i or h_j is a Rademacher completely independent of all other elements, which means that $\mathbb{E}[h_ih_j] = 0$. Therefore, the overall expected value of h_ih_j equals $g_{c_1c_2} \cdot \frac{1}{g} + \frac{|g_{c_1c_2}|}{g} \cdot \pm 1$ where the ± 1 is positive if and only if $g_{c_1c_2} \leq 0$, so the expected value is 0.

If i, j are in the same bucket, then we can compute $\mathbb{E}_{h\in\mathcal{H}_3}(h_ih_j)$ as follows. With probability $\frac{1}{g}$, we are choosing h from $\mathcal{H}_2$, and if i, j are in the cth bucket, then $\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = g_{cc}$. For all $c' \neq c$, there is a $\frac{|g_{cc'}|}{g}$ probability of everything in the cth block having the same sign and everything in the c'th block having the same sign. For the other cases, i, j are independent Rademachers. Therefore,

$$\mathbb{E}_{h\in\mathcal{H}_3}(h_ih_j) = \frac{g_{cc}}{g} + \sum_{c'\neq c} \frac{|g_{cc'}|}{g} = \frac{1}{g}\left(g_{cc} + \sum_{c'\neq c} |g_{cc'}|\right).$$

However, note that $g_{cc} \geq 0$ since $\mathbb{E}_{h\in\mathcal{H}_2}(h_ih_j) = \frac{1}{\sqrt{n}}\sum_d \mathbb{E}_{h\in\mathcal{H}_1}(h_{i+d\sqrt{n}}h_{j+d\sqrt{n}})$ and for all d, we have $\mathbb{E}_{h\in\mathcal{H}_1}(h_{i+d\sqrt{n}}h_{j+d\sqrt{n}}) \geq 0$ since $i + d\sqrt{n}, j + d\sqrt{n}$ are in the same block for all d. Furthermore, for all indices c_1, c_2, $g_{c_1c_2} = g_{(c_1+1)(c_2+1)}$, where indices are taken modulo $\sqrt{n}$, by Lemma 3 a). Combining these gives

$$\mathbb{E}_{h\in\mathcal{H}_3}(h_ih_j) = \frac{1}{\sqrt{n}} \cdot \left(\frac{1}{g} \cdot \left(\sum_{c_1,c_2} |g_{c_1c_2}|\right)\right).$$

However, we know that $g \geq 1$ and $\sum_{c_1,c_2} |g_{c_1c_2}| \leq C_5$ by the arguments of Lemma 3 c), so the lemma follows. ◀

Now, we are almost done. To create $\mathcal{H}$, with probability $p = \frac{1}{1+C_6(\sqrt{n}-1)/\sqrt{n}} \geq \frac{1}{1+C_6}$, choose h from $\mathcal{H}_3$, and assuming we chose from $\mathcal{H}_3$, with probability $\frac{1}{2}$ negate $h_1, \ldots, h_n$. With probability $1 - p$, for each block of $\sqrt{n}$ elements, choose uniformly at random a subset of size ℓ from the block, and make the corresponding elements 1 and the remaining elements -1. It is easy to see that now, $\mathbb{E}_{h\in\mathcal{H}}(h_i) = 0$ because of the possibility of negating. Moreover, $\mathbb{E}_{h\in\mathcal{H}}(h_ih_j) = 0$ for all $i \neq j$. To see why, if i and j are in different blocks then $\mathbb{E}_{h\in\mathcal{H}_3}(h_ih_j) = 0$ and if we do not choose h from $\mathcal{H}_3$, then h_i and h_j are independent. If i, j are in the same block, then if we condition on choosing from $\mathcal{H}_3$, $\mathbb{E}(h_ih_j) = \frac{C_6}{\sqrt{n}}$. If we condition on not choosing from $\mathcal{H}_3$, the probability of i, j being the same sign is $\frac{(\sqrt{n}/2)-1}{\sqrt{n}-1} = \frac{\ell-1}{2\ell-1}$, meaning $\mathbb{E}(h_ih_j) = -\frac{1}{\sqrt{n}-1}$. Therefore, $\mathbb{E}_{h\in\mathcal{H}}(h_ih_j) = p \cdot \frac{C_6}{\sqrt{n}} - (1-p) \cdot \frac{1}{\sqrt{n}-1} = 0$.

To finish, it suffices to show that

$$\mathbb{E}_{h\in\mathcal{H}}\left[\sup_{1\leq t\leq n} |h_1 + \cdots + h_t|\right] = \Omega(\sqrt{n}\lg n).$$

To check this, note that with probability at least $\frac{1}{1+C_6}$ we are picking something from $\mathcal{H}_3$, so we need to just verify that

$$\mathbb{E}_{h\in\mathcal{H}_3}\left[\sup_{1\leq t\leq n} |h_1 + \cdots + h_t|\right] = \Omega(\sqrt{n}\lg n).$$

But for $\mathcal{H}_3$, we are choosing something from $\mathcal{H}_2$ with probability $\frac{1}{g}$ but $g \leq 1 + C_5$ by the arguments of Lemma 3 c), so it suffices to verify that

$$\mathbb{E}_{h\in\mathcal{H}_2}\left[\sup_{1\leq t\leq n} |h_1 + \cdots + h_t|\right] = \Omega(\sqrt{n}\lg n).$$

But for $\mathcal{H}_2$, if we condition on the shifting index d, we know that

$$\mathbb{E}[h_{1+d\sqrt{n}} + h_{2+d\sqrt{n}} + \cdots + h_{(d+\ell)\sqrt{n}}] \geq \sqrt{n}\left(1 + \cdots + \frac{1}{\ell}\right) \geq C_7\sqrt{n}\lg n$$

for some C_7, and likewise

$$\mathbb{E}[h_{1+(d+\ell)\sqrt{n}} + h_{2+(d+\ell)\sqrt{n}} + \cdots + h_{(d+2\ell)\sqrt{n}}] \leq \sqrt{n}\left(-1 - \cdots - \frac{1}{\ell}\right) \leq -C_7\sqrt{n}\lg n,$$

which means that regardless of whether $d \leq \ell$ or $d > \ell$,

$$\mathbb{E}_{h\in\mathcal{H}_2}\left[\max\left(|h_1 + \cdots + h_{d\sqrt{n}}|, |h_1 + \cdots + h_{(d+\ell)\sqrt{n}}|\right)\right] \geq \frac{C_7}{2}\sqrt{n}\lg n$$

by the triangle inequality. But for any $h \in \mathcal{H}_2$,

$$\max\left(|h_1 + \cdots + h_{d\sqrt{n}}|, |h_1 + \cdots + h_{(d+\ell)\sqrt{n}}|\right) \leq \sup_{1\leq t\leq n}(h_1 + \cdots + h_t),$$

so the result follows by taking the expected value of both sides, which proves our upper bound is tight in the case of a random walk. Thus, we have proven Theorem 1.

3 Moment Bounds for Pairwise Independence

We show that the bound established in Section 2 and the induced bound on the second moment are tight for the 2-wise independent random walk case by proving Equation (2) in Section 1.1:

▶ **Theorem 5.** *For all 2-wise families $\mathcal{H}$ from $[n]$ to $\{-1, 1\}$,*

$$\mathbb{E}_{h\in\mathcal{H}}\left(\sup_{1\leq i\leq n}(h_1 + \cdots + h_i)^2\right) = O(n\lg^2 n).$$

We provide a generalization of this theorem in Section 4, with a slightly different method. To prove this, we first establish the following lemma:

▶ **Lemma 6.** *Suppose that there exists a positive definite matrix $A \in \mathbb{R}^{n\times n}$ such that $Tr(A) = d_1$ for some $d_1 > 0$ and there exists some function f such that for all vectors $x \in \mathbb{R}^n$ and integers $1 \leq i \leq n$, if $x_1 + \cdots + x_i = 1$, then $x^T Ax \geq \frac{1}{d_2}$ for some $d_2 > 0$. Then, for all 2-wise families $\mathcal{H}$,*

$$\mathbb{E}_{h\in\mathcal{H}}\left(\sup_{1\leq i\leq n}(h_1 + \cdots + h_i)^2\right) \leq d_1 d_2.$$

Proof. Note that $\mathbb{E}_{h\in\mathcal{H}}h_i^2 = 1$ for all i and $\mathbb{E}_{h\in\mathcal{H}}(h_i h_j) = 0$ for all $i \neq j$. Therefore,

$$\begin{aligned}\mathbb{E}_{h\in\mathcal{H}}(h^T Ah) &= \sum_{1\leq i,j\leq n}\mathbb{E}_{h\in\mathcal{H}}(h_i h_j A_{ij}) = \sum_{1\leq i,j\leq n}A_{ij}\left(\mathbb{E}_{h\in\mathcal{H}}(h_i h_j)\right)\\ &= \sum_{1\leq i\leq n}A_{ii} = Tr(A) = d_1.\end{aligned}$$

However, for any $1 \leq i \leq n$, for any $h \in \mathcal{H}$, if $h_1 + \cdots + h_i \neq 0$, then

$$h^T Ah \geq (h_1 + \cdots + h_i)^2 \cdot \frac{1}{d_2},$$

since the vector $\frac{1}{h_1+\cdots+h_i} \cdot h$ has its first i components sum to 1, so we can let this vector equal x to get $x^T Ax \geq \frac{1}{f(n)}$. If $h_1 + \cdots + h_i = 0$, then the above inequality is still true as A is positive definite.

Therefore,

$$h^T Ah \geq \frac{1}{d_2} \cdot \sup_{1 \leq i \leq n} (h_1 + \cdots + h_i)^2,$$

which means that

$$d_1 = \mathbb{E}_{h \in \mathcal{H}}(h^T Ah) \geq \frac{1}{d_2} \cdot \mathbb{E}_{h \in H} \left(\sup_{1 \leq i \leq n} (h_1 + \cdots + h_i)^2 \right),$$

so we are done. ◀

▶ **Lemma 7.** *There exists a positive definite matrix $A \in \mathbb{R}^{n \times n}$ such that $Tr(A) = n \lg n$ and for all $x \in \mathbb{R}^n$ and $1 \leq i \leq n$, if $x_1 + \cdots + x_i = 1$, then $x^T Ax \geq \frac{1}{\lg n}$. This clearly implies Theorem 5.*

Proof. Consider the matrix A such that for all $1 \leq i, j \leq n$, $A_{ij} = \lg n - k$ if k is the smallest nonnegative integer such that $\lfloor \frac{i-1}{2^k} \rfloor = \lfloor \frac{j-1}{2^k} \rfloor$. Alternatively, we can think of A as the sum of all matrices B^{ij}, where B^{ij} is a matrix such that $B^{ij}_{kl} = 1$ if $i \leq k, l \leq j$ and 0 otherwise. However, we sum this not over all $1 \leq i, j \leq n$ but for $i = 2^r \cdot (s-1) + 1, j = 2^r \cdot s$ for $0 \leq r \leq \lg n - 1$ and $1 \leq s \leq 2^{\lg n - r}$. As an illustrative example, for $n = 8$, A equals

$$\begin{pmatrix} 3 & 2 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 3 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 3 & 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 2 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 2 & 1 & 1 \\ 0 & 0 & 0 & 0 & 2 & 3 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 3 & 2 \\ 0 & 0 & 0 & 0 & 1 & 1 & 2 & 3 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

It is easy to see that $Tr(A) = n \lg n$, since $A_{ii} = \lg n$ for all i. For any $1 \leq i < n$, define $i_0 = 0$ and for any $1 \leq r \leq \lg n$, define $i_r = 2^{\lg n - r} \cdot \lfloor \frac{i}{2^{\lg n - r}} \rfloor$. Then, for any $1 \leq i < n$, one can see that $i_{\lg n} = i$ and for any $1 \leq i \leq n$, $A = B^{1 i_1} + B^{(i_1+1) i_2} + \cdots + B^{(i_{\lg n - 1}+1) i_{\lg n}} + C$, where C is some positive semidefinite matrix and we assume B^{ij} is the 0 matrix if $i = j + 1$, because $B^{1 i_1}$ and $B^{(i_{r-1}+1) i_r}$ for all $1 \leq r \leq \lg n$ are verifiable as matrices in the summation of A. Therefore, if $x_1 + \cdots + x_i = 1$,

$$\begin{aligned} x^T Ax &\geq \sum_{i=1}^{r} x^T B^{(i_{r-1}+1) i_r} x \\ &= (x_1 + \cdots + x_{i_1})^2 + (x_{i_1+1} + \cdots + x_{i_2})^2 + \cdots + (x_{i_{\lg n - 1}+1} + \cdots + x_{i_{\lg n}})^2 \\ &\geq \frac{1}{\lg n}, \end{aligned}$$

since $(x_1 + \cdots + x_{i_1}) + (x_{i_1+1} + \cdots + x_{i_2}) + \cdots + (x_{i_{\lg n - 1}+1} + \cdots + x_i) = 1$ and by Cauchy-Schwarz.

Finally, if $i = n$, then $A = B^{1(n/2)} + B^{(n/2+1)n} + C$, where C is some positive semidefinite matrix. Therefore, if $x_1 + \cdots + x_n = 1$,

$$x^T Ax \geq x^T B^{1(n/2)} x + x^T B^{(n/2+1)n} x$$
$$= (x_1 + \cdots + x_{n/2})^2 + (x_{n/2+1} + \cdots + x_n)^2 \geq \frac{1}{2} \geq \frac{1}{\lg n}. \quad \blacktriangleleft$$

As a final note, for any positive definite matrix A and vector v, the minimum value of $w^T Aw$ over all w such that $w^T v = 1$ is known to equal $(v^T A^{-1} v)^{-1}$. This can be checked with Lagrange Multipliers, since the Lagrangian $f(w, \lambda)$ of $f(w) = w^T Aw$ subject to $w^T v = 1$ equals $w^T Aw - \lambda(w^T v - 1)$, which is a convex function in w and has its derivatives vanish on the hyperplane $w^T v = 1$ when $\lambda = 2(v^T A^{-1} v)^{-1}, w = \frac{\lambda}{2}(A^{-1} v)$ (See for example [7], Chapter 5, for more details of Lagrange Multipliers). By Lemma 6 and Theorem 1, we have the following corollary:

▶ **Corollary 8.** *For all positive definite A, if we define v^i as the vector with first i components 1 and last $n - i$ components 0,*

$$Tr(A) \cdot \max_{1 \leq i \leq n} (v^i A^{-1} v^i) = \Omega(n \lg^2 n)$$

and this bound is tight for the matrix of Lemma 7.

Proof. If the first part were not true, then there would be matrices A_n such that $Tr(A) = d_1$, $w^T Aw = \frac{1}{d_2}$ where $w^T v^i = 0$ for some i, and $d_1 d_2 = o(n \lg^2 n)$. However, this would mean by Lemma 6 that for all pairwise independent $\mathcal{H}$,

$$\mathbb{E}_{h \in \mathcal{H}} \left(\sup_{1 \leq i \leq n} (h_1 + \cdots + h_i)^2 \right) \leq d_1 d_2 = o(n \lg^2 n),$$

contradicting Theorem 1. The second part is immediate by the analysis of Lemma 7. ◀

4 Generalized Upper Bounds

In this section, our goal is to prove Equation (3) of Section 1.1.

4.1 Proof of Equation 3

In this subsection, we prove a generalization of Kolmogorov's inequality [16] by proving an identical result even if we only know that our random variables $X_1, \ldots, X_n$ are 4-wise independent.

▶ **Theorem 9.** *Suppose that $X_1, \ldots, X_n$ are 4-wise independent random variables satisfying $\mathbb{E}[X_i] = 0$ and $Var(X_i) < \infty$ for all i. Then, for all $\lambda > 0$,*

$$\mathbb{P}\left(\sup_{1 \leq i \leq n} (X_1 + \cdots + X_i) \geq \lambda \right) \leq \frac{\sum \mathbb{E}[X_i^2]}{\lambda^2}.$$

Proof. Assume WLOG that $\lambda \geq 1$, $\sum \mathbb{E}[X_i^2] = 1$, and $\mathbb{E}[X_i^2] > 0$ for all i, i.e. none of the variables are almost surely 0. Also, define $S_i = X_1 + \cdots + X_i$ and $T_i = \mathbb{E}[X_1^2 + \cdots + X_i^2]$ for $0 \leq i \leq n$. Note that $T_0 = 0$ and $T_n = 1$.

We proceed by constructing a series of nested intervals $[a_{r,s}, b_{r,s}]$ and our analysis will be similar to that of Lemma 18 in [5]. We construct $a_{r,s}$ and $b_{r,s}$ for $0 \le r \le d = \Theta(\max_i \lg\left(\mathbb{E}[X_i^2]^{-1}\right))$ and $1 \le s \le 2^r$, as integers between 0 and n, inclusive. First define $a_{0,1} = 0$ and $b_{0,1} = n$. Next, we inductively define $a_{r,s}, b_{r,s}$. Define $a_{r+1,2s-1} := a_{r,s}$ and $b_{r+1,2s} := b_{r,s}$. Then, if there exists any index $a_{r,s} \le t \le b_{r,s}$ such that

$$0.45 \cdot \left|T_{b_{r,s}} - T_{a_{r,s}}\right| \le \left|T_t - T_{a_{r,s}}\right| \le 0.55 \cdot \left|T_{b_{r,s}} - T_{a_{r,s}}\right|,$$

let $a_{r+1,2s} = b_{r+1,2s-1} = t$ (if there are multiple such indices t, choose any one). Else, define $b_{r+1,2s-1}$ to be the largest index $t \ge a_{r,s}$ such that

$$\left|T_t - T_{a_{r,s}}\right| \le 0.45 \cdot \left|T_{b_{r,s}} - T_{a_{r,s}}\right|$$

and similarly define $a_{r+1,2s}$ to be the smallest index $t \le b_{r,s}$ such that

$$\left|T_t - T_{a_{r,s}}\right| \ge 0.55 \cdot \left|T_{b_{r,s}} - T_{a_{r,s}}\right|.$$

Note that in this case, $a_{r,2s} = b_{r,2s-1} + 1$.

It is clear that intervals are all nested in each other and for every r, all integers between 0 and n are in an interval $[a_{r,s}, b_{r,s}]$ for some s (possibly at an endpoint). Also, we always have $a_{r,0} \le b_{r,0} \le a_{r,1} \le \cdots \le b_{r,2^r}$, and any interval $[a_{r,s}, b_{r,s}]$ satisfies $T_{b_{r,s}} - T_{a_{r,s}} \le 0.55^r$. The previous point implies that since $d = \Theta(\max_i(\lg \mathbb{E}[X_i^2]^{-1}))$, every integer equals $a_{d,s} = b_{d,s}$ for some s.

We now call an interval $[a_{r,s}, b_{r,s}]$ *bad* if either s is odd and $b_{r,s} \ne a_{r,s+1}$ or s is even and $a_{r,s} \ne b_{r,s-1}$. Define the *rank* $q_{r,s}$ of a bad interval as the number of distinct $r' \le r$ such that $[a_{r,s}, b_{r,s}] \subseteq [a_{r',s'}, b_{r',s'}]$ for some bad interval $[a_{r',s'}, b_{r',s'}]$, which may equal $[a_{r,s}, b_{r,s}]$. Define the *relative rank* of a bad interval with respect to some interval $[a, b]$ as the number of distinct $r' \le r$ such that $[a_{r,s}, b_{r,s}] \subseteq [a_{r',s'}, b_{r',s'}] \subsetneq [a, b]$ for some bad interval $[a_{r',s'}, b_{r',s'}]$. Note that $[a_{r,2s-1}, b_{r,2s-1}]$ and $[a_{r,2s}, b_{r,2s}]$ are either both bad or both *good*, i.e. not bad. We now show the following:

▶ **Lemma 10.** *Given distinct bad intervals $[a_{r_i,s_i}, b_{r_i,s_i}]$ for $1 \le i \le \ell$ all contained in some interval $[a_{r,s}, b_{r,s}]$, where each interval has relative rank exactly q with respect to $[a_{r,s}, b_{r,s}]$,*

$$\sum_{i=1}^{\ell} \left(T_{b_{r_i,s_i}} - T_{a_{r_i,s_i}}\right) \le 0.9^q \cdot \left(T_{a_{r,s}} - T_{b_{r,s}}\right).$$

As an immediate consequence, given distinct bad intervals $[a_{r_i,s_i}, b_{r_i,s_i}]$ with absolute rank q,

$$\sum_{i=1}^{\ell} \left(T_{b_{r_i,s_i}} - T_{a_{r_i,s_i}}\right) \le 0.9^q.$$

Proof. First, note that the bad intervals cannot overlap, except at endpoints, as the only way for such intervals to overlap is for one to be contained in another, which would mean they have different ranks. Now, we prove this by induction on $b_{r,s} - a_{r,s}$. If $b_{r,s} - a_{r,s} = 1$, then for any value of q, this is quite straightforward, since there cannot exist bad intervals of nonzero length with positive relative rank. Now, given $b_{r,s} - a_{r,s} > 1$, then $a_{r,s} = a_{r+1,2s-1} \le b_{r+1,2s-1} \le a_{r+1,2s} \le b_{r+1,2s} = b_{r,s}$, and at least one of the two outer inequalities must be strict. If $b_{r+1,2s-1} = a_{r+1,2s}$, then neither $[a_{r+1,2s-1}, b_{r+1,2s-1}]$ nor $[a_{r+1,2s}, b_{r+1,2s}]$ are bad intervals. We can separately look at intervals which are subintervals of $[a_{r+1,2s-1}, b_{r+1,2s-1}]$ or $[a_{r+1,2s}, b_{r+1,2s}]$ to see which ones have rank q. By induction on $b_{r,s} - a_{r,s}$, the total length of the subintervals of relative rank q is at most

$$0.9^q \cdot (T_{b_{r+1,2s-1}} - T_{a_{r+1,2s-1}}) + 0.9^q \cdot (T_{b_{r+1,2s}} - T_{a_{r+1,2s}}) = 0.9^q \cdot (T_{b_{r,s}} - T_{a_{r,s}}).$$

If $b_{r+1,2s-1} \neq a_{r+1,2s}$, then if $q = 1$, we can only choose the subintervals $[a_{r+1,2s-1}, b_{r+1,2s-1}]$ and $[a_{r+1,2s}, b_{r+1,2s}]$, and clearly

$$(T_{b_{r+1,2s-1}} - T_{a_{r+1,2s-1}}) + (T_{b_{r+1,2s}} - T_{a_{r+1,2s}}) \leq (0.45+0.45) \cdot (T_{b_{r,s}} - T_{a_{r,s}}) = 0.9 \cdot (T_{b_{r,s}} - T_{a_{r,s}}).$$

If $q > 1$, we can separately look at intervals which are subintervals of $[a_{r+1,2s-1}, b_{r+1,2s-1}]$ and $[a_{r+1,2s}, b_{r+1,2s}]$ to see which ones have relative rank $q-1$, where we have to subtract one from the rank since $[a_{r+1,2s-1}, b_{r+1,2s-1}]$ and $[a_{r+1,2s}, b_{r+1,2s}]$ are both bad. Then, the total length of the subintervals of relative rank q is at most

$$\begin{aligned}
&0.9^{q-1} \cdot (T_{b_{r+1,2s-1}} - T_{a_{r+1,2s-1}}) + 0.9^{q-1} \cdot (T_{b_{r+1,2s}} - T_{a_{r+1,2s}}) \\
&\leq 0.9^{q-1} \cdot (0.45 + 0.45) \cdot (T_{b_{r,s}} - T_{a_{r,s}}) = 0.9^q (T_{b_{r,s}} - T_{a_{r,s}}).
\end{aligned}$$

◀

Next, for any λ, we bound the probability that there exists either a bad interval $[a_{r,s}, b_{r,s}]$ with rank q such that $|S_{b_{r,s}} - S_{a_{r,s}}| \geq 0.99^q \cdot \lambda$ or good intervals $[a_{r,2s-1}, b_{r,2s-1}], [a_{r,2s}, b_{r,2s}]$ such that $\min(|S_{b_{r,2s-1}} - S_{a_{r,2s-1}}|, |S_{b_{r,2s}} - S_{a_{r,2s}}|) \geq 0.99^r \cdot \lambda$. Note that by the Chebyshev inequality,

$$\mathbb{P}\left(|S_{b_{r,s}} - S_{a_{r,s}}| \geq 0.99^q \cdot \lambda\right) \leq \frac{T_{b_{r,s}} - T_{a_{r,s}}}{0.99^{2q} \cdot \lambda^2},$$

since $\mathbb{E}\left[(S_{b_{r,s}} - S_{a_{r,s}})^2\right] = T_{b_{r,s}} - T_{a_{r,s}}$ by pairwise independence. Therefore, the probability of us having this for any bad interval is at most

$$\sum_{q=1}^{\infty} \sum_{\substack{\text{bad interval} \\ \text{rank } q}} \frac{T_{b_{r,s}} - T_{a_{r,s}}}{0.99^{2q} \cdot \lambda^2} \leq \sum_{q=1}^{\infty} \frac{0.9^q}{0.99^{2q} \cdot \lambda^2} = O(\lambda^{-2}).$$

Next, note that for any good intervals $[a_{r,2s-1}, b_{r,2s-1}]$ and $[a_{r,2s}, b_{r,2s}]$, we have that

$$\begin{aligned}
&\mathbb{P}\left(\min\left(|S_{b_{r,2s-1}} - S_{a_{r,2s-1}}|, |S_{b_{r,2s}} - S_{a_{r,2s}}|\right) \geq \lambda \cdot 0.99^r\right) \\
&\leq \mathbb{P}\left((S_{b_{r,2s-1}} - S_{a_{r,2s-1}})^2 (S_{b_{r,2s}} - S_{a_{r,2s}})^2 \geq \lambda^4 \cdot 0.99^{4r}\right) \\
&\leq \frac{\mathbb{E}\left[(S_{b_{r,2s-1}} - S_{a_{r,2s-1}})^2 (S_{b_{r,2s}} - S_{a_{r,2s}})^2\right]}{\lambda^4 \cdot 0.99^{4r}} \\
&\leq \frac{(T_{b_{r,2s-1}} - T_{a_{r,2s-1}})(T_{b_{r,2s}} - T_{a_{r,2s}})}{\lambda^4 \cdot 0.99^{4r}} \leq \frac{0.55^{2r}}{\lambda^4 \cdot 0.99^{4r}}
\end{aligned}$$

using 4-wise independence of $X_1, \ldots, X_n$. Since there are at most 2^r such pairs of good intervals for any r, the probability of $|S_{b_{r,2s-1}} - S_{a_{r,2s-1}}|, |S_{b_{r,2s}} - S_{a_{r,2s}}|$ both being greater than $\lambda \cdot 0.99^r$ for any pair of good intervals, is at most

$$\sum_{r=1}^{\infty} 2^r \cdot \frac{0.55^{2r}}{\lambda^4 \cdot 0.99^{4r}} = O(\lambda^{-4}).$$

Finally, the probability of $|S_n - S_0| = |S_{b_{0,1}} - S_{a_{0,1}}| > \lambda$ is at most $\frac{\mathbb{E}[S_n^2]}{\lambda^2} = O(\lambda^{-2})$.

These imply the following result:

▶ **Lemma 11.** *The probability of there existing a bad interval $[a_{r,s}, b_{r,s}]$ with rank q such that $|S_{b_{r,s}} - S_{a_{r,s}}| \geq 0.99^q \cdot \lambda$, or good intervals $[a_{r,2s-1}, b_{r,2s-1}]$ and $[a_{r,2s}, b_{r,2s}]$ such that $|S_{b_{r,2s-1}} - S_{a_{r,2s-1}}|, |S_{b_{r,2s}} - S_{a_{r,2s}}|$ are both greater than $\lambda \cdot 0.99^r$, or of $|S_n - S_0| \geq \lambda$ is $O(\lambda^{-2})$.*

Next, we prove the following:

▶ **Lemma 12.** *For any $0 \le i \le n$, there exists a sequence $0 \le i_0, i_1, i_2, \ldots, i_d \le n$ with $i_0 = 0, i_d = i$, and a sequence of nested intervals $[a_{0,s_0}, b_{0,s_0}] \supset \cdots \supset [a_{d,s_d}, b_{d,s_d}]$ such that for any $1 \le j \le d-1$, i_j is an endpoint of the interval $[a_{j,s_j}, b_{j,s_j}]$ and of the interval $[a_{j-1,s_{j-1}}, b_{j-1,s_{j-1}}]$. Furthermore, for any $1 \le j \le d$, either $i_{j-1} = i_j$, or $[a_{j,s_j}, b_{j,s_j}]$ is a bad interval, or i_j equals $a_{j,2s} = b_{j,2s-1}$ and i_{j-1} is either $a_{j,2s-1}$ or $b_{j,2s}$ such that $|S_{i_j} - S_{i_{j-1}}| = \min(|S_{b_{j,2s}} - S_{a_{j,2s}}|, |S_{a_{j,2s-1}} - S_{a_{j,2s-1}}|)$. The intervals and values $i_0, \ldots, i_d$ may depend on the actual values of $X_1, \ldots, X_n$.*

Proof. We know that $i = i_d$ equals $a_{d,s_d} = b_{d,s_d}$ for some s_d, and thus must also equal either $a_{d-1,s_{d-1}}$ or $b_{d-1,s_{d-1}}$ for some s_{d-1}. If we are given i_{j+1} for some $1 \le j < d$, if i_{j+1} equals $a_{j,2s-1}$ or $b_{j,2s}$ for some s, then let $i_j = i_{j+1}$ which equals $a_{j-1,s}$ or $b_{j-1,s}$, respectively. If i_j equals $a_{j,2s}$ or $b_{j,2s-1}$ for some s, then if $a_{j,2s} = b_{j,2s-1}$, we can choose i_{j-1} accordingly as either $a_{j,2s-1} = a_{j-1,s}$ or $b_{j,2s} = b_{j-1,s}$ based on whether $|S_{b_{j,2s-1}} - S_{a_{j,2s-1}}|$ or $|S_{b_{j,2s}} - S_{a_{j,2s}}|$ is smaller. If $a_{j,2s} \ne b_{j,2s-1}$, then if $i_j = a_{j,2s}$ we choose $i_{j-1} = a_{j-1,s}$ and if $i_j = b_{j,2s-1}$ then we choose $i_{j-1} = b_{j-1,s}$. ◀

As a result, we have that if the conditions of Lemma 11 do not hold, which happens with probability $1 - O(\lambda^{-2})$, then for any i, then every $|S_i|$ satisfies

$$|S_i| \le \sum_{j=1}^{d} |S_{i_j} - S_{i_{j-1}}| \le \lambda + \sum_{q=1}^{\infty} 0.99^q \cdot \lambda + \sum_{r=1}^{\infty} 0.99^r \cdot \lambda = O(\lambda),$$

where I am using the fact that the intervals $[a_{j,s_j}, b_{j,s_j}]$ are nested in each other, so no two bad intervals can have the same rank.

In summary, we have with probability at least $1 - O(\lambda^{-2})$, the supremum of $|S_i| = |X_1 + \cdots + X_i|$ over all i doesn't exceed $O(\lambda)$, so we have proven Theorem 9. ◀

5 Open Problems

We note a few further directions that could be taken after this work. The first main open problem is whether 3-wise independent random walks on n steps have supremum distance bounded by $O(\sqrt{n})$ in expectation. As all 3-wise independent random walks are also 2-wise independent, we know that the supremum distance is bounded by $O(\sqrt{n}\log n)$ in expectation and the supremum square distance is bounded by $O(n\log^2 n)$ in expectation. However, we do not know if better bounds are true for 3-wise independent random walks. Likewise, for odd $k \ge 5$, we could ask if the kth moment of the supremum distance is bounded by $O(n^{k/2})$ in expectation, as we only know that the $(k-1)$th moment of the supremum distance is bounded by $O(n^{(k-1)/2})$ in expectation by Equation (5), since $k-1$ is even and at least 4.

Finally, we could ask how the constants increase as k grows. By our proof of Equation (5) and the constants in Khintchine's inequality [15], we know that the kth moment of the supremum distance of a random walk is $O(k)^{3k/2} \cdot n^{k/2}$. Therefore, we could ask if the constant $O(k)^{3k/2}$ could be improved if k grows with respect to n (such as if $k = \Theta(\log n)$).

References

1 Noga Alon, Yossi Matias, and Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. doi:10.1006/jcss.1997.1545.

2 Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An Introduction to MCMC for Machine Learning. *Machine Learning*, 50(1):5–43, January 2003. doi:10.1023/A:1020281327116.

3 Itai Benjamini, Gady Kozma, and Dan Romik. Random walks with k-wise independent increments. *Electron. Commun. Probab.*, 11:100–107, 2006. doi:10.1214/ECP.v11-1201.

4 Howard Berg. *Random Walks in Biology*. Princeton University Press, 1993.

5 Jarosław Błasiok. Optimal streaming and tracking distinct elements with high probability. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2432–2448, 2018. doi:10.1137/1.9781611975031.156.

6 Jarosław Błasiok, Jian Ding, and Jelani Nelson. Continuous Monitoring of ℓ_p Norms in Data Streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, pages 32:1–32:13, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.32.

7 Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

8 Vladimir Braverman, Stephen R. Chestnut, Nikita Ivkin, Jelani Nelson, Zhengyu Wang, and David P. Woodruff. BPTree: An ℓ_2 heavy hitters algorithm using constant memory. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017*, pages 361–376, 2017. doi:10.1145/3034786.3034798.

9 Vladimir Braverman, Stephen R. Chestnut, Nikita Ivkin, and David P. Woodruff. Beating CountSketch for heavy hitters in insertion streams. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 740–753, 2016. doi:10.1145/2897518.2897558.

10 Larry Carter and Mark N. Wegman. Universal Classes of Hash Functions. *J. Comput. Syst. Sci.*, 18(2):143–154, 1979. doi:10.1016/0022-0000(79)90044-8.

11 R. M. Dudley. The sizes of compact subsets of Hilbert space and continuity of Gaussian processes. *Journal of Functional Analysis*, 1(3):290–330, 1967. doi:10.1016/0022-1236(67)90017-1.

12 M. P. Etienne and P. Vallois. Approximation of the Distribution of the Supremum of a Centered Random Walk. Application to the Local Score. *Methodology And Computing In Applied Probability*, 6(3):255–275, September 2004. doi:10.1023/B:MCAP.0000026559.87023.ec.

13 Eugene F. Fama. Random Walks in Stock Market Prices. *Financial Analysts Journal*, 21(5):55–59, 1965. URL: http://www.jstor.org/stable/4469865.

14 Pierre-Gilles Gennes. *Scaling Concepts in Polymer Physics*. Cornell University Press, 1 edition, November 1979.

15 Uffe Haagerup. The best constants in the Khintchine inequality. *Studia Mathematica*, 70(3):231–283, 1981. URL: http://eudml.org/doc/218383.

16 A. Kolmogoroff. Über die summen durch den zufall bestimmter unabhängiger größen. *Mathematische Annalen*, 99:309–319, 1928. URL: http://eudml.org/doc/159258.

17 László Lovász. Random Walks on Graphs: A Survey, 1993.

18 Jelani Nelson. Chaining introduction with some computer science applications. *Bulletin of the EATCS*, 120, 2016. URL: http://eatcs.org/beatcs/index.php/beatcs/article/view/450.

19 Anna Pagh, Rasmus Pagh, and Milan Ružić. Linear Probing with Constant Independence. *SIAM J. Comput.*, 39(3):1107–1120, 2009. doi:10.1137/070702278.

20 Mihai Pătraşcu and Mikkel Thorup. On the k-Independence Required by Linear Probing and Minwise Independence. *ACM Trans. Algorithms*, 12(1):8:1–8:27, 2016. doi:10.1145/2716317.

21 Yao-Feng Ren and Han-Ying Liang. On the best constant in Marcinkiewicz–Zygmund inequality. *Statistics & Probability Letters*, 53(3):227–233, 2001. doi:10.1016/S0167-7152(01)00015-3.

22 Mathias Knudsen (via Jelani Nelson). Personal communication.

A Generalized Upper Bounds: Proof of Equations 4 and 5

Before we prove Equations (4) and (5), we construct $2^{-r/2}$-nets for $0 \leq r \leq 2\lg m + 1$ in a very similar way as in Theorem 1 in [8]. We define an *ε-net* to be a finite set of points $a_{r,0}, a_{r,1}, \ldots, a_{r,d_r}$ such that for every $z^{(t)}$, $||z^{(t)} - a_{r,s}||_2 \leq \varepsilon||z||_2$ for some $0 \leq s \leq d_r$. The constructions are defined identically for both equations. Define $a_{0,0} := z^{(0)}$ as the only element of the $2^{-0/2} = 1$-net. For $r \geq 1$, define $a_{r,0} = z^{(0)}$, and given $a_{r,s} = z^{(t_1)}$ then define $a_{r,s+1}$ as the smallest $t > t_1$ such that

$$||z^{(t)} - z^{(t_1)}||_2 > 2^{-r/2} \cdot ||z||_2^2,$$

unless such a t does not exist, in which case let $s = d_r$ and do not define $a_{r,s'}$ for any $s' > s$.

We define the set $A_r = \{a_{r,s} : 0 \leq s \leq d_r\}$. The following is directly true from our construction:

▶ **Proposition 13.** *For any $0 \leq t \leq m$ and fixed r, if $t_1 \leq t$ is the largest t_1 such that $z^{(t_1)} = a_{r,s}$ for some s, then $||z^{(t)} - z^{(t_1)}||_2 \leq 2^{-r/2} \cdot ||z||_2$. Consequently, $A_r = \{a_{r,0}, \ldots, a_{r,d_r}\}$ is a $2^{-r/2}$-net.*

The above proposition implies the following:

▶ **Proposition 14.** *For all $1 \leq t \leq m$, $z^{(t)} = a_{2\lg m+1,s}$ for some s.*

Proof. Let t_1 be the largest integer at most t such that $z^{(t_1)} = a_{2\lg m+1,s}$ for some s. Then, $||z^{(t)} - a_{2\lg m+1,s}||_2^2 \leq 2^{-(2\lg m+1)} \cdot ||z||_2^2 < 1$, which is clearly impossible unless $z^{(t)} = a_{2\lg m+1,s}$. ◀

Next, to prove Equations (4) and (5), we will need the Marcinkiewicz–Zygmund inequality (see for example [21]), which is a generalization of Khintchine's inequality (see for example [15]):

▶ **Theorem 15.** *For any even $k \geq 2$, there exists a constant B_k only depending on k such that for any fixed vector v and totally independent random variables $\overline{Y} = (Y_1, \ldots, Y_n)$,*

$$\mathbb{E}\left[\left(\sum_{i=1}^{n} Y_i\right)^k\right] \leq B_k \mathbb{E}\left[\left(\sum_{i=1}^{n} Y_i^2\right)^{k/2}\right].$$

This implies the following result:

▶ **Proposition 16.** *For any $k \geq 2$ and vector v, there exists a B_k only dependent on k such that*

$$\mathbb{E}\left[\langle v, \vec{X}\rangle\right]^k = \mathbb{E}\left[\left(\sum_{i=1}^{n} v_i X_i\right)^k\right] \leq B_k ||v||_2^k.$$

Proof. Since the expected value of $(\sum v_i X_i)^k$ is only dependent on k-wise independence, we can assume that the X_i's are totally independent but have the same marginal distribution. This implies

$$\mathbb{E}\left[\left(\sum_{i=1}^{n} v_i X_i\right)^k\right] \leq B_k \mathbb{E}\left[\left(\sum_{i=1}^{n} v_i^2 X_i^2\right)^{k/2}\right]$$

by Theorem 15. However, we know that $\mathbb{E}[X_i^{2d}] \leq 1$ for all i and all $1 \leq d \leq k/2$, since $\mathbb{E}[X_i^k] \leq 1$ and $\mathbb{E}[X_i^{2d}]^{k/d} \leq \mathbb{E}[X_i^k]$ by Jensen's inequality, so simply expanding and using independence and linearity of expectation gets us the desired result. ◀

We now prove equations (4) and (5).

Proof of Equation (4). For $r \geq 1$ and s, suppose $a_{r,s} = z^{(t)}$ and $t_1 \leq t$ is the largest index such that $z^{(t_1)} \in A_{r-1}$. Then, define $f(s,t)$ to be the index s' such that $z^{(t_1)} = a_{r-1,s'}$. Consider the quadratic form

$$\sum_{r=1}^{2\lg m+1} \sum_{s=0}^{d_r} \langle (a_{r,s} - a_{r-1,f(r,s)}), \vec{X} \rangle^2.$$

By Proposition 13, $||a_{r,s} - a_{r-1,f(r,s)}||_2 \leq 2^{-(r-1)/2} \cdot ||z||_2$. Thus, by Proposition 16, we get the expected value of the quadratic form equals

$$\sum_{r=1}^{2\lg m+1} \sum_{s=0}^{d_r} \mathbb{E}[\langle (a_{r,2s+1} - a_{r,2s}), \vec{X} \rangle^2] \leq B_2 \sum_{r=1}^{2\lg m+1} \sum_{s=0}^{d_r} ||a_{r,2s+1} - a_{r,2s}||_2^2$$

$$\leq B_2 \sum_{r=1}^{2\lg m+1} \left(2^r \cdot 2^{-(r-1)} ||z||_2^2 \right) \leq 2B_2(2\lg m + 1)(||z||_2^2).$$

Here, I am using the fact that an ε-net has size at most ε^{-2}, which is easy to see since $z^{(0)}, \ldots, z^{(m)}$ is tracking an insertion stream (it is proven, for example, in Theorem 1 of [8]), and thus $d_r \leq 2^r$.

Now, for any $0 \leq i \leq n$, consider $z^{(i)}$ and let $z^{(i)} = a_{2\lg m+1,s}$. Then, define $s_r = s$ if $r = 2\lg m + 1$ and $s_{r-1} = f(r, s_r)$ for $1 \leq r \leq 2\lg m + 1$. Note that $s_0 = 0$ and for any $r \geq 1$, if $a_{r,s_r} \in A_{r-1}$, then $a_{r,s_r} = a_{r-1,s_{r-1}}$. Thus, each $\langle (a_{r,s_r} - a_{r-1,s_{r-1}}), \vec{X} \rangle^2$ for $1 \leq 2\lg m + 1$ is either 0 (because $a_{r,s_r} - a_{r-1,s_{r-1}} = 0$) or is a summand in our quadratic form. Therefore,

$$\sum_{r=1}^{2\lg m+1} \sum_{s=0}^{d_r} \langle (a_{r,s} - a_{r,f(r,s)}), \vec{X} \rangle^2 \geq \sum_{r=1}^{2\lg m+1} \langle (a_{r,s_r} - a_{r-1,s_{r-1}}), \vec{X} \rangle^2 \geq \frac{1}{2\lg m + 1} \cdot \langle z^{(i)}, \vec{X} \rangle^2,$$

with the last inequality true since $a_{2\lg m+1,s_{2\lg m+1}} = z^{(i)}$, $a_{0,s_0} = z^{(0)}$, and by the Cauchy-Schwarz inequality. As this is true for all i, taking the supremum over i and then expected values gives us

$$2B_2(2\lg m + 1)(||z||_2^2) \geq \mathbb{E}\left[\sum_{r=1}^{2\lg m+1} \sum_{s=0}^{d_r} \langle (a_{r,2s+1} - a_{r,2s}), \vec{X} \rangle^2 \right]$$
$$\geq \frac{1}{2\lg m + 1} \cdot \sup_i \mathbb{E}\left[\langle z^{(i)}, \vec{X} \rangle^2 \right],$$

and therefore,

$$\mathbb{E}\left[\langle z^{(i)}, \vec{X} \rangle^2 \right] = O\left(||z||_2^2 \cdot \lg^2 m \right). \qquad \blacktriangleleft$$

Proof of Equation (5). Consider the form

$$\sum_{r=1}^{2\lg m+1} 2^{r/2} \sum_{s=0}^{d_r} \langle (a_{r,s} - a_{r-1,f(r,s)}), \vec{X} \rangle^k,$$

with $f(r,s)$ defined as in the proof of Equation (4). We again note that by Proposition 13, $||a_{r,s} - a_{r-1,f(r,s)}||_2 \leq 2^{-(r-1)/2} \cdot ||z||_2$. Thus, by Proposition 16, we get the expected value of the form equals

$$\sum_{r=1}^{2\lg m+1} 2^{r/2} \sum_{s=0}^{d_r} \mathbb{E}[\langle (a_{r,s} - a_{r-1,f(r,s)}), \vec{X} \rangle^k] \leq B_k \sum_{r=1}^{2\lg m+1} 2^{r/2} \sum_{s=0}^{d_r} ||a_{r,s} - a_{r-1,f(r,s)}||_2^k$$

$$\leq B_k \sum_{r=0}^{2\lg m+1} 2^{r/2} \cdot 2^r \cdot 2^{-(r-1)k/2} ||z||_2^k \leq B_k \cdot 2^{k/2} \sum_{r=0}^{\infty} 2^{r(3-k)/2} ||z||_2^k = O(2^{k/2} B_k) ||z||_2^k,$$

since $k \geq 4$. Again, I am using the fact that $d_r \leq 2^r$ as an ε-net has size at most ε^{-2}.

Now, for any $0 \leq i \leq n$, suppose s satisfies $z^{(i)} = a_{2\lg m+1,s}$. define $s_r = s$ if $r = 2\lg m+1$ and $s_{r-1} = f(r, s_r)$ for $1 \leq r \leq 2\lg m + 1$. Then, similarly to in the proof of Equation (4),

$$\sum_{r=1}^{2\lg m+1} 2^{r/2} \sum_{s=0}^{d_r} \langle (a_{r,s} - a_{r-1,f(r,s)}), \vec{X} \rangle^k \geq \sum_{r=1}^{2\lg m+1} 2^{r/2} \langle (a_{r,s_r} - a_{r-1,s_{r-1}}), \vec{X} \rangle^k$$
$$\geq \Omega(k^{-1})^k \cdot \langle z^{(i)}, \vec{X} \rangle^k.$$

The last inequality requires justification, specifically that if $x_1 + \cdots + x_{2\lg m+1} = 1$, $\sum 2^{r/2} x_r^k = \Omega(k^{-1})^k$. This is sufficient since we can let $x_r = \langle (a_{r,s_r} - a_{r-1,s_{r-1}}), \vec{X} \rangle$. To prove this, define $x'_1, x'_2, \ldots, x'_{2\lg m+1}$ such that $x'_1 + \cdots + x'_{2\lg m+1} = 1$ and $x'_1 > \cdots > x'_{2\lg m+1}$ are in a geometric series with common ratio $2^{-1/(2k)} = 1 - \Theta(1/k)$. Then, note that for any $x_1, \ldots, x_{2\lg m+1}$ such that $x_1 + \cdots + x_{2\lg m+1} = 1$, $x_i \geq x'_i$ for some i. But note that $(x'_r)^k 2^{r/2}$ are equal for all r because of our geometric series, and equals $(x'_1)^k = \Omega(k^{-1})^k$ since $x'_1 = \Omega(k^{-1})^k$ is clearly true. Thus, $\sum 2^{r/2} x_r^k \geq 2^{i/2} (x'_i)^k = \Omega(k^{-1})^k$, so we are done.

As this is true for all i, we can take the supremum over i and then take expected values to get

$$2^{k/2} B_k \cdot ||z||_2^k \geq \mathbb{E}\left[\sum_{r=1}^{2\lg m+1} 2^{r/2} \sum_{s=0}^{d_r} \langle (a_{r,s} - a_{r-1,f(r,s)}), \vec{X} \rangle^k \right] \geq \Omega(k^{-1})^k \cdot \sup_i \mathbb{E}\left[\langle z^{(i)}, \vec{X} \rangle^k \right]$$

and therefore, for a fixed k,

$$\mathbb{E} \sup_i \left[\langle z^{(i)}, \vec{X} \rangle^k \right] = O\left(||z||_2^k \right). \qquad \blacktriangleleft$$

Optimal Convergence Rate of Hamiltonian Monte Carlo for Strongly Logconcave Distributions

Zongchen Chen
School of Computer Science, Georgia Institute of Technology, USA
chenzongchen@gatech.edu

Santosh S. Vempala
School of Computer Science, Georgia Institute of Technology, USA
vempala@gatech.edu

Abstract

We study *Hamiltonian Monte Carlo* (HMC) for sampling from a strongly logconcave density proportional to e^{-f} where $f : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex and L-smooth (the condition number is $\kappa = L/\mu$). We show that the relaxation time (inverse of the spectral gap) of ideal HMC is $O(\kappa)$, improving on the previous best bound of $O(\kappa^{1.5})$; we complement this with an example where the relaxation time is $\Omega(\kappa)$. When implemented using a nearly optimal ODE solver, HMC returns an ε-approximate point in 2-Wasserstein distance using $\widetilde{O}((\kappa d)^{0.5}\varepsilon^{-1})$ gradient evaluations per step and $\widetilde{O}((\kappa d)^{1.5}\varepsilon^{-1})$ total time.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains; Theory of computation → Design and analysis of algorithms

Keywords and phrases logconcave distribution, sampling, Hamiltonian Monte Carlo, spectral gap, strong convexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.64

Category RANDOM

Funding *Zongchen Chen*: This work was supported in part by CCF-1563838 and CCF-1617306. *Santosh S. Vempala*: This work was supported in part by CCF-1563838, CCF-1717349 and DMS-1839323.

Acknowledgements We are grateful to Yin Tat Lee for helpful discussions.

1 Introduction

Sampling logconcave densities is a basic problem that arises in machine learning, statistics, optimization, computer science and other areas. The problem is described as follows. Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex function. Our goal is to sample from the density proportional to $e^{-f(x)}$. We study *Hamiltonian Monte Carlo* (HMC), one of the most widely-used *Markov chain Monte Carlo* (MCMC) algorithms for sampling from a probability distribution. In many settings, HMC is believed to outperform other MCMC algorithms such as the Metropolis-Hastings algorithm or Langevin dynamics. In terms of theory, rapid mixing has been established for HMC in recent papers [9, 10, 13, 14, 15] under various settings. However, in spite of much progress, there is a gap between known upper and lower bounds even in the basic setting when f is strongly convex (e^{-f} is strongly logconcave) and has a Lipschitz gradient.

Many sampling algorithms such as the Metropolis-Hastings algorithm or Langevin dynamics maintain a position $x = x(t)$ that changes with time, so that the distribution of x will eventually converge to the desired distribution, i.e., proportional to $e^{-f(x)}$. In HMC, besides the position $x = x(t)$, we also maintain a velocity $v = v(t)$. In the simplest Euclidean setting, the Hamiltonian $H(x, v)$ is defined as

$$H(x, v) = f(x) + \frac{1}{2}\|v\|^2.$$


© Zongchen Chen and Santosh S. Vempala;
licensed under Creative Commons License CC-BY



Then in every step the pair (x, v) is updated using the following system of differential equations for a fixed time interval T:

$$\begin{cases} \dfrac{\mathrm{d}x(t)}{\mathrm{d}t} = \dfrac{\partial H(x,v)}{\partial v} = v(t), \\ \dfrac{\mathrm{d}v(t)}{\mathrm{d}t} = -\dfrac{\partial H(x,v)}{\partial x} = -\nabla f(x(t)). \end{cases} \tag{1}$$

The initial position $x(0) = x_0$ is the position from the last step, and the initial velocity $v(0) = v_0$ is chosen randomly from the standard Gaussian distribution $N(0, I)$. The updated position is $x(T)$ where T can be thought of as the step-size. It is well-known that the stationary distribution of HMC is the density proportional to e^{-f}. Observe that

$$\frac{\mathrm{d}H(x,v)}{\mathrm{d}t} = \frac{\partial H(x,v)}{\partial x}x'(t) + \frac{\partial H(x,v)}{\partial v}v'(t) = 0,$$

so the Hamiltonian $H(x, v)$ does not change with t. We can also write (1) as the following ordinary differential equation (ODE):

$$x''(t) = -\nabla f(x(t)), \quad x(0) = x_0, \quad x'(0) = v_0. \tag{2}$$

We state HMC explicitly as the following algorithm.

Algorithm 1 Hamiltonian Monte Carlo algorithm.

Input: $f : \mathbb{R}^d \to \mathbb{R}$ that is μ-strongly convex and L-smooth, ε the error parameter.

1. Set starting point $x^{(0)}$, step-size T, number of steps N, and ODE error tolerance δ.
2. For $k = 1, \ldots, N$:
   a. Let $v \sim N(0, I)$;
   b. Denote by $x(t)$ the solution to (1) with initial position $x(0) = x^{(k-1)}$ and initial velocity $v(0) = v$. Use the ODE solver to find a point $x^{(k)}$ such that
   $$\left\| x^{(k)} - x(T) \right\| \le \delta.$$
3. Output $x^{(N)}$.

In our analysis, we first consider *ideal* HMC where in every step we have the exact solution to the ODE (1) and neglect the numerical error from solving the ODEs or integration ($\delta = 0$).

1.1 Preliminaries

We recall standard definitions here. Let $f : \mathbb{R}^d \to \mathbb{R}$ be a continuously differentiable function. We say f is μ-strongly convex if for all $x, y \in \mathbb{R}^d$,

$$f(y) \ge f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \|y - x\|^2 .$$

We say f is L-smooth if ∇f is L-Lipschitz; i.e., for all $x, y \in \mathbb{R}^d$,

$$\|\nabla f(x) - \nabla f(y)\| \le L \|x - y\| .$$

If f is μ-strongly convex and L-smooth, then the condition number of f is $\kappa = L/\mu$.

Consider a discrete-time *reversible* Markov chain $\mathcal{M}$ on $\mathbb{R}^d$ with stationary distribution π. Let

$$L_2(\pi) = \left\{ f : \mathbb{R}^d \to \mathbb{R} \,\middle|\, \int_{\mathbb{R}^d} f(x)^2 \pi(\mathrm{d}x) < \infty \right\}$$

be the Hilbert space with inner product

$$\langle f, g \rangle = \int_{\mathbb{R}^d} f(x) g(x) \pi(\mathrm{d}x)$$

for $f, g \in L_2(\pi)$. Denote by P the transition kernel of $\mathcal{M}$. We can view P as a self-adjoint operator from $L_2(\pi)$ to itself: for $f \in L_2(\pi)$,

$$(Pf)(x) = \int_{\mathbb{R}^d} f(y) P(x, \mathrm{d}y).$$

Let $L_2^0(\pi) = \{f \in L_2(\pi) : \int_{\mathbb{R}^d} f(x)\pi(\mathrm{d}x) = 0\}$ be a closed subspace of $L_2(\pi)$. The (absolute) *spectral gap* of P is defined to be

$$\gamma(P) = 1 - \sup_{f \in L_2^0(\pi)} \frac{\|Pf\|}{\|f\|} = 1 - \sup_{\substack{f \in L_2^0(\pi) \\ \|f\|=1}} |\langle Pf, f \rangle|.$$

The relaxation time of P is

$$\tau_{\mathrm{rel}}(P) = \frac{1}{\gamma(P)}.$$

Let ν_1, ν_2 be two distributions on $\mathbb{R}^d$. The 2-Wasserstein distance between ν_1 and ν_2 is defined as

$$W_2(\nu_1, \nu_2) = \left(\inf_{(X,Y) \in \mathcal{C}(\nu_1, \nu_2)} \mathbb{E}\left[\|X - Y\|^2 \right] \right)^{1/2},$$

where $\mathcal{C}(\nu_1, \nu_2)$ is the set of all couplings of ν_1 and ν_2.

1.2 Related work

Various versions of Langevin dynamics have been studied in many recent papers, see [5, 6, 21, 17, 7, 4, 3, 2, 8, 20, 19, 12]. The convergence rate of HMC is also studied recently in [9, 10, 13, 14, 15, 18]. The first bound for our setting was obtained by Mangoubi and Smith [13], who gave an $O(\kappa^2)$ bound on the convergence rate of ideal HMC.

▶ **Theorem 1** ([13, Theorem 1]). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a twice differentiable function such that $\mu I \preceq \nabla^2 f(x) \preceq LI$ for all $x \in \mathbb{R}^d$. Then the relaxation time of ideal HMC for sampling from the density $\propto e^{-f}$ with step-size $T = \sqrt{\mu}/(2\sqrt{2}L)$ is $O(\kappa^2)$.*

This was improved by [9], which showed a bound of $O(\kappa^{1.5})$. They also gave a nearly optimal method for solving the ODE that arises in the implementation of HMC.

▶ **Theorem 2** ([9, Lemma 1.8]). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a twice differentiable function such that $\mu I \preceq \nabla^2 f(x) \preceq LI$ for all $x \in \mathbb{R}^d$. Then the relaxation time of ideal HMC for sampling from the density $\propto e^{-f}$ with step-size $T = \mu^{1/4}/(2L^{3/4})$ is $O(\kappa^{1.5})$.*

Both papers suggest that the correct bound is linear in κ: [13] says linear is the best one can expect while [9] shows that there *exists* a choice of step-sizes (time for running the ODE) that might achieve a linear rate (Lemma 1.8, second part); however it was far from clear how to determine these step-sizes algorithmically.

Other papers focus on various aspects and use stronger assumptions (e.g., bounds on higher-order gradients) to get better bounds on the overall convergence time or the number of gradient evaluations in some ranges of parameters. For example, [15] shows that the dependence on dimension for the number of gradient evaluations can be as low as $d^{1/4}$ with suitable regularity assumptions (and higher dependence on the condition number). We note also that sampling logconcave functions is a polynomial-time solvable problem, without the assumptions of strong convexity or gradient Lipschitzness, and even when the function e^{-f} is given only by an oracle with no access to gradients [1, 11]. The Riemannian version of HMC provides a faster polynomial-time algorithm for uniformly sampling polytopes [10]. However, the dependence on the dimension is significantly higher for these algorithms, both for the contraction rate and the time per step.

1.3 Results

In this paper, we show that the relaxation time of ideal HMC is $\Theta(\kappa)$ for strongly logconcave functions with Lipschitz gradient.

▶ **Theorem 3.** *Suppose that f is μ-strongly convex and L-smooth. Then the relaxation time (inverse of spectral gap) of ideal HMC for sampling from the density $\propto e^{-f}$ with step-size $T = 1/(2\sqrt{L})$ is $O(\kappa)$, where $\kappa = L/\mu$ is the condition number.*

We remark that the only assumption we made about f is strongly convexity and smoothness (in particular, we do not require that f is twice differentiable, which is assumed in both [9] and [13]).

We also establish a matching lower bound on the relaxation time of ideal HMC, implying the tightness of Theorem 3.

▶ **Theorem 4.** *For any $0 < \mu \le L$, there exists a μ-strongly convex and L-smooth function f, such that the relaxation time of ideal HMC for sampling from the density $\propto e^{-f}$ with step-size $T = O(1/\sqrt{L})$ is $\Omega(\kappa)$, where $\kappa = L/\mu$ is the condition number.*

Using the nearly optimal ODE solver from [9], we obtain the following convergence rate in 2-Wasserstein distance for the HMC algorithm. We note that since our new convergence rate allows larger steps, the ODE solver is run for a longer time step.

▶ **Theorem 5.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a twice differentiable function such that $\mu I \preceq \nabla^2 f(x) \preceq LI$ for all $x \in \mathbb{R}^d$. Let $\pi \propto e^{-f}$ be the target distribution, and let π_{HMC} be the distribution of the output of HMC with starting point $x^{(0)} = \arg\min_x f(x)$, step-size $T = 1/(16000\sqrt{L})$, and ODE error tolerance $\delta = \sqrt{\mu} T^2 \varepsilon/16$. For any $0 < \varepsilon < \sqrt{d}$, if we run HMC for $N = O\left(\kappa \log(d/\varepsilon)\right)$ steps where $\kappa = L/\mu$, then we have*

$$W_2(\pi_{\text{HMC}}, \pi) \le \frac{\varepsilon}{\sqrt{\mu}}.$$

Each step takes $O\left(\sqrt{\kappa} d^{3/2} \varepsilon^{-1} \log(\kappa d/\varepsilon)\right)$ time and $O\left(\sqrt{\kappa d} \varepsilon^{-1} \log(\kappa d/\varepsilon)\right)$ evaluations of ∇f, amortized over all steps.

The comparison of convergence rates, running times and numbers of gradient evaluations is summarized in the following table with polylog factors omitted.

reference	convergence rate	# gradients	total time
[13]	κ^2	$\kappa^{6.5}d^{0.5}$	$\kappa^{6.5}d^{1.5}$
[9]	$\kappa^{1.5}$	$\kappa^{1.75}d^{0.5}$	$\kappa^{1.75}d^{1.5}$
this paper	κ	$\kappa^{1.5}d^{0.5}$	$\kappa^{1.5}d^{1.5}$

2 Convergence of ideal HMC

In this section we show that the spectral gap of ideal HMC is $\Omega(1/\kappa)$, and thus prove Theorem 3. We first show a contraction bound for ideal HMC, which roughly says that the distance of two points is shrinking after one step of ideal HMC.

▶ **Lemma 6** (Contraction bound). *Suppose that f is μ-strongly convex and L-smooth. Let $x(t)$ and $y(t)$ be the solution to (1) with initial positions $x(0)$, $y(0)$ and initial velocities $x'(0) = y'(0)$. Then for $0 \le t \le 1/(2\sqrt{L})$ we have*

$$\|x(t) - y(t)\|^2 \le \left(1 - \frac{\mu}{4}t^2\right)\|x(0) - y(0)\|^2.$$

In particular, by setting $t = T = 1/(c\sqrt{L})$ for some constant $c \ge 2$ we get

$$\|x(T) - y(T)\|^2 \le \left(1 - \frac{1}{4c^2\kappa}\right)\|x(0) - y(0)\|^2$$

where $\kappa = L/\mu$.

Proof. Consider the two ODEs for HMC:

$$\begin{cases} x'(t) = u(t); \\ u'(t) = -\nabla f(x(t)). \end{cases} \quad \text{and} \quad \begin{cases} y'(t) = v(t); \\ v'(t) = -\nabla f(y(t)). \end{cases}$$

with initial points $x(0), y(0)$ and initial velocities $u(0) = v(0)$. For the sake of brevity, we shall write $x = x(t)$, $y = y(t)$, $u = u(t)$, $v = v(t)$ and omit the variable t, as well as letting $x_0 = x(0)$, $y_0 = y(0)$. We are going to show that

$$\|x - y\|^2 \le \left(1 - \frac{\mu}{4}t^2\right)\|x_0 - y_0\|^2$$

for all $0 \le t \le 1/(2\sqrt{L})$.

Consider the derivative of $\frac{1}{2}\|x - y\|^2$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{2}\|x - y\|^2\right) = \langle x' - y', x - y\rangle = \langle u - v, x - y\rangle. \tag{3}$$

Taking derivative on both sides, we get

$$\begin{aligned} \frac{\mathrm{d}^2}{\mathrm{d}t^2}\left(\frac{1}{2}\|x - y\|^2\right) &= \langle u' - v', x - y\rangle + \langle u - v, x' - y'\rangle \\ &= -\langle \nabla f(x) - \nabla f(y), x - y\rangle + \|u - v\|^2 \\ &= -\rho\|x - y\|^2 + \|u - v\|^2, \end{aligned} \tag{4}$$

where we define

$$\rho = \rho(t) = \frac{\langle \nabla f(x) - \nabla f(y), x - y \rangle}{\|x - y\|^2}.$$

Since f is μ-strongly convex and L-smooth, we have $\mu \le \rho \le L$ for all $t \ge 0$.

We will upper bound the term $-\rho \|x - y\|^2 + \|u - v\|^2$, while keeping its dependency on ρ. To lower bound $\|x - y\|^2$, we use the following crude bound.

▷ Claim 7 (Crude bound). For all $0 \le t \le 1/(2\sqrt{L})$ we have

$$\frac{1}{2} \|x_0 - y_0\|^2 \le \|x - y\|^2 \le 2 \|x_0 - y_0\|^2 . \tag{5}$$

The proof of this claim is postponed to Section 2.1.

Next we derive an upper bound on $\|u - v\|^2$. The derivative of $\|u - v\|$ is given by

$$\begin{aligned} \|u - v\| \left(\frac{\mathrm{d}}{\mathrm{d}t} \|u - v\| \right) &= \frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{1}{2} \|u - v\|^2 \right) \\ &= \langle u' - v', u - v \rangle \\ &= - \langle \nabla f(x) - \nabla f(y), u - v \rangle . \end{aligned}$$

Thus, its absolute value is upper bounded by

$$\left| \frac{\mathrm{d}}{\mathrm{d}t} \|u - v\| \right| = \frac{|- \langle \nabla f(x) - \nabla f(y), u - v \rangle|}{\|u - v\|} \le \|\nabla f(x) - \nabla f(y)\| .$$

Since f is L-smooth and convex, we have

$$\|\nabla f(x) - \nabla f(y)\|^2 \le L \langle \nabla f(x) - \nabla f(y), x - y \rangle = L\rho \|x - y\|^2 \le 2L\rho \|x_0 - y_0\|^2 ,$$

where the last inequality follows from the crude bound (5). Then, using the fact that $u_0 = v_0$ and the Cauchy-Schwarz inequality, we can upper bound $\|u - v\|^2$ by

$$\begin{aligned} \|u - v\|^2 &\le \left(\int_0^t \left| \frac{\mathrm{d}}{\mathrm{d}s} \|u - v\| \right| \mathrm{d}s \right)^2 \\ &\le \left(\int_0^t \sqrt{2L\rho} \|x_0 - y_0\| \, \mathrm{d}s \right)^2 \\ &\le 2Lt \left(\int_0^t \rho \, \mathrm{d}s \right) \|x_0 - y_0\|^2 . \end{aligned}$$

Define the function

$$P = P(t) = \int_0^t \rho \, \mathrm{d}s,$$

so $P(t)$ is nonnegative and monotone increasing, with $P(0) = 0$. Also we have $\mu t \le P(t) \le Lt$ for all $t \ge 0$. Then,

$$\|u - v\|^2 \le 2LtP \|x_0 - y_0\|^2 . \tag{6}$$

Plugging (5) and (6) into (4), we deduce that

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2} \left(\frac{1}{2} \|x - y\|^2 \right) \le -\rho \left(\frac{1}{2} \|x_0 - y_0\|^2 \right) + 2LtP \|x_0 - y_0\|^2 .$$

If we define

$$\alpha(t) = \frac{1}{2}\|x-y\|^2,$$

then we have

$$\alpha''(t) \le -\alpha(0)\big(\rho(t) - 4LtP(t)\big).$$

Integrating both sides and using $\alpha'(0) = 0$, we obtain

$$\begin{aligned}\alpha'(t) &= \int_0^t \alpha''(s)\mathrm{d}s \\ &\le -\alpha(0)\left(\int_0^t \rho(s)\mathrm{d}s - 4L\int_0^t sP(s)\mathrm{d}s\right) \\ &\le -\alpha(0)\left(P(t) - 4LP(t)\int_0^t s\mathrm{d}s\right) \\ &= -\alpha(0)P(t)\left(1-2Lt^2\right),\end{aligned}$$

where the second inequality is due to the monotonicity of $P(s)$. Since for all $0 \le t \le 1/(2\sqrt{L})$ we have $P(t) \ge \mu t$ and $1 - 2Lt^2 \ge 1/2$, we deduce that

$$\alpha'(t) \le -\alpha(0)\frac{\mu}{2}t.$$

Finally, one more integration yields

$$\alpha(t) = \alpha(0) + \int_0^t \alpha'(s)\mathrm{d}s \le \alpha(0)\left(1 - \frac{\mu}{4}t^2\right),$$

and the theorem follows. ◀

Proof of Theorem 3. Lemma 6 implies that for any constant $c \ge 2$, the Ricci curvature of ideal HMC with step-size $T = 1/(c\sqrt{L})$ is at least $1/(8c^2\kappa)$. Then, it follows from [16, Proposition 29] that the spectral gap of ideal HMC is at least $1/(8c^2\kappa)$. Hence, the relaxation time is upper bounded by $8c^2\kappa = O(\kappa)$. ◀

2.1 Proof of Claim 7

We present the proof of Claim 7 in this section. We remark that a similar crude bound was established in [9] for general matrix ODEs. Here we prove the crude bound specifically for the Hamiltonian ODE, but without assuming that f is twice differentiable.

Proof of Claim 7. We first derive a crude upper bound on $\|u-v\|$. Since f is L-smooth, we have

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\|u-v\| &= \frac{-\langle \nabla f(x) - \nabla f(y), u-v\rangle}{\|u-v\|} \\ &\le \|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|.\end{aligned}$$

Then from $u_0 = v_0$ we get

$$\|u-v\| = \int_0^t \left(\frac{\mathrm{d}}{\mathrm{d}s}\|u-v\|\right)\mathrm{d}s \le L\int_0^t \|x-y\|\,\mathrm{d}s.$$

To obtain the upper bound for $\|x-y\|$, we first bound its derivative by

$$\left|\frac{\mathrm{d}}{\mathrm{d}t}\|x-y\|\right| = \frac{|\langle u-v, x-y\rangle|}{\|x-y\|} \le \|u-v\| \le L\int_0^t \|x-y\|\,\mathrm{d}s. \tag{7}$$

Therefore,

$$\begin{aligned}\|x-y\| &= \|x_0-y_0\| + \int_0^t \left(\frac{\mathrm{d}}{\mathrm{d}s}\|x-y\|\right)\mathrm{d}s\\ &\le \|x_0-y_0\| + L\int_0^t\int_0^s \|x-y\|\,\mathrm{d}r\mathrm{d}s\\ &= \|x_0-y_0\| + L\int_0^t (t-s)\|x-y\|\,\mathrm{d}s.\end{aligned}$$

We then deduce from [9, Lemma A.5] that

$$\|x-y\| \le \|x_0-y_0\|\cosh\left(\sqrt{L}t\right) \le \sqrt{2}\,\|x_0-y_0\|, \tag{8}$$

where we use the fact that $\cosh(\sqrt{L}t) \le \cosh(1/2) \le \sqrt{2}$.

Next, we deduce from (7) and (8) that

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\|x-y\| &\ge -L\int_0^t \|x-y\|\,\mathrm{d}s\\ &\ge -L\|x_0-y_0\|\int_0^t \cosh\left(\sqrt{L}s\right)\mathrm{d}s\\ &= -\sqrt{L}\|x_0-y_0\|\sinh\left(\sqrt{L}s\right).\end{aligned}$$

Thus, we obtain

$$\begin{aligned}\|x-y\| &= \|x_0-y_0\| + \int_0^t \left(\frac{\mathrm{d}}{\mathrm{d}s}\|x-y\|\right)\mathrm{d}s\\ &\ge \|x_0-y_0\| - \sqrt{L}\|x_0-y_0\|\int_0^t \sinh\left(\sqrt{L}s\right)\mathrm{d}s\\ &= \|x_0-y_0\|\left(2-\cosh\left(\sqrt{L}t\right)\right) \ge \frac{1}{\sqrt{2}}\|x_0-y_0\|,\end{aligned}$$

where we use $2-\cosh(\sqrt{L}t) \ge 2-\cosh(1/2) \ge 1/\sqrt{2}$. ◁

3 Lower bound for ideal HMC

In this section, we show that the relaxation time of ideal HMC can achieve $\Theta(\kappa)$ for some μ-strongly convex and L-smooth function, and thus prove Theorem 4.

Consider a two-dimensional quadratic function:

$$f(x_1,x_2) = \frac{x_1^2}{2\sigma_1^2} + \frac{x_2^2}{2\sigma_2^2},$$

where $\sigma_1 = 1/\sqrt{\mu}$ and $\sigma_2 = 1/\sqrt{L}$. Thus, f is μ-strongly convex and L-smooth. The probability density ν proportional to e^{-f} is essentially the bivariate Gaussian distribution: for $(x_1,x_2)\in\mathbb{R}^2$,

$$\nu(x_1,x_2) = \frac{1}{2\pi\sigma_1\sigma_2}\exp\left(-\frac{x_1^2}{2\sigma_1^2} - \frac{x_2^2}{2\sigma_2^2}\right).$$

The following lemma shows that ideal HMC for the bivariate Gaussian distribution ν has relaxation time $\Omega(\kappa)$, and then Theorem 4 follows immediately.

▶ **Lemma 8.** *For any constant $c > 0$, the relaxation time of ideal HMC for sampling from ν with step-size $T = 1/(c\sqrt{L})$ is at least $2c^2\kappa$.*

Proof. The Hamiltonian curve for f is given by the ODE

$$(x_1'', x_2'') = -\nabla f(x_1, x_2) = \left(-\frac{x_1}{\sigma_1^2}, -\frac{x_2}{\sigma_2^2}\right)$$

with initial position $(x_1(0), x_2(0))$ and initial velocity $(x_1'(0), x_2'(0))$ from the bivariate standard Gaussian $N(0, I)$. Observe that $\nu = \nu_1 \otimes \nu_2$ is a product distribution of the two coordinates and HMC for f is a product chain. Thus, we can consider the dynamics for each coordinate separately. The Hamiltonian ODE for one coordinate becomes

$$x_i'' = -\frac{x_i}{\sigma_i^2}, \quad x_i(0), \quad x_i'(0) = v_i(0) \sim N(0, 1)$$

where $i = 1, 2$. Solving the ODE above and plugging in the step-size $t = T$, we get

$$x_i(T) = x_i(0)\cos(T/\sigma_i) + v_i(0)\sigma_i \sin(T/\sigma_i).$$

Let P_i be the transition kernel of ideal HMC for the ith coordinate (considered as a Markov chain on $\mathbb{R}$). Then for $x, y \in \mathbb{R}$ we have

$$P_i(x, y) = \frac{1}{\sqrt{2\pi}\sigma_i \sin(T/\sigma_i)} \exp\left(-\frac{(y - x\cos(T/\sigma_i))^2}{2\sigma_i^2 \sin^2(T/\sigma_i)}\right).$$

Namely, given the current position x, the next position y is from a normal distribution with mean $x\cos(T/\sigma_i)$ and variance $\sigma_i^2 \sin^2(T/\sigma_i)$. Denote the spectral gap of P_i by γ_i for $i = 1, 2$ and that of ideal HMC by γ. Let $h(x) = x$ and note that $h \in L_2^0(\nu_i)$. Using the properties of product chains and spectral gaps, we deduce that

$$\gamma \le \min\{\gamma_1, \gamma_2\} \le \gamma_1 = 1 - \sup_{f \in L_2^0(\nu_1)} \frac{|\langle P_1 f, f\rangle|}{\|f\|^2} \le 1 - \frac{|\langle P_1 h, h\rangle|}{\|h\|^2}.$$

Since we have

$$\|h\|^2 = \int_{-\infty}^{\infty} \nu_1(x) h(x)^2 \mathrm{d}x = \sigma_1^2$$

and

$$\langle P_1 h, h\rangle = \int_{-\infty}^{\infty} \nu_1(x) P_1(x, y) h(x) h(y) \mathrm{d}x\mathrm{d}y = \sigma_1^2 \cos(T/\sigma_1),$$

it follows that $\gamma \le 1 - |\cos(T/\sigma_1)|$. Suppose that $T = 1/(c\sqrt{L})$ for some $c > 0$. Then we get

$$\gamma \le \frac{T^2}{2\sigma_1^2} = \frac{1}{2c^2}\frac{\mu}{L},$$

and consequently $\tau_{\mathrm{rel}} = 1/\gamma \ge 2c^2\kappa$. ◀

4 Convergence rate of discretized HMC

In this section, we show how our improved contraction bound (Lemma 6) implies that HMC returns a good enough sample after $\widetilde{O}((\kappa d)^{1.5})$ steps. We will use the framework from [9] to establish Theorem 5.

We first state the ODE solver from [9], which solves an ODE in nearly optimal time when the solution to the ODE can be approximated by a piece-wise polynomial. We state here only for the special case of second order ODEs for the Hamiltonian system. We refer to [9] for general kth order ODEs.

▶ **Theorem 9** ([9, Theorem 2.5]). *Let $x(t)$ be the solution to the ODE*

$$x''(t) = -\nabla f(x(t)), \quad x(0) = x_0, \quad x'(0) = v_0.$$

where $x_0, v_0 \in \mathbb{R}^d$ and $0 \le t \le T$. Suppose that the following conditions hold:

1. *There exists a piece-wise polynomial $q(t)$ such that $q(t)$ is a polynomial of degree D on each interval $[T_{j-1}, T_j]$ where $0 = T_0 < T_1 < \cdots < T_m = T$, and for all $0 \le t \le T$ we have*

$$\|q(t) - x''(t)\| \le \frac{\delta}{T^2};$$

2. *$\{T_j\}_{j=1}^m$ and D are given as input to the ODE solver;*
3. *The function f has a L-Lipschitz gradient; i.e., for all $x, y \in \mathbb{R}^d$,*

$$\|\nabla f(x) - \nabla f(y)\| \le L \|x - y\|.$$

If $\sqrt{L}T \le 1/16000$, then the ODE solver can find a piece-wise polynomial $\tilde{x}(t)$ such that for all $0 \le t \le T$,

$$\|\tilde{x}(t) - x(t)\| \le O(\delta).$$

The ODE solver uses $O(m(D+1)\log(CT/\delta))$ evaluations of ∇f and $O(dm(D+1)^2 \log(CT/\delta))$ time where

$$C = O\left(\|v_0\| + T\|\nabla f(x_0)\|\right).$$

The following lemma, which combines Theorem 3.2, Lemma 4.1 and Lemma 4.2 from [9], establishes the conditions of Theorem 9 in our setting. We remark that Lemmas 4.1 and 4.2 hold for all $T \le 1/(8\sqrt{L})$, and Theorem 3.2, though stated only for $T \le O(\mu^{1/4}/L^{3/4})$ in [9], holds in fact for the whole region $T \le 1/(2\sqrt{L})$ where the contraction bound (Lemma 6) is true. We omit these proofs here and refer the readers to [9] for more details.

▶ **Lemma 10.** *Let f be a twice differentiable function such that $\mu I \preceq \nabla^2 f(x) \preceq LI$ for all $x \in \mathbb{R}^d$. Choose the starting point $x^{(0)} = \arg\min_x f(x)$, step-size $T = 1/(16000\sqrt{L})$, and ODE error tolerance $\delta = \sqrt{\mu}T^2\varepsilon/16$ in the HMC algorithm. Let $\{x^{(k)}\}_{k=1}^N$ be the sequence of points we get from the HMC algorithm and $\{v_0^{(k)}\}_{k=1}^N$ be the sequence of random Gaussian vector we choose in each step. Let $\pi \propto e^{-f}$ be the target distribution and let π_{HMC} be the distribution of $x^{(N)}$, i.e., the output of HMC. For any $0 < \varepsilon < \sqrt{d}$, if we run HMC for*

$$N = O\left(\frac{\log(d/\varepsilon)}{\mu T^2}\right) = O\left(\kappa \log(d/\varepsilon)\right)$$

steps where $\kappa = L/\mu$, then:

1. ([9, Theorem 3.2]) *We have that*

$$W_2(\pi_{\text{HMC}}, \pi) \le \frac{\varepsilon}{\sqrt{\mu}};$$

2. ([9, Lemma 4.1]) *For each k, let $x_k(t)$ be the solution to the ODE (2) in the kth step of HMC. Then there is a piece-wise constant function q_k of m_k pieces such that $\|q_k(t) - x_k''(t)\| \le \delta/T^2$ for all $0 \le t \le T$, where*

$$m_k = \frac{2LT^3}{\delta}\left(\left\|v_0^{(k-1)}\right\| + T\left\|\nabla f(x^{(k-1)})\right\|\right);$$

3. ([9, Lemma 4.2]) *We have that*

$$\frac{1}{N}\,\mathbb{E}\left[\sum_{k=1}^{N}\left\|\nabla f(x^{(k-1)})\right\|^2\right] \le O(Ld).$$

Proof of Theorem 5. The convergence of HMC is guaranteed by part 1 of Lemma 10. In the kth step, the number of evaluations of ∇f is $O(m_k \log(C_k\sqrt{\kappa}/\varepsilon))$ by Theorem 9 and part 2 of Lemma 10, where

$$m_k = O\left(\frac{\sqrt{\kappa}}{\varepsilon}\right)\left(\left\|v_0^{(k-1)}\right\| + T\left\|\nabla f(x^{(k-1)})\right\|\right)$$

and

$$C_k = O\left(\left\|v_0^{(k-1)}\right\| + T\left\|\nabla f(x^{(k-1)})\right\|\right).$$

Thus, the average number of evaluations of ∇f per step is upper bounded by

$$\frac{1}{N}\,\mathbb{E}\left[\sum_{k=1}^{N} O(m_k \log(C_k\sqrt{\kappa}/\varepsilon))\right] \le \frac{1}{N}\,\mathbb{E}\left[\sum_{k=1}^{N} O(m_k \log m_k)\right] \le \frac{1}{N}\,O\left(\mathbb{E}\left[M \log M\right]\right),$$

where $M = \sum_{k=1}^{N} m_k$. Since each $v_0^{(k-1)}$ is sampled from the standard Gaussian distribution, we have $\mathbb{E}\left[\left\|v_0^{(k-1)}\right\|^2\right] = d$. Thus, by the Cauchy-Schwarz inequality and part 3 of Lemma 10, we get

$$\begin{aligned}\mathbb{E}\left[M^2\right] \le N\sum_{k=1}^{N}\mathbb{E}\left[m_k^2\right] &\le O\left(\frac{N\kappa}{\varepsilon^2}\right)\sum_{k=1}^{N}\mathbb{E}\left[\left\|v_0^{(k-1)}\right\|^2\right] + T^2\mathbb{E}\left[\left\|\nabla f(x^{(k-1)})\right\|^2\right] \\ &\le O\left(\frac{N^2\kappa d}{\varepsilon^2}\right).\end{aligned}$$

We then deduce again from the Cauchy-Schwarz inequality that

$$(\mathbb{E}[M\log M])^2 \le \mathbb{E}\left[M^2\right]\cdot\mathbb{E}\left[\log^2 M\right] \le \mathbb{E}\left[M^2\right]\cdot\log^2\left(\mathbb{E}M\right) \le \mathbb{E}[M^2]\cdot\log^2\left(\sqrt{\mathbb{E}\left[M^2\right]}\right),$$

where the second inequality is due to that $h(x) = \log^2(x)$ is concave when $x \ge 3$. Therefore, the number of evaluations of ∇f per step, amortized over all steps, is

$$\frac{1}{N}\,O\left(\sqrt{\mathbb{E}[M^2]}\log\left(\sqrt{\mathbb{E}\left[M^2\right]}\right)\right) \le O\left(\frac{\sqrt{\kappa d}}{\varepsilon}\log\left(\frac{\kappa d}{\varepsilon}\right)\right).$$

Using a similar argument we have the bound for the expected running time per step. This completes the proof. ◀

References

1 David Applegate and Ravi Kannan. Sampling and integration of near log-concave functions. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 156–163. ACM, 1991.

2 Niladri S. Chatterji, Nicolas Flammarion, Yi-An Ma, Peter L. Bartlett, and Michael I. Jordan. On the theory of variance reduction for stochastic gradient Monte Carlo. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

3 Xiang Cheng, Niladri S. Chatterji, Yasin Abbasi-Yadkori, Peter L. Bartlett, and Michael I. Jordan. Sharp convergence rates for Langevin dynamics in the nonconvex setting. *ArXiv preprint*, 2018. arXiv:1805.01648.

4 Xiang Cheng, Niladri S. Chatterji, Peter L. Bartlett, and Michael I. Jordan. Underdamped Langevin MCMC: A non-asymptotic analysis. In *Proceedings of the 2018 Conference on Learning Theory (COLT)*, 2018.

5 Arnak S. Dalalyan. Theoretical guarantees for approximate sampling from smooth and log-concave densities. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(3):651–676, 2017.

6 Arnak S. Dalalyan and Avetik Karagulyan. User-friendly guarantees for the Langevin Monte Carlo with inaccurate gradient. *Stochastic Processes and their Applications*, 2019.

7 Arnak S. Dalalyan and Lionel Riou-Durand. On sampling from a log-concave density using kinetic Langevin diffusions. *ArXiv preprint*, 2018. arXiv:1807.09382.

8 Raaz Dwivedi, Yuansi Chen, Martin J. Wainwright, and Bin Yu. Log-concave sampling: Metropolis-Hastings algorithms are fast! In *Proceedings of the 2018 Conference on Learning Theory (COLT)*, 2018.

9 Yin Tat Lee, Zhao Song, and Santosh S. Vempala. Algorithmic Theory of ODEs and Sampling from Well-conditioned Logconcave Densities. *ArXiv preprint*, 2018. arXiv:1812.06243.

10 Yin Tat Lee and Santosh S. Vempala. Convergence rate of Riemannian Hamiltonian Monte Carlo and faster polytope volume computation. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1115–1121. ACM, 2018.

11 László Lovász and Santosh S. Vempala. Fast Algorithms for Logconcave Functions: Sampling, Rounding, Integration and Optimization. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 57–68. IEEE, 2006.

12 Yi-An Ma, Niladri Chatterji, Xiang Cheng, Nicolas Flammarion, Peter Bartlett, and Michael I. Jordan. Is There an Analog of Nesterov Acceleration for MCMC? *ArXiv preprint*, 2019. arXiv:1902.00996.

13 Oren Mangoubi and Aaron Smith. Mixing of Hamiltonian Monte Carlo on strongly log-concave distributions 1: continuous dynamics. *ArXiv preprint*, 2017. arXiv:1708.07114.

14 Oren Mangoubi and Aaron Smith. Mixing of Hamiltonian Monte Carlo on strongly log-concave distributions 2: Numerical integrators. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 586–595, 2019.

15 Oren Mangoubi and Nisheeth Vishnoi. Dimensionally tight bounds for second-order Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems (NIPS)*, pages 6027–6037, 2018.

16 Yann Ollivier. Ricci curvature of Markov chains on metric spaces. *Journal of Functional Analysis*, 256(3):810–864, 2009.

17 Maxim Raginsky, Alexander Rakhlin, and Matus Telgarsky. Non-convex learning via Stochastic Gradient Langevin Dynamics: a nonasymptotic analysis. In *Proceedings of the 2017 Conference on Learning Theory (COLT)*, 2017.

18 Christof Seiler, Simon Rubinstein-Salzedo, and Susan Holmes. Positive curvature and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems (NIPS)*, pages 586–594, 2014.

19 Santosh S. Vempala and Andre Wibisono. Rapid Convergence of the Unadjusted Langevin Algorithm: Isoperimetry Suffices. *ArXiv preprint*, 2019. arXiv:1903.08568.

20 Andre Wibisono. Sampling as optimization in the space of measures: The Langevin dynamics as a composite optimization problem. In *Proceedings of the 2018 Conference on Learning Theory (COLT)*, 2018.

21 Yuchen Zhang, Percy S. Liang, and Moses Charikar. A Hitting Time Analysis of Stochastic Gradient Langevin Dynamics. In *Proceedings of the 2017 Conference on Learning Theory (COLT)*, 2017.

Exploring Differential Obliviousness

Amos Beimel[1]
Dept. of Computer Science, Ben-Gurion University, Israel
amos.beimel@gmail.com

Kobbi Nissim
Dept. of Computer Science, Georgetown University, Washington, D.C., USA
kobbi.nissim@georgetown.edu

Mohammad Zaheri
Dept. of Computer Science, Georgetown University, Washington, D.C., USA
mz394@georgetown.edu

Abstract

In a recent paper, Chan et al. [SODA '19] proposed a relaxation of the notion of (full) memory obliviousness, which was introduced by Goldreich and Ostrovsky [J. ACM '96] and extensively researched by cryptographers. The new notion, *differential obliviousness*, requires that any two neighboring inputs exhibit similar memory access patterns, where the similarity requirement is that of differential privacy. Chan et al. demonstrated that differential obliviousness allows achieving improved efficiency for several algorithmic tasks, including sorting, merging of sorted lists, and range query data structures.

In this work, we continue the exploration of differential obliviousness, focusing on algorithms that do not necessarily examine all their input. This choice is motivated by the fact that the existence of logarithmic overhead ORAM protocols implies that differential obliviousness can yield at most a logarithmic improvement in efficiency for computations that need to examine all their input. In particular, we explore property testing, where we show that differential obliviousness yields an almost linear improvement in overhead in the dense graph model, and at most quadratic improvement in the bounded degree model. We also explore tasks where a non-oblivious algorithm would need to explore different portions of the input, where the latter would depend on the input itself, and where we show that such a behavior can be maintained under differential obliviousness, but not under full obliviousness. Our examples suggest that there would be benefits in further exploring which class of computational tasks are amenable to differential obliviousness.

2012 ACM Subject Classification Security and privacy → Privacy-preserving protocols

Keywords and phrases Differential Obliviousness, Differential Privacy, Oblivious RAM, Graph Property Testing

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.65

Category RANDOM

Related Version A full version of the paper is available at `https://arxiv.org/abs/1905.01373`.

Funding Work supported by NSF grant No. 1565387 TWC: Large: Collaborative: Computing Over Distributed Sensitive Data. A.B. is additionally supported by ISF grant no. 152/17, a grant from the Cyber Security Research Center at Ben-Gurion University, and ERC grant 742754 (project NTSC).

1 Introduction

A program's memory access pattern can leak significant information about the private information used by the program even if the memory content is encrypted. Such leakage can turn into a data protection problem in various settings. In particular, where data

[1] Work done while A.B. was visiting Georgetown University.


© Amos Beimel, Kobbi Nissim, and Mohammad Zaheri;
licensed under Creative Commons License CC-BY



is outsourced to be stored on an external server, it has been shown that access pattern leakage can be exploited in practical attacks and lead to the compromise of the underlying data [20, 4, 29, 21, 23]. Such leakages can also be exploited when a program is executed in a secure enclave environment but needs to access memory that is external to the enclave.

Memory access pattern leakage can be avoided by employing a strategy that makes the sequence of memory accesses (computationally or statistically) independent of the content being processed. Beginning with the seminal work of Goldreich and Ostrovsky, it is well known how to transform any program running on a random access memory (RAM) machine to one with an *oblivious* memory access pattern while retaining efficiency by using an Oblivious RAM protocol (ORAM) [10, 30, 13]. Current state-of-the-art ORAM protocols achieve logarithmic overhead [2], matching a recent lowerbound by Larsen and Nielsen [24], and protocols with $O(1)$ overhead exist when the server is allowed to perform computation and large blocks are retrieved [6, 28]. To further reduce the overhead, oblivious memory access pattern protocols have been devised for specific tasks, including graph algorithms [3, 17], geometric algorithms [8] and sorting [16, 25]. The latter is motivated by sorting being a fundamental and well researched computational task as well as its ubiquity in data processing.

1.1 Differential Obliviousness

Full obliviousness is rather a strong requirement: any two possible inputs (of the same size) should exhibit identical or indistinguishable sequences of memory accesses. Achieving full obliviousness via a generic use of ORAM protocols requires a setup phase with running time (at least) linear in the memory size and then a logarithmic overhead per each memory access.

A recent work by Chan, Chung, Maggs, and Shi [5] put forward a relaxation of the obliviousness requirement where indistinguishability is replaced with differential privacy. Intuitively, this means that any two possible neighboring inputs should exhibit memory access patters that are similar enough to satisfy differential privacy, but may still be too dissimilar to be "cryptographically" indistinguishable. It is not a priori clear whether differential obliviousness can be achieved without resorting to full obliviousness. However, the recent work Chan et al. showed that differential obliviousness does allow achieving improved efficiency for several algorithmic tasks, including sorting (over very small domains), merging of sorted lists, and range query data structures.

Also of relevance are the works by He et al. [19] and Mazloom and Gordon [27], which study protocols for secure multiparty computation in which the parties are allowed to learn information from the computation as long as this information preserves the differential privacy of the input. He et al. and Mazloom and Gordon demonstrate that this leakage is useful: He et al. construct protocols for the private record linkage problem for two databases; Mazloom and Gordon present protocols for histograms, PageRank, and matrix factorization.

Furthermore, even the use of ORAM protocols may be insufficient for preventing leakage in cases where the number of memory probes is input dependent. In fact, Kellaris et al. [21] show that such leakage can result in a complete reconstruction in the case of retrieving elements specified by range queries, as the number of records returned depends on the contents of the data structure. Full obliviousness would require that the sequence of memory accesses would be padded to a maximal one to avoid such leakage, a solution that would have a dire effect on the efficiency of many algorithms. Differential obliviousness may in some cases allow achieving meaningful privacy while maintaining efficiency. Examples of such protocols include the combination of ORAM with differentially private sanitization by Kellaris et al. [22] and the recent work of Chan et al. [5] on range query data structures, which avoids using ORAM.

1.2 This Work: Exploring Differential Obliviousness

Noting that the existence of logarithmic overhead ORAM protocols implies that differential obliviousness can yield at most a logarithmic improvement in efficiency for computations that need to examine all their input, we explore tasks where this is not the case. In particular, we focus on property testing and on tasks where the number of memory accesses can depend on the input.

Property testing. As evidence that differential obliviousness can provide a significant improvement over full obliviousness, we show in Section 3 that property testers in the dense graph model, where the input is in the adjacency matrix representation [12], can be made differentially oblivious. This result captures a large set of testable graph properties [12, 1] including, e.g., graph bipartitness and having a large clique. Testers in this class probe a uniformly random subgraph and hence are fully oblivious without any modification, as their access pattern does not depend on the input graph. However, this is not the case if the tester reveals its output to the adversary, as this allows learning information about the specific probed subgraph. A fully oblivious tester would need to access a linear-sized subgraph, whereas we show that a differentially oblivious tester only needs to apply the original tester $O(1)$ times.[2]

We also consider property testing in the bounded degree model, where the input is in the incidence lists model [14]. In this model we provide negative results, demonstrating that adaptive testers cannot, generally, be made differentially oblivious without a significant loss in efficiency. In particular, in Section 4 we consider differentially oblivious property testers for connectivity in graphs of degree at most two. For non-oblivious testers, it is known that constant number of probes suffice when the tester is adaptive [14].[3] It is also known that any non-adaptive tester for this task requires probing $\Omega(\sqrt{n})$ nodes [32]. We show that this lowerbound extends to differentially oblivious testers, i.e., any differentially oblivious tester for connectivity in graphs of maximal degree 2 requires $\Omega(\sqrt{n})$ probes. While this still improves over full obliviousness, the gap between full and differential obliviousness is in this case diminished.

Locating an Object Satisfying a Property. Here, our goal is to check whether a given data set of objects includes an object that satisfies a specified property. Without obliviousness requirements, a natural approach is to probe elements in a random order until an element satisfying the property is found or all elements were probed. If a p fraction of the elements satisfy the property, then the expected number of probes is $1/p$. This algorithm is in fact instance optimal when the data set is randomly permuted.[4]

A fully oblivious algorithm would require $\Omega(n)$ probes on any dataset even when $p = 1$. In contrast, we demonstrate in Section 5 that with differential obliviousness instance optimality can, to a large extent, be preserved. Our differentially oblivious algorithm always returns a correct answer and makes at most m probes with probability at least $1 - e^{-O(mp)}$.

Prefix Sum. Our last example considers a sorted dataset (possibly, the result of an earlier phase in the computation). Our goal is to compute the sum of all records in the (sorted) dataset that are less than or equal to a given value a (see Section 6 for the definition of privacy).

[2] We omit dependencies on privacy and accuracy parameters from this introductory description.
[3] In an adaptive tester at least one choice of a node to probe should depend on information gathered from incidence lists of previously probed nodes.
[4] Our treatment of instance optimality is rather informal. The concept was originally presented in [9].

Without obliviousness requirements, one can find the greatest record less than or equal to value a, say, using binary search, and then compute the prefix sum by a quick scan through all records appearing before this record. This algorithm is in fact nearly instance optimal, as it can be shown that any algorithm which returns the correct exact answer with non-negligible probability must probe all entries greater than a. However, fully oblivious algorithms would have to probe the entire dataset.

In Section 6, we give our nearly instance optimal differentially oblivious prefix sum algorithm. As the probes of a binary search would leak information about the memory content, we introduce a differentially oblivious "simulation" of the binary search. Our differentially oblivious binary search runs in time $O(\log^2 n)$.

We also address the scenario where there are multiple prefix sum queries to the same database. If the number of queries is bounded by some integer t, then each differentially oblivious binary search will run in time $O(t \log^2 n)$ (as we need to run the search algorithm with a smaller privacy parameter ε). Using ORAM, one can answer such queries with $O(n \log n)$ prepossessing time and $O(\log^2 n)$ time per query. Combining our algorithm and ORAM, we can amortize the pre-processing time over $O(\sqrt{n})$ queries, that is, without any pre-processing, the running of time of answering the i-th query is $O(i \log^4 n)$ for the first $O(\sqrt{n})$ queries and $O(\log^2 n)$ for any further query.

1.3 Background Work

The papers by Chan, Chung, Maggs, and Shi [5], He, Machanavajjhala, Flynn, and Srivastava [19], and by Mazloom and Gordon [27] mentioned above are most relevant for this article. As mentioned above, Kellaris et al. [22] examined a similar concept with the goal of preventing reconstruction attacks in secure remote databases. Goldreich, Goldwasser, and Ron [12] initiated the research on graph property testing. Persiano and Yeo [31] showed that the $O(\log n)$ lowerbound for ORAM of [24] also holds when the security requirement is relaxed to differetial privacy. Goldreich's book on property testing [11] gives sufficient background for our discussion. Dwork, McSherry, Nissim, and Smith [7] defined differential privacy. For more details on ORAM and a list of relevant papers, the reader can consult [2].

2 Definitions

2.1 Model of Computation

We consider the standard Random Access Memory (RAM) model of computation that consists of a CPU and a memory. The CPU executes a program and is allowed to perform two types of memory operations: read a value from a specified physical address, and write a value to a specified physical address. We assume that the CPU has a private cache of where it can store $O(1)$ values (and/or a polylogarithmic number of bits). As an example, in the setting of a client storing its data on the cloud, the client plays the role of the CPU and the cloud server plays the role of the memory.

We assume that a program's sequence of read and write operations may be visible to an adversary. We will call this sequence the program's access pattern. We will further assume that the memory content is encrypted so that no other information is leaked about the content read from and stored in memory location. The program's access pattern may depend on the program's input, and may hence leak information about it.

Algorithm 1 Experiment $\mathsf{Exp}_b^{A,M}$ for defining differential obliviousness.

$(\mathbf{x}_0, \mathbf{x}_1, st) \leftarrow\!\!\$\ A_1(\lambda, n)$
$b' \leftarrow\!\!\$\ A_2^{\mathcal{M}(\mathbf{x}_b,\cdot)}(st)$
Return b'

Oracle $\mathcal{M}(\mathbf{x}, q)$
$\mathsf{out} \leftarrow\!\!\$\ M(\mathbf{x}, q, state)$
Return $\mathsf{Access}^M(\mathbf{x}, q, state)$

2.2 Oblivious Algorithms

There are various works focused on oblivious algorithms [8, 15, 26] and Oblivious RAM (ORAM) constructions [13]. These works adopt "full obliviousness" as a privacy notion. Suppose that $M(\lambda, \mathbf{x})$ is an algorithm that takes in two inputs, a security parameter λ and an input dataset denoted $\mathbf{x}$. We denote by $\mathsf{Access}^M(\lambda, \mathbf{x})$, the ordered sequence of memory accesses the algorithm M makes on the input λ and $\mathbf{x}$.

▶ **Definition 1** (Fully Oblivious Algorithms). *Let δ be a function in a security parameter λ. We say that algorithm M is δ-statistically oblivious, iff for all inputs $\mathbf{x}$ and $\mathbf{y}$ of equal length, and for all λ, it holds that $\mathsf{Access}^M(\lambda, \mathbf{x}) \approx^{\delta(\lambda)} \mathsf{Access}^M(\lambda, \mathbf{y})$ where $\approx^{\delta(\lambda)}$ denotes that the two distributions have at most $\delta(\lambda)$ statistical distance. We say that M is perfectly oblivious when $\delta = 0$.*

2.3 Differentially Oblivious Algorithms

Suppose that $M(\lambda, \mathbf{x}, q)$ is a (stateful) algorithm that takes in three inputs, a security parameter $\lambda > 0$, an input dataset denoted by $\mathbf{x}$ and a value q. We slightly change the definition of differentially oblivious algorithms given in [5]:

▶ **Definition 2** (Neighbor-respecting). *We say that two input datasets $\mathbf{x}$ and $\mathbf{y}$ are neighboring iff they are of the same length and differ in exactly one entry. We say that $A = (A_1, A_2)$ is neighbor-respecting adversary iff for every λ and every n, A_1 outputs neighboring datasets $\mathbf{x}_0, \mathbf{x}_1$, with probability 1.*

▶ **Definition 3.** *Let ε, δ be privacy parameters. Let M be a (possibly stateful) algorithm described as above. To an adversary A we associate the experiment in Algorithm 1, for every $\lambda \in \mathbb{N}$. We say that M is (ε, δ)-adaptively differentially oblivious if for all (computationally unbounded) stateful neighbor-respecting adversary A we have*

$$\Pr[\mathsf{Exp}_0^{A,M}(\lambda, n) = 1] \leq e^{\varepsilon} \cdot \Pr[\mathsf{Exp}_1^{A,M}(\lambda, n) = 1] + \delta.$$

In Algorithm 1, $\mathsf{Access}^M(\mathbf{x}, q, state)$ denotes the ordered sequence of memory accesses the algorithm M makes on the inputs $\mathbf{x}, q$ and state.

▶ Remark 4. The notion of adaptivity here is different from the one defined in [5]. We require that the dataset $\mathbf{x}$ remain the same through the experiment whereas in [5] the adaptive adversary can add or remove entries from the dataset.

As with differential privacy, we usually think about ε as a small constant and require that $\delta = o(1/n)$ where $n = |\mathbf{x}|$ [7]. Observe that if M is δ-statistically oblivious then it is also $(0, \delta)$-differentially oblivious.

The following simple lemma will be useful to analyze our algorithms. The proof of the lemma appears in Appendix A.

▶ **Lemma 5.** *Let $\mathcal{A}$ be an $(\varepsilon, 0)$-differentially oblivious algorithm and $\mathcal{B}$ be an algorithm such that for every dataset $\mathbf{x}$ the statistical distance between $\mathcal{A}(\mathbf{x})$ and $\mathcal{B}(\mathbf{x})$ is at most γ (that is, $|\Pr[\mathcal{A}(\mathbf{x}) \in S] - \Pr[\mathcal{B}(\mathbf{x}) \in S]| \leq \gamma$ for every S). Then, $\mathcal{B}$ is an $(\varepsilon, (1 + e^{\varepsilon})\gamma)$-differentially oblivious algorithm.*

3 Differentially Oblivious Property Testing of Dense Graphs Properties

In this section, we present a differentially oblivious property tester for dense graphs properties in the adjacency matrix representation model. A property tester is an algorithm that decides whether a given object has a predetermined property or is far from any object having this property by examining a small random sample of its input. The correctness requirement of property testers ignores objects that neither have the property nor are far from having the property. However, the privacy requirement is "worst case" and should hold for any two neighboring graphs. For the definition of privacy we say that two graphs G, G' of size n are neighbors if one can get G' by changing the neighbors of exactly one node of G.

Property testing of graph properties in the adjacency matrix representation was introduced in [12]. A graph $G = (V, E)$ is represented by the predicate $f_G : V \times V \to \{0, 1\}$ such that $f_G(u, v) = 1$ if and only if u and v are adjacent in G. The distance between graphs is defined to be the number of different matrix entries over $|V|^2$. This model is most suitable for dense graphs where the number of edges is $O(|V|^2)$. We define a property $\mathcal{P}$ of graphs to be a subset of the graphs. We write $G \in \mathcal{P}$ to show that graph G has the property $\mathcal{P}$. For example, we can define the bipartiteness property, where $\mathcal{P}$ is the set of all bipartite graphs.[5] We say that an n-vertex G is γ-far from $\mathcal{P}$ if for every n-vertex graph $G' = (V', E') \in \mathcal{P}$ it holds that the symmetric difference between E and E' is greater than γn^2. We define the property testing in this model as follows:

▶ **Definition 6** ([12])**.** *A (β, γ)-tester for a graph property $\mathcal{P}$ is a probabilistic algorithm that, on inputs n, β, γ, and an adjacency matrix of an n-vertex graph $G = (V, E)$:*

1. *Outputs 1 with probability at least β, if $G \in \mathcal{P}$.*
2. *Outputs 0 with probability at least β, if G is γ-far from $\mathcal{P}$.*

We say a tester has one-sided error, if it accepts every graph in $\mathcal{P}$ with probability 1. We say a tester is non-adaptive if it determines all its queries to adjacency matrix only based on n, β, γ, and its randomness; otherwise, we say it is adaptive.

▶ **Example 7** ([12])**.** Consider the following $(2/3, \gamma)$-tester for bipartiteness: Choose a random subset $A \subset V$ of size $\tilde{O}(1/\gamma^2)$ with uniform distribution and output 1 iff the graph induced by A is bipartite. Clearly, if G is bipartite, then the tester will always return 1. Goldreich et al. [12] proved that if G is γ-far from a bipartite graph, then the probability that the algorithm returns 1 is at most 1/3.

Recall that in the graph property testing, the tester $\mathcal{T}$ chooses a random subset of the graph with uniform distribution to test the property $\mathcal{P}$. Given the access pattern of the tester $\mathcal{T}$, an adversary will learn nothing since it is uniformly random. Thus, the access pattern by itself does not reveal any information about the input graph. However, we assume that the adversary also learns the tester's output and can hence learn some information

[5] Recall that an undirected graph is bipartite (or 2-colorable) if its vertices can be partitioned into two parts, V_1 and V_2, such that each part is an independent set (i.e., $E \subseteq \{(u, v) : (u, v) \in V_1 \times V_2\}$).

Algorithm 2 Differentially Oblivious Property Tester $\mathsf{Tester}_{\mathcal{T}}$ for Dense Graphs.

Input: graph $G = (V, E)$
Let $c \leftarrow 0$ and $T \leftarrow \frac{\ln(1/2\delta)}{\varepsilon}$
for $i = 1$ to $4T$ **do**
  if $\mathcal{T}(G) = 1$ **then**
    $c \leftarrow c + 1$
  end if
  Let A be the subset of vertices chosen by tester $\mathcal{T}$
  Update graph G to be the induced sub-graph on $V \backslash A$
end for
$\hat{T} \leftarrow 3T + \mathrm{Lap}(\frac{1}{\varepsilon})$
if $c \geq \min(\hat{T}, 4T)$ **then**
  output 1
else
  output 0
end if

about the input graph based on the output of the tester. To protect this information, we run tester $\mathcal{T}$ for constant number of times and output 1 iff the number of times $\mathcal{T}$ outputs 1 exceed a (randomly chosen) threshold.

Let $\mathcal{T}$ be a (β, γ)-tester for a graph property $\mathcal{P}$ where $\beta \leq 1/4$. We write $c_{\beta,\gamma}$ for the number of nodes that $\mathcal{T}$ samples. Note that $c_{\beta,\gamma}$ is constant in the graph size and a function of β and γ. For simplicity, we only consider property testers with one-sided error. In Algorithm 2, we describe a (β', γ')-tester $\mathsf{Tester}_{\mathcal{T}}$ that outputs 1 with probability 1 if $G \in \mathcal{P}$ and outputs 0 with probability at least β', if G is γ'-far from $\mathcal{P}$, where β' and γ' are defined below.

▶ **Theorem 8.** *Let $\varepsilon, \delta > 0$ and $\gamma' = \gamma - \frac{4\ln(1/2\delta)c_{\beta,\gamma}}{n\varepsilon}$. Algorithm $\mathsf{Tester}_{\mathcal{T}}$ is an $(\varepsilon, \delta(1+e^{\varepsilon}))$-differentially oblivious algorithm that outputs 1 with probability 1 if $G \in \mathcal{P}$, and output 0 with probability at least $1 - \delta - (2\delta)^{\frac{1}{3\varepsilon}}$ if G is γ'-far from $\mathcal{P}$.*

The proof of Theorem 8 appears in Appendix A.2.

4 Lower Bounds on Testing Connectivity in the Incidence Lists Model

We now consider differentially oblivious testing of connectivity in the incidence lists model [14]. In this model a graph has a bounded degree d and is represented as a function $f : V \times [d] \to V \cup \{0\}$, where $f(v, i)$ is the i-th neighbor of v (if no such neighbor exists, then $f(v, i) = 0$). In this model, the relative distance between graphs is normalized by dn – the maximal number of edges in the graph. Formally, for two graphs with n vertices,

$$\mathrm{dist}_d(G_1, G_2) \triangleq \frac{|\{(v, i) : v \in V, i \in [d], f_{G_1}(v, i) \neq f_{G_2}(v, i)\}|}{dn}.$$

A (β, γ)-tester in the incidence lists model is defined as in Definition 6, where a property $\mathcal{P}$ is a set of graphs whose maximal degree is d and the distance to a property is defined with respect to dist_d.

Goldreich and Ron [14] showed how to test if a graph is connected in the incidence list model in time $\tilde{O}(1/\gamma)$. Raskhodnikova and Smith [32] showed that a tester for connectivity (or any non-trivial property) with run-time $o(\sqrt{n})$ has to be adaptive, that is, the nodes that

the algorithm probes should depend on the neighbors of nodes the algorithm has already probed (e.g., the algorithm probes some node u, discovers that v is a neighbor and u, and probes v). We strengthen their results by showing that any tester for connectivity in graphs of maximal degree 2 and run-time $o(\sqrt{n})$ cannot be a differentially oblivious algorithm. We stress that adaptivity alone is not a reason for inefficiency with differential obliviousness. In fact, there exist differentially oblivious algorithms that are adaptive (e.g., our algorithm in Section 6).

▶ **Theorem 9.** *Let $\varepsilon, \delta > 0$ such that $e^{4\varepsilon}\delta < 1/16n$. Every (ε, δ)-differentially private $(3/4, 1/3)$-tester for connectivity in graphs with maximal degree 2 runs in time $\Omega(\sqrt{n}/e^{2\varepsilon})$.*

Proof. Let Tester be a $(3/4, 1/3)$-tester for connectivity in graphs of degree at most 2. We somewhat relax the definition of probes and assume that once the tester probes a node, it sees all edges adjacent to this node. We prove that if Tester probes less than $c\sqrt{n}/e^{2\varepsilon}$ nodes (for some constant c), then it is not (ε, δ)-oblivious. Assume that $n \equiv 0 \pmod 3$. Let $G_1 = (V, E_1)$ be a cycle of length n and $G_2 = (V, E_1)$ consist of $n/3$ disjoint triangles. Clearly, G_1 is connected and G_2 is 1/3-far from a connected graph. For a permutation $\pi : V \to V$, define $\pi(G_i) = (V, \pi(E_i))$, where $\pi(E_i) = \{(\pi(u), \pi(v)) : (u, v) \in E_i\}$, and let $\mathrm{perm}(G_i)$ be a random graph isomorphic to G_i, that is, $\mathrm{perm}(G_i) = \pi(G_i)$ for a permutation π chosen with uniform distribution.[6] On the random graph $\mathrm{perm}(G_)$ Tester has to say "yes" with probability at least 3/4 and on the random graph $\mathrm{perm}(G_2)$ Tester has to say "no" with probability at least 3/4.

▶ **Observation 10.** *If* Tester *does not probe two distinct nodes whose distance is at most two, then* Tester *sees a collection of paths of length two and cannot know if the graph is* $\mathrm{perm}(G_1)$ *or* $\mathrm{perm}(G_2)$.

▷ Claim 11. Given the random graph $\mathrm{perm}(G_1)$, the tester has to probe two distinct nodes whose distance is at most 2 with probability at least 1/2.

Proof. Consider Tester's answer when it sees a collection of paths of length 2. Assume first that the tester returns "No" with probability at least half in this case and let p be the probability that Tester probes two distinct nodes whose distance is at most two on the random graph $\mathrm{perm}(G_1)$. The probability that Tester returns "Yes" on $\mathrm{perm}(G_1)$ is at most $p + 0.5(1-p) = 0.5 + 0.5p$. Thus, $0.5 + 0.5p \geq 3/4$, i.e., $p \geq 0.5$.

If the tester returns "Yes" with probability at least half, then, by symmetric arguments, with probability at least 1/2 Tester has to probe two nodes whose distance is at most two on the random graph $\mathrm{perm}(G_2)$. For a permutation π, if the distance between two nodes in $\pi(G_2)$ is at most 2, then the distance between these two nodes in $\pi(G_1)$ is at most 2. Thus, by Observation 10,

$$\Pr[\text{Tester probes 2 nodes whose distance is 1 or 2 on } \mathrm{perm}(G_1)] \geq \Pr[\text{Tester probes 2 nodes whose distance is 1 or 2 } \mathrm{perm}(G_2)] \geq 1/2. \quad \triangleleft$$

[6] When we permute a graph, we also permute its incident list representation, i.e., if $(u, v) \in \pi(E)$, then with probability half v will be the first neighbor of u and with probability half it will be the second.

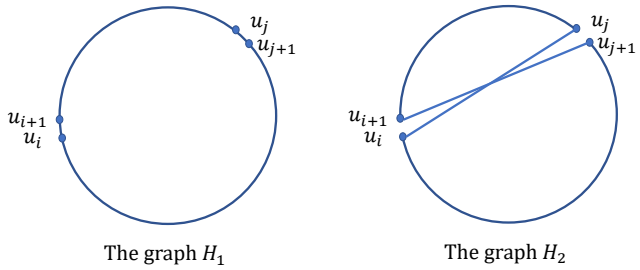


Figure 1 The graphs H_1 and H_2.

Denote the nodes of G_1 by $V = \{v_0, \ldots, v_{n-1}\}$ and define a distribution on pairs of graphs H_1, H_2, obtained by the following process:

- Choose a permutation $\pi : V \to V$ with uniform distribution and let $H_1 = \pi(G_1)$.
- Denote $H_1 = (V, E_1)$ and $u_j = \pi(v_j)$ for $j \in [n]$.
- Choose with uniform distribution two indices i, j such that $j \in \{i+4, i+3, \ldots, i-3\}$ (where the addition is done modulo n).
- Let $H_2 = (V, E_2)$, where $E_2 = E_1 \setminus \{(u_i, u_{i+1}), (u_j, u_{j+1})\} \cup \{(u_i, u_j), (u_{i+1}, u_{j+1})\}$.

The graphs are described in Figure 1. Note that H_2 is also a a random graph isomorphic to G_1, thus, given H_2 one cannot know which pair of non-adjacent nodes u_i, u_j was used to create H_2.

Observe that H_1 and H_2 differ on 4 nodes. Since Tester is (ε, δ)-differentially oblivious, for every algorithm $\mathcal{A}$,

$$\Pr[\mathcal{A}(H_1, H_2, \mathsf{Access}^{\textsc{Tester}}(H_1)) = 1] \leq e^{4\varepsilon} \cdot \Pr[\mathcal{A}(H_1, H_2, \mathsf{Access}^{\textsc{Tester}}(H_2)) = 1] + 4e^{4\varepsilon}\delta. \quad (1)$$

Consider the following algorithm $\mathcal{A}$:

If u_i and at least one of u_{i+1}, u_{i+2} is probed by $\textsc{Tester}(H)$ prior to seeing any other pair of nodes of distance at most 2 in H_1 or H_2, then return 1 otherwise return 0.

▷ Claim 12. Let $i \in \{1, 2\}$. Suppose that Tester probes at most q nodes. Pick at random with uniform distribution two nodes in V with distance at least 3 in H_i. The probability that $\textsc{Tester}(H_i)$ probes both u and v prior to seeing any two nodes of distance at most 2 in H_i is $O(q^2/n^2)$ (where the probability is over the random choice of u, v and the randomness of Tester).

Proof. The node u is a uniformly distributed node in H_i and v is any node of distance at least 3 from v, thus there are $n(n-5)/2$ options for $\{u, v\}$. Given a collection of paths of length at most 2 in H_i all options are equally likely.

Let $w_1, \ldots, w_k$ be the nodes probed in some execution of Tester. Fix some pair of indices $k_1 < k_2$. The probability that $\{u_i, u_{i+1}\} = \{w_{k_1}, w_{k+2}\}$ is at most $1/n(n-5)$. Thus, the probability that u and v are probed is at most $\frac{\binom{q}{2}}{n(n-5)/2} = O(q^2/n^2)$. ◁

▷ Claim 13. Assume that TESTER probes at most q nodes. The probability that $\mathcal{A}(H_1) = 1$ is at least $1/2n - O(q^2/n^2)$.

Proof. By Claim 11, the probability that TESTER probes at least one pair of nodes with distance at most 2 is at least $1/2$. Given that this event occurs, the probability that the random u_i (chosen with uniform distribution) has the smallest index in the first such pair in H_1 (i.e., the first pair is either (u_i, u_{i+1}) or (u_i, u_{i+2})) is at least $1/n$.

Clearly, given these events no two nodes with distance at most 2 in H_1 were probed prior to probing the pair containing u_i. Furthermore, there are $O(1)$ pairs of nodes that are of distance at most 2 in H_2 and are of distance greater than 2 in H_1. By Claim 12, the probability that such pair is probed prior to TESTER probing a pair of distance at most 2 in H_1 is $O(q^2/n^2)$. ◁

▷ Claim 14. Suppose that TESTER probes at most q nodes. The probability that $\mathcal{A}(H_2) = 1$ is $O(q^2/n^2)$.

Proof. The node u_i is a uniformly distributed node in H_2. Furthermore, the nodes u_{i+1} is a uniformly distributed node of distance at least 3 from u_i in H_2, thus by Claim 12, the probability that TESTER probes both u_i and u_{i+1} prior to seeing any pair of distance at least 2 in H_2 is $O(q^2/n^2)$. This probability can only decrease if we require that TESTER probes both u_i and u_{i+1} prior to seeing any pair of distance at least 2 in H_1 and in H_2.

By the same arguments, the probability that TESTER probes both u_i and u_{i+2} prior to seeing any pair of distance at least 2 in H_1 and in H_2 is $O(q^2/n^2)$. ◁

To conclude the proof of Theorem 9, we note that by (1) and Claim 13 and 14

$$\frac{1}{2n} - O(q^2/n) \leq \Pr[\mathcal{A}(H_1) = 1] \leq e^{4\varepsilon} \Pr[\mathcal{A}(H_2) = 1] + e^{4\varepsilon}\delta \leq e^{4\varepsilon} O(q^2/n^2) + e^{4\varepsilon}\delta.$$

Since $e^{4\varepsilon}\delta \leq 1/4n$, it follows that $q = \Omega(\sqrt{n}/e^{2\varepsilon})$. ◀

5 Differentially Oblivious Algorithm for Locating an Object

Given a dataset of objects $\mathbf{x}$ our goal is to locate an object that satisfies a property $\mathcal{P}$, if one exists. E.g., given a dataset consisting of employee records, find an employee with income in the range $\$35,000–$\$70,000$ if such an employee exists in the dataset.

Absent privacy requirements, a simple approach is to probe elements of the dataset in a random order until an element satisfying the property is found or all elements were probed. If a p fraction of the dataset entries satisfy $\mathcal{P}$ then the expected number of elements sampled by the non-private algorithm is $1/p$. However, a perfectly oblivious algorithm would require $\Omega(n)$ probes on any dataset, in particular on a dataset where all elements satisfy $\mathcal{P}$, where non-privately one probe would suffice. To see why, let $\mathcal{P}(x) = 1$ if $x = 1$ and $\mathcal{P}(x) = 0$ otherwise and let $\mathbf{x}$ include exactly one 1-entry in a uniformly random location. Observe that in expectation it requires $\Omega(n)$ memory probes to locate the 1-entry in $\mathbf{x}$. Perfect obliviousness would hence imply an $\Omega(n)$ probes on any input.

We give a nearly instance optimal differentially oblivious algorithm that always returns a correct answer. Except for probability $e^{-\Omega(mp)}$ the algorithm halts after m steps.

Our Algorithm. Given the access pattern of the non-private algorithm, an adversary can learn that the last probed element satisfies $\mathcal{P}$. To hide this information, we change the stopping condition to having probed at least a (randomly chosen) threshold of elements

Algorithm 3 Differentially Oblivious Locate Algorithm $\mathsf{Locate}_{\mathcal{P}}$.

Input: dataset $\mathbf{x} = (x_1, \ldots, x_n)$

Let $c \leftarrow 0$, $\varepsilon' = \frac{\varepsilon}{2\log(2/\delta)}$, and $T \leftarrow \frac{1}{\varepsilon'} \ln \frac{\log n}{\delta}$
for $i = 1$ to $n/2$ **do**
  Choose $j \in [n]$ with uniform distribution
  if $\mathcal{P}(x_j) = 1$ **then**
    $c \leftarrow c + 1$
  end if
  if i is an integral power of 2 **then**
    $\hat{T} \leftarrow T + \mathrm{Lap}(\frac{1}{\varepsilon'})$
    if $c > \max(\hat{T}, 0)$ **then**
      output 1
    end if
  end if
end for
Scan the entire dataset, if there is an element satisfying $\mathcal{P}$ then output 1, else output 0

satisfying $\mathcal{P}$. If after $n/2$ probes the number of elements satisfying $\mathcal{P}$ is below the threshold the entire dataset is scanned. Our algorithm $\mathsf{Locate}_{\mathcal{P}}$ is described in Algorithm 3. On a given array $\mathbf{x}$, algorithm $\mathsf{Locate}_{\mathcal{P}}$ outputs 1 iff there exists an element in $\mathbf{x}$ satisfying the property $\mathcal{P}$.

We remark that Algorithm $\mathsf{Locate}_{\mathcal{P}}$ uses a mechanism similar to the the sparse vector mechanism of [18]. However, in our case instead of using a single noisy threshold across all steps, Algorithm $\mathsf{Locate}_{\mathcal{P}}$ generates in each step a noisy threshold $\hat{T} = T + \mathrm{Lap}(\frac{1}{\varepsilon'})$. The value of T ensures that with high probability $\hat{T} > 0$. The proof of Theorem 15 is given in Appendix A.3.

▶ **Theorem 15.** *Algorithm $\mathsf{Locate}_{\mathcal{P}}$ is an $(\varepsilon, \delta(1 + e^{\varepsilon}))$-differentially oblivious algorithm that outputs 1 iff there exists an element in the array that satisfies property $\mathcal{P}$. For $m = \Omega(T/p \log(T/p))$, with probability $1 - e^{-\Omega(mp)}$ it halts in time at most m, where $T = \frac{2\log(2/\delta)}{\varepsilon} \ln \frac{\log n}{\delta}$.*

6 Differentially Oblivious Prefix Sum

Suppose that there is a dataset consisting of sorted sensitive user records, and one would like to compute the sum of all records in the (sorted) dataset that are less than or equal to a value a in a way that respects individual user's privacy. We call this task differentially oblivious prefix sum. For the definition of privacy we say that two datasets of size n are neighbors if they agree on $n - 1$ elements (although, as sorted arrays they can disagree on many indices). For example, $(1, 2, 3, 4)$ and $(1, 3, 4, 5)$ are neighbors and should have similar access pattern.

Without privacy one can find the greatest record less than or equal to value a, and then compute the prefix sum by a quick scan through all records appearing before such record. Any perfectly secure algorithm must read the entire dataset (since it is possible that all elements are smaller than a). Here, we give a differentially oblivious prefix sum algorithm that for many instances is much faster than any perfectly oblivious algorithm.

Algorithm 4 Differentially Oblivious Search Algorithm SEARCH.

Input: a dataset $\mathbf{x} = (x_1, \ldots, x_n)$ and a value a

Let $\varepsilon' \leftarrow \frac{\varepsilon}{2.5 \log n}$, $\delta' \leftarrow \frac{\delta}{2.5 \log n}$, $k \leftarrow \lceil \frac{4 \log(1/\delta')}{\varepsilon'} \rceil$, min $\leftarrow 0$, and max $\leftarrow n$
while max $-$ min $> k$ **do**
  $c \leftarrow \lfloor (\max - \min)/k \rfloor$
  Let $\mathbf{y} = (y_1, \ldots, y_k)$, where $y_i = x_{\min + i \cdot c}$ for every $i \in [k]$
  Scan the entire dataset $\mathbf{y}$ and find the maximal index I such that $y_I \leq a$; if there is no such element then $I \leftarrow 0$
  noise $\leftarrow \mathrm{Lap}(\frac{1}{\varepsilon'})$
  min $= \max\{0, \min + \lfloor (I + \text{noise} - \frac{\log 1/\delta'}{\varepsilon'}) \cdot c \rfloor\}$ and max $= \min\{n, \min + \lfloor (I + \text{noise} + \frac{\log 1/\delta'}{\varepsilon'} + 1) \cdot c \rfloor\}$
end while
Scan the entire dataset $\mathbf{x}$ between min and max and return the the maximal index I such that $x_I \leq a$; if there is no such element then $I \leftarrow 0$

Intuition. Absent privacy requirements, using binary search, one can find the greatest element less than or equal to a, and then compute the prefix sum by a quick scan through all records that appear before such record. However, the binary search access pattern allows the adversary to gain sensitive information about the input. Our main idea is to approximately simulate the binary search and obfuscate the memory accesses to obtain differential obliviousness. In order to do that, we first divide the input array into k chunks (where k is polynomial in $1/\varepsilon, \log 1/\delta$, and $\log n$). Then, we find the chunk that contains the greatest element less than or equal to a by comparing the first element (hence, the smallest element) of each chunk to a. Let I be the index of such chunk. Next, we compute a noisy interval that contains I using the Laplacian distribution. We iteratively repeat this process on the noisy interval, where in each step we eliminate more than a quarter of the elements of the interval. We continue until the size of the array is less than or equal to k. Next, we scan all elements in the remaining array and find the index of the greatest element smaller than or equal to a. Let i be the index of such element; we compute the prefix sum by scanning the array $\mathbf{x}$ until index i.

The Search Algorithm. We present a search algorithm in Algorithm 4; on input $\mathbf{x} = (x_1, \ldots, x_n)$ and a this algorithm finds the largest index I such that $x_I \leq a$. To compute the prefix sum, we compute $\hat{I} = I + \mathrm{Lap}(1/\varepsilon) + \frac{\log 1/\delta}{\varepsilon}$ and scan the first $\hat{I}$ elements of the dataset, summing only the first I. We show in Theorem 17 that our search algorithm is (ε, δ)-differentially oblivious.

▶ Remark 16. We prove that algorithm SEARCH is an $(\varepsilon, 0)$-differentially private algorithm that returns a correct index with probability at least $1 - \beta$. We could change it to an (ε, δ)-differentially private algorithm that never errs. This is done by truncating the noise to $\frac{\log 1/\delta'}{\varepsilon'}$.

▶ **Theorem 17.** *Let $\beta < 1/n$ and $\varepsilon < \log^2 n$. Algorithm* SEARCH *is an $(\varepsilon, 0)$-differentially oblivious algorithm that, for any input array with size n and $a \in \mathbb{R}$, returns a correct index with probability at least $1 - \beta$. The running time of Algorithm* SEARCH *is $O(\frac{1}{\varepsilon} \log^2 n \log \frac{1}{\beta})$.*

Theorem 17 is proved in Appendix A.4.

Algorithm 5 Differentially Oblivious Search Algorithm MultiSearch for Multiple Queries.

Input: a dataset $\mathbf{x} = (x_1, \ldots, x_n)$

$t \leftarrow 1$ and $M \leftarrow 0$
for every query a **do**
  if the greatest element in the ORAM is greater than a or all records are in the ORAM (that is $M = n$) **then**
    answer the query using the ORAM
  else
    execute algorithm Search with privacy parameter $\frac{\varepsilon}{t \log n}$ and accuracy parameter $\beta/\sqrt{n}$ for the database starting at record $M+1$ and let I the largest index in this database such that $x_I \le a$
    insert the first $\max\{I, 2t\}$ elements of this database to the ORAM; for each element also insert the sum of all elements in the array up to this element
    $t \leftarrow t+1$, $M \leftarrow M + \max\{I, 2t\}$
  end if
end for

6.1 Dealing with Multiple Queries

We extend our prefix sum algorithm to answer multiple queries. We can answer a bounded number of queries by running the differentially oblivious prefix sum algorithm multiple times. That is, when we want an $(\varepsilon, 0)$-oblivious algorithm correctly answering t queries with probability at least $1-\beta$, we execute algorithm Search t times with privacy parameter ε/t and error probability β/t (each time also computing the appropriate prefix sum). Thus, the running time of the algorithm is $O(\frac{t^2}{\varepsilon} \log^2 n \log \frac{t}{\beta})$ (excluding the scan time for computing the sum).

On the other hand, we can use an ORAM to answer unbounded number of queries. That is, in a pre-processing stage we store the n records and for each record we store the sum of all records up to this record. Thereafter, answering each query will require one binary search. Using the ORAM of [2], the pre-processing will take time $O(n \log n)$ and answering each query will take time $O(\log^2 n)$. Thus, the ORAM algorithm is more efficient when $t \ge \sqrt{n}$.

We use ORAM along with our differentially oblivious prefix sum algorithm to answer unbounded number of queries while preserving privacy, combining the advantages of both of the previous algorithms.

▶ **Theorem 18.** *Algorithm* MultiSearch*, described in Algorithm 5, is an* $(\varepsilon, 0)$*-oblivious algorithm, which executes Algorithm* Search *at most* $O(\sqrt{n})$ *times, where the run time of the* t*-th execution is* $O(\frac{t}{\varepsilon} \log^3 n \log \frac{n}{\beta})$*, scans the original database at most once, and in addition each query run time is at most* $O(\log^2 n)$*.*

Proof. First note that we only pay for privacy in the executions of algorithm Search (reading and writing to the ORAM is perfectly private). In the t-th execution of algorithm Search, we insert at least $2t$ elements to the ORAM, thus after $\sqrt{n}$ executions we inserted at least $\sum_{t=1}^{\sqrt{n}} 2t = n$ elements to the ORAM.

By simple composition, algorithm MultiSearch is $(\varepsilon', 0)$-differentially private, where

$$\varepsilon' = \sum_{t=1}^{\sqrt{n}} \frac{\varepsilon}{t \log n} \le \frac{\varepsilon}{\log n}(\ln \sqrt{n} + 1) \le \varepsilon,$$

where the last inequality is implied by the sum of the harmonic series. ◀

References

1. Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. Efficient Testing of Large Graphs. *Combinatorica*, 20(4):451–476, 2000. doi:10.1007/s004930070001.
2. Gilad Asharov, Ilan Komargodski, Wei-Kai Lin, Kartik Nayak, and Elaine Shi. OptORAMa: Optimal Oblivious RAM. *IACR Cryptology ePrint Archive*, 2018:892, 2018. URL: https://eprint.iacr.org/2018/892.
3. Marina Blanton, Aaron Steele, and Mehrdad Aliasgari. Data-oblivious graph algorithms for secure computation and outsourcing. In Kefei Chen, Qi Xie, Weidong Qiu, Ninghui Li, and Wen-Guey Tzeng, editors, *8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '13*, pages 207–218. ACM, 2013. doi:10.1145/2484313.2484341.
4. David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-Abuse Attacks Against Searchable Encryption. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015*, pages 668–679. ACM, 2015.
5. T.-H. Hubert Chan, Kai-Min Chung, Bruce M. Maggs, and Elaine Shi. Foundations of Differentially Oblivious Algorithms. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019*, pages 2448–2467. SIAM, 2019.
6. Srinivas Devadas, Marten van Dijk, Christopher W. Fletcher, Ling Ren, Elaine Shi, and Daniel Wichs. Onion ORAM: A Constant Bandwidth Blowup Oblivious RAM. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A*, volume 9563 of *Lecture Notes in Computer Science*, pages 145–174. Springer, 2016. doi:10.1007/978-3-662-49099-0_6.
7. Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006. doi:10.1007/11681878_14.
8. David Eppstein, Michael T. Goodrich, and Roberto Tamassia. Privacy-preserving data-oblivious geometric algorithms for geographic data. In Divyakant Agrawal, Pusheng Zhang, Amr El Abbadi, and Mohamed F. Mokbel, editors, *18th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2010*, pages 13–22. ACM, 2010. doi:10.1145/1869790.1869796.
9. Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal Aggregation Algorithms for Middleware. In Peter Buneman, editor, *Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 102–113. ACM, 2001.
10. Oded Goldreich. Towards a Theory of Software Protection and Simulation by Oblivious RAMs. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 182–194. ACM, 1987. doi:10.1145/28395.28416.
11. Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017. doi:10.1017/9781108135252.
12. Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property Testing and its Connection to Learning and Approximation. *J. ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.
13. Oded Goldreich and Rafail Ostrovsky. Software Protection and Simulation on Oblivious RAMs. *J. ACM*, 43(3):431–473, 1996. doi:10.1145/233551.233553.
14. Oded Goldreich and Dana Ron. Property Testing in Bounded Degree Graphs. *Algorithmica*, 32(2):302–343, 2002.
15. M. T. Goodrich, O. Ohrimenko, and R. Tamassia. Data-oblivious graph drawing model and algorithms. *CoRR*, 2012.
16. Michael T. Goodrich. Zig-zag sort: a simple deterministic data-oblivious sorting algorithm running in $O(n \log n)$ time. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014*, pages 684–693. ACM, 2014. doi:10.1145/2591796.2591830.

17 Michael T. Goodrich and Joseph A. Simons. Data-Oblivious Graph Algorithms in Outsourced External Memory. In Zhao Zhang, Lidong Wu, Wen Xu, and Ding-Zhu Du, editors, *Combinatorial Optimization and Applications - 8th International Conference, COCOA 2014*, volume 8881 of *Lecture Notes in Computer Science*, pages 241–257. Springer, 2014. doi:10.1007/978-3-319-12691-3_19.

18 Moritz Hardt and Guy N. Rothblum. A Multiplicative Weights Mechanism for Privacy-Preserving Data Analysis. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010*, pages 61–70. IEEE Computer Society, 2010.

19 Xi He, Ashwin Machanavajjhala, Cheryl Flynn, and Divesh Srivastava. Composing Differential Privacy and Secure Computation: A Case Study on Scaling Private Record Linkage. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1389–1406. ACM, 2017. doi:10.1145/3133956.3134030.

20 Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. Inference attack against encrypted range queries on outsourced databases. In Elisa Bertino, Ravi S. Sandhu, and Jaehong Park, editors, *Fourth ACM Conference on Data and Application Security and Privacy, CODASPY'14*, pages 235–246. ACM, 2014. doi:10.1145/2557547.2557561.

21 Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic Attacks on Secure Outsourced Databases. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1329–1340. ACM, 2016. doi:10.1145/2976749.2978386.

22 Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Accessing Data while Preserving Privacy. *CoRR*, abs/1706.01552, 2017. arXiv:1706.01552.

23 Marie-Sarah Lacharité, Brice Minaud, and Kenneth G. Paterson. Improved Reconstruction Attacks on Encrypted Data Using Range Query Leakage. In *2018 IEEE Symposium on Security and Privacy, SP 2018*, pages 297–314. IEEE Computer Society, 2018. doi:10.1109/SP.2018.00002.

24 Kasper Green Larsen and Jesper Buus Nielsen. Yes, There is an Oblivious RAM Lower Bound! In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference*, volume 10992 of *Lecture Notes in Computer Science*, pages 523–542. Springer, 2018. doi:10.1007/978-3-319-96881-0_18.

25 Wei-Kai Lin, Elaine Shi, and Tiancheng Xie. Can We Overcome the $n \log n$ Barrier for Oblivious Sorting? In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019*, pages 2419–2438. SIAM, 2019. doi:10.1137/1.9781611975482.148.

26 Chang Liu, Xiao Shaun Wang, Kartik Nayak, Yan Huang, and Elaine Shi. ObliVM: A Programming Framework for Secure Computation. In *2015 IEEE Symposium on Security and Privacy, SP 2015*, pages 359–376. IEEE Computer Society, 2015.

27 Sahar Mazloom and S. Dov Gordon. Secure Computation with Differentially Private Access Patterns. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 490–507. ACM, 2018. doi:10.1145/3243734.3243851.

28 Tarik Moataz, Travis Mayberry, and Erik-Oliver Blass. Constant Communication ORAM with Small Blocksize. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 862–873. ACM, 2015.

29 Muhammad Naveed, Seny Kamara, and Charles V. Wright. Inference Attacks on Property-Preserving Encrypted Databases. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security 2015*, pages 644–655. ACM, 2015.

30 Rafail Ostrovsky. Efficient Computation on Oblivious RAMs. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 514–523. ACM, 1990. doi:10.1145/100216.100289.

31 Giuseppe Persiano and Kevin Yeo. Lower Bounds for Differentially Private RAMs. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 404–434. Springer, 2019. doi:10.1007/978-3-030-17653-2_14.

32 Sofya Raskhodnikova and Adam D. Smith. A Note on Adaptivity in Testing Properties of Bounded Degree Graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(089), 2006. URL: http://eccc.hpi-web.de/eccc-reports/2006/TR06-089/index.html.

A Missing Proofs

A.1 Proof of Lemma 5

Proof. Let $\mathbf{x}$ and $\mathbf{y}$ be two neighboring datasets and S be a sets of outputs. Then,

$$\begin{aligned}\Pr[\mathcal{B}(\mathbf{x}) \in S] &\leq \Pr[\mathcal{A}(\mathbf{x}) \in S] + \gamma \\ &\leq e^{\varepsilon} \Pr[\mathcal{A}(\mathbf{y}) \in S] + \gamma \\ &\leq e^{\varepsilon}(\Pr[\mathcal{B}(\mathbf{y}) \in S] + \gamma) + \gamma \\ &= e^{\varepsilon} \Pr[\mathcal{B}(\mathbf{y}) \in S] + (1 + e^{\varepsilon})\gamma.\end{aligned}$$

◀

A.2 Proof of the Correctness and Privacy of Algorithm Tester$_{\mathcal{T}}$

Theorem 8 is implied by the following lemmas.

▶ **Lemma 19.** *Algorithm Tester$_{\mathcal{T}}$ is $(\varepsilon, \delta(1 + e^{\varepsilon}))$-differentially oblivious.*

Proof. We first analyze a variant of Tester$_{\mathcal{T}}$, denoted by Tester$'_{\mathcal{T}}$, in which Step 10 is replaced by "If $c > \hat{T}$ then output 1" (that is, the algorithm does not check if $c > \min\{4T, \hat{T}\}$ before deciding in the positive).

Let $G = (V, E)$ and $G' = (V', E')$ be two neighboring graphs such that they differ on node $v \in V$. Fix the random choices of subsets A in Step 6 and observe that after the execution of for loop, the count c can differ by at most 1 between the executions on G and G'. Let $\tilde{T}$ be the smallest integer greater than $\hat{T}$. Since algorithm Tester$'_{\mathcal{T}}$ uses the Laplace mechanism $e^{-\varepsilon} \Pr[\tilde{T} < a] \leq \Pr[\tilde{T} < a - 1] \leq e^{\varepsilon} \Pr[\tilde{T} < a]$ for every a. Thus,

$$\begin{aligned}\Pr[\textsc{Tester}'_{\mathcal{T}}(G) = 1] &= \sum_a \Pr[\tilde{T} = a] \Pr[c(G) > a] \\ &\leq \sum_a \Pr[\tilde{T} = a] \Pr[c(G') > a - 1] \\ &\leq e^{\varepsilon} \sum_a \Pr[\tilde{T} = a - 1] \Pr[c(G') > a - 1] \\ &\leq e^{\varepsilon} \Pr[\textsc{Tester}'_{\mathcal{T}}(G') = 1].\end{aligned}$$

Similarly, $\Pr[\textsc{Tester}'_{\mathcal{T}}(G) = 1] \geq e^{-\varepsilon} \Pr[\textsc{Tester}'_{\mathcal{T}}(G') = 1]$. Hence, Tester$'_{\mathcal{T}}$ is $(\varepsilon, 0)$-differentially oblivious.

We next prove that Tester$_{\mathcal{T}}$ is $(\varepsilon, \delta(1 + e^{\varepsilon}))$-differentially oblivious using Lemma 5, that is we prove that for every graph G, the statistical distance between Tester$_{\mathcal{T}}(G)$ and Tester$'_{\mathcal{T}}(G)$ is at most δ. Let E be the event that $\hat{T} > 4T$ and observe that the probability

E occurs is at most δ.[7] We have that $\left|\Pr[\text{Tester}_{\mathcal{T}}(G) = 1] - \Pr[\text{TESTER}'_{\mathcal{T}}(G) = 1]\right| \leq \left|\Pr[\text{Tester}_{\mathcal{T}}(G) = 1|E] - \Pr[\text{TESTER}'_{\mathcal{T}}(G) = 1|E]\right| \Pr[E] \leq \Pr[E] \leq \delta$. Thus, by Lemma 5, algorithm $\text{Tester}_{\mathcal{T}}$ is $(\varepsilon, \delta(1+e^{\varepsilon}))$-differentially oblivious. ◀

Observe that Algorithm $\text{Tester}_{\mathcal{T}}$ never errs when $G \in \mathcal{P}$ as in that case after the for loop is executed $c = 4T$ and hence in Step 10 $\text{Tester}_{\mathcal{T}}$ outputs 1. The next lemma analyses the error probability when G is γ'-far from $\mathcal{P}$.

▶ **Lemma 20.** *Algorithm $\text{Tester}_{\mathcal{T}}$ is $(1-\delta-(2\delta)^{\frac{1}{3\varepsilon}}, \gamma')$-tester for the graph property $\mathcal{P}$.*

Proof. Observe that on Step 7 of the algorithm, we are eliminating at most $n \cdot c_{\beta,\gamma}$ edges. Thus, we are eliminating at most $4Tnc_{\beta,\gamma}$ edges in total. Then, when G is γ'-far from $\mathcal{P}$, it is also γ-far from $\mathcal{P}$ after the removal of the observed nodes in each step of the for loop. We next prove that Algorithm $\text{Tester}_{\mathcal{T}}$ fails with probability at most $2\delta^{\frac{1}{3\varepsilon}}$. Observe that if Algorithm $\text{Tester}_{\mathcal{T}}$ fails on G then $c \geq 2T$ or $\text{Lap}(\frac{1}{\varepsilon}) \leq -T$. We define Z_i to be output of $\mathcal{T}(G)$ in the i-th step of the for loop. Let $Z = \sum_i Z_i$. Observe that all Z_i are independent and $\mathbb{E}[Z] \leq T$. Using the Chernoff Bounds[8], we obtain that $\Pr[Z \geq 2T] \leq e^{-T/3} = (2\delta)^{\frac{1}{3\varepsilon}}$. We also know $\Pr[\text{Lap}(\frac{1}{\varepsilon}) \leq -\frac{\ln(1/2\delta)}{\varepsilon}] = 0.5e^{-\ln(1/2\delta)} = \delta$. Therefore, Algorithm $\text{Tester}_{\mathcal{T}}$ fails with probability $\delta + (2\delta)^{\frac{1}{3\varepsilon}}$. ◀

A.3 Proof of the Correctness and Privacy of Algorithm $\text{Locate}_{\mathcal{P}}$

The proof of Theorem 15 follows from the following claim and lemmas.

▷ Claim 21. Let $\ell \geq \log n / \log\log n$. The probability that there exists an element $j \in [n]$ such that algorithm $\text{Locate}_{\mathcal{P}}$ samples the element j in Step 3 more than 2ℓ times is less that $2^{-\ell}$.

Proof. Fix an index j. The probability that the element j is sampled more than 2ℓ times is less than $\binom{n/2}{2\ell}\frac{1}{n^{2\ell}} \leq \left(\frac{en}{4\ell}\right)^{2\ell}\frac{1}{n^{2\ell}} < \ell^{-2\ell} < 2^{-2\log n+2} < 2^{2-\ell}/n$. The claim follows by the union bound. ◁

▶ **Lemma 22.** *Let $\delta < 1/n$. Algorithm $\text{Locate}_{\mathcal{P}}$ is $(\varepsilon, \delta(1+e^{\varepsilon}))$-differentially oblivious.*

Proof. We first analyze a variant of $\text{Locate}_{\mathcal{P}}$, denoted by $\text{LOCATE}'_{\mathcal{P}}$, in which Step 9 is replaced by "If $c > \hat{T}$ then output 1" (that is, the algorithm does not check if $\hat{T} > 0$) and no element is sampled more than $2\log(2/\delta)$ times. We analyze the privacy of $\text{LOCATE}'_{\mathcal{P}}(\mathbf{x}')$ similarly to the analysis of the sparse vector mechanism in [18].

Let $\mathbf{x}$ and $\mathbf{x}'$ be two neighboring datasets that such that $\mathcal{P}(x_j) = 1$ and $\mathcal{P}(x'_j) = 0$ for some j. Denote by $\tau = (\tilde{T}_1, \ldots, \tilde{T}_{\log n})$ the values of the thresholds in an execution of $\text{LOCATE}'_{\mathcal{P}}$, where each threshold is rounded up to the smallest integer greater than $\hat{T}$. Furthermore, let $\ell_\tau \in [\log n]$ be the index such that $\text{LOCATE}'_{\mathcal{P}}$ on input $\mathbf{x}$ outputs 1 when $i = 2^{\ell_\tau}$ (if no such i exists, then $\ell_\tau \in \lceil \log n \rceil + 1$). Observe that in each execution of Step 9 the count c on input $\mathbf{x}$ is at least the count on input $\mathbf{x}'$ and can exceed it by at most $2\log(2/\delta)$ (since j is sampled at most $2\log(2/\delta)$ times). Thus, $\text{LOCATE}'_{\mathcal{P}}$ on input $\mathbf{x}'$ with thresholds $\tau' = (\tilde{T}_1, \ldots, \tilde{T}_{\ell_\tau - 1}, \tilde{T}_{\ell_\tau} - 2\log(2/\delta), \tilde{T}_{\ell_\tau+1}, \ldots, \tilde{T}_{\log n})$ outputs 1 when $i = 2^{\ell_\tau}$. Since algorithm $\text{LOCATE}'_{\mathcal{P}}$ uses the Laplace mechanism with $\varepsilon' = \varepsilon/(2\log(1/\delta))$,

$$e^{-\varepsilon}\Pr[\tilde{T}_{\ell_\tau} = a] \leq \Pr[\tilde{T}_{\ell_\tau} = a - 2\log(2/\delta)] \leq e^{\varepsilon}\Pr[\tilde{T}_{\ell_\tau} = a]$$

[7] $\Pr[\text{Lap}(\frac{1}{\varepsilon}) \geq t/\varepsilon] = \frac{1}{2}e^{-t}$ for every $t > 0$. Thus, $\Pr[E] = \Pr[\text{Lap}(\frac{1}{\varepsilon}) \geq \frac{\ln(1/2\delta)}{\varepsilon}] = \delta$.

[8] $\Pr[Z \geq (1+\eta)\mu] \leq e^{-\eta^2\mu/(2+\eta)}$ for any $\eta > 0$ where μ is the expectation of Z.

for every a. Thus,

$$\begin{aligned}
&\Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}) \in S] \\
&\quad= \sum_{\tau=(\tilde{T}_1,\ldots,\tilde{T}_{\log n})} \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}) \in S \,|\, \tilde{T}_1,\ldots,\tilde{T}_{\log n}] \Pr[\tilde{T}_1,\ldots,\tilde{T}_{\log n}] \\
&\quad= \sum \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}') \in S \,|\, \tilde{T}_1,\ldots,\tilde{T}_{\ell_\tau}-1,\tilde{T}_{\ell_\tau}-2\log(2/\delta),\tilde{T}_{\ell_\tau+1},\ldots,\tilde{T}_{\log n}] \\
&\qquad\qquad \cdot \Pr[\tilde{T}_1,\ldots,\tilde{T}_{\log n}] \\
&\quad\le e^{\varepsilon} \sum \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}') \in S \,|\, \tilde{T}_1,\ldots,\tilde{T}_{\ell_\tau}-1,\tilde{T}_{\ell_\tau}-2\log(2/\delta),\tilde{T}_{\ell_\tau+1},\ldots,\tilde{T}_{\log n}] \\
&\qquad\qquad \cdot \Pr[\tilde{T}_1,\ldots,\tilde{T}_{\ell_\tau}-1,\tilde{T}_{\ell_\tau}-2\log(2/\delta),\tilde{T}_{\ell_\tau+1},\ldots,\tilde{T}_{\log n}] \\
&\quad= e^{\varepsilon} \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}') \in S].
\end{aligned}$$

Similarly,

$$\begin{aligned}
&\Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}) \in S] \\
&\quad\ge e^{-\varepsilon} \sum \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}') \in S \,|\, \tilde{T}_1,\ldots,\tilde{T}_{\ell_\tau}-1,\tilde{T}_{\ell_\tau}-2\log(2/\delta),\tilde{T}_{\ell_\tau+1},\ldots,\tilde{T}_{\log n}] \\
&\qquad\qquad \cdot \Pr[\tilde{T}_1,\ldots,\tilde{T}_{\ell_\tau}-1,\tilde{T}_{\ell_\tau}-2\log(2/\delta),\tilde{T}_{\ell_\tau+1},\ldots,\tilde{T}_{\log n}] \\
&\quad= e^{-\varepsilon} \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}') \in S].
\end{aligned}$$

We next prove that $\text{Locate}_{\mathcal{P}}$ is $(\varepsilon, \delta(1+e^{\varepsilon}))$-differentially oblivious using Lemma 5. I.e, we prove that for every dataset $\mathbf{x}$, the statistical distance between $\mathsf{Access}^{\text{Locate}_{\mathcal{P}}}(\mathbf{x})$ and $\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x})$ is at most δ. Notice that if all the thresholds are positive and all elements are sampled at most $2\log(2/\delta)$ times then $\text{Locate}_{\mathcal{P}}(\mathbf{x})$ and $\textsc{Locate}'_{\mathcal{P}}(\mathbf{x})$ have the same access pattern. By Claim 21, the probability that there exists a j that is sampled more than $2\log(2/\delta)$ is at $2^{-\log(2/\delta)} = \delta/2$. We next observe that the probability that a threshold $\hat{T} = T + \text{Lap}(\frac{1}{\varepsilon'})$ is negative is at most $\delta/2$. Recall that $\Pr[\text{Lap}(\frac{1}{\varepsilon'}) \le -t/\varepsilon'] = \frac{1}{2}e^{-t}$ for every $t > 0$. Thus, $\Pr[\hat{T} \le 0] = \Pr[\text{Lap}(\frac{1}{\varepsilon'}) \le -\frac{1}{\varepsilon}\ln(\frac{\log n}{\delta})] = \frac{\delta}{2\log n}$. Let A be the event that at least one of the $\log n$ thresholds $\hat{T}$ is at most 0 or some j is sampled more that $2\log(2/\delta)$ times. By the union bound the probability of A is at most δ. Therefore, for every set of access patterns S

$$\begin{aligned}
&|\Pr[\mathsf{Access}^{\text{Locate}_{\mathcal{P}}}(\mathbf{x}) \in S] - \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}) \in S]| \\
&\quad= \Big|\Pr[\mathsf{Access}^{\text{Locate}_{\mathcal{P}}}(\mathbf{x}) \in S|A]\Pr[A] + \Pr[\mathsf{Access}^{\text{Locate}_{\mathcal{P}}}(\mathbf{x}) \in S|\bar{A}]\Pr[\bar{A}] \\
&\qquad - \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}) \in S|A]\Pr[A] - \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}) \in S|\bar{A}]\Pr[\bar{A}]\Big| \\
&\quad= \Big|\Pr[\mathsf{Access}^{\text{Locate}_{\mathcal{P}}}(\mathbf{x}) \in S|A] - \Pr[\mathsf{Access}^{\textsc{Locate}'_{\mathcal{P}}}(\mathbf{x}) \in S|A]\Big|\Pr[A] \\
&\quad\le \Pr[A] \le \delta.
\end{aligned}$$

Thus, by Lemma 5, algorithm $\text{Locate}_{\mathcal{P}}$ is $(\varepsilon, \delta(1+e^{\varepsilon}))$-differentially oblivious. ◀

We next analyze the running and probe complexity of our algorithm. Let p be the probability that a uniformly chosen element in $\mathbf{x}$ satisfies $\mathcal{P}$. The non-private algorithm that samples elements until it finds an element satisfying $\mathcal{P}$ has expected running time $1/p$ and the probability that it does not stop after m steps is $(1-p)^m = ((1-p)^{1/p})^{mp} \le e^{-mp}$. We show that $\text{locate}_{\mathcal{P}}$ has a similar behavior.

▶ **Lemma 23.** *Let p be the probability that a uniformly chosen element in $\mathbf{x}$ satisfies $\mathcal{P}$. Then, for every integral power of two m the probability that algorithm locate$_{\mathcal{P}}$ probes more than m memory locations is less than $\delta/\log n + e^{-(m-2T)p+2T\ln m}$. In particular, for $m = \Omega(\frac{T}{p}\log(\frac{T}{p}))$, the probability is less than $\delta/\log n + e^{-O(mp)}$.*

Proof. Let $t = 2^i$. The probability that $\hat{T} \geq 2T$ is $\Pr[\mathrm{Lap}(\frac{1}{\varepsilon'}) \geq \frac{1}{\varepsilon'}\ln\frac{\log n}{\delta}] = 0.5e^{-\ln(\log n/\delta)} = \frac{\delta}{\log n}$. Assuming that $\hat{T} \geq 2T$, the probability that the algorithm does not halt after $m = 2^i$ steps is less than

$$\binom{m}{2T}(1-p)^{m-2t} \leq m^{2T}e^{-(m-2T)p} \leq e^{-(m-2T)p+2T\ln m}. \qquad \blacktriangleleft$$

A.4 Proof of the Correctness and Privacy of Algorithm Search

Theorem 17 is proved in the next 3 claims. We start by analyzing the running time of the algorithm.

▷ Claim 24. Let $\beta < 1/n$ and $\varepsilon < \log^2 n$. The while loop in Algorithm Search is executed at most $2.5\log n$ time. Furthermore, the total running time of the algorithm is $O(\frac{1}{\varepsilon}\log^2 n\log\frac{1}{\beta})$.

Proof. Let $\min_0, \max_0$ and $\min_1, \max_1$ be the values of min, max before and after an execution of a step of the while loop in Algorithm Search. Note that

$$\max_1 - \min_1 \leq 1 + (2\cdot\frac{\log 1/\beta'}{\varepsilon'} + 1)\cdot\frac{\max_0 - \min_0}{\frac{4\log(1/\beta')}{\varepsilon'}} \leq 3\cdot\frac{\log 1/\beta'}{\varepsilon'}\cdot\frac{\max_0 - \min_0}{\frac{4\log(1/\beta')}{\varepsilon'}} = \frac{3(\max_0 - \min_0)}{4}.$$

Therefore, algorithm Search eliminates more than a quarter of the elements in each step of the while loop and the algorithm will halt after less than $2.5\log n$ steps.

Moreover, observe that Algorithm Search makes k memory accesses in each step of the while loop and additional k memory accesses after the loop. Thus, its running time is $O(\frac{1}{\varepsilon}\log^2 n(\log\log n + \log\frac{1}{\beta})) = O(\frac{1}{\varepsilon}\log^2 n\log\frac{1}{\beta})$ (since $\beta < 1/n$). ◁

▷ Claim 25. Algorithm Search returns the correct index with probability at least $1-\beta$.

Proof. Let $\bar{I}$ be the maximal index such that $x_{\bar{I}} \leq a$ (i.e., $\bar{I}$ is the index that algorithm Search should return). We prove by induction that if all Laplace noises in the algorithm satisfy $|\mathrm{Lap}(\frac{1}{\varepsilon'})| < \frac{\log 1/\beta'}{\varepsilon'}$ then in each step of the algorithm $\min \leq \bar{I} \leq \max$, hence the algorithm will return $\bar{I}$ in its last scan of $\mathbf{x}$ between min and max.

The basis of the induction is trivial since $0 \leq \bar{I} \leq n$. For the induction step, let $\min_0, \max_0$ and $\min_1, \max_1$ be the values of min, max before and after an execution of a step of the while loop in Algorithm Search. By the induction hypothesis, $\min_0 \leq \bar{I} \leq \max_0$. The algorithm finds an index I such that $\min_0 + Ic \leq \bar{I} \leq \min_0 + (I+1)c$. By our assumption on the Laplace noise, $\min_1 \leq \min_0 + Ic$, thus, $\min_1 \leq \bar{I}$. Similarly, $\max_1 \geq \min_0 + (I+1)c$, thus, $\max_1 \geq \bar{I}$.

Recall that $\Pr[|\mathrm{Lap}(\frac{1}{\varepsilon'})| \geq t/\varepsilon'] = e^{-t}$ for every $t > 0$. Thus, by Claim 24 and the union bound, the probability that one of the Laplace noises is greater than $\frac{\log 1/\beta'}{\varepsilon'}$ is at most $(2.5\log n)\cdot\beta' = \beta$. Hence, the probability that algorithm Search returns the correct index $\bar{I}$ is at least $1-\beta$. ◁

Next, we show that algorithm Search is $(\varepsilon, 0)$-differentially oblivious.

▷ Claim 26. Algorithm Search is an $(\varepsilon, 0)$-differentially oblivious algorithm.

Proof. We show below that each step of the while loop in algorithm SEARCH is $(\varepsilon', 0)$-differentially oblivious. Applying the basic composition theorem and Claim 24, we obtain that the SEARCH algorithm is $(\varepsilon = (2.5 \log n)\varepsilon', 0)$-differentially oblivious.

Fix a step of the loop and view it as an algorithm that returns min and max (given these values the access pattern of the next step is fixed). Let $\mathbf{x}$ and $\mathbf{x}'$ be two neighboring datasets such that for some j we have $x_j > x'_j$ and $x_i = x'_i$ for all $i < j$. It holds that $x_{i-1} \leq x'_i \leq x_i$ for every i. Let $I(\mathbf{x})$ and $I(\mathbf{x}')$ be the values computed in step 5 of the algorithm on inputs $\mathbf{x}$ and $\mathbf{x}'$ respectively. Thus, the value $I(\mathbf{x})$ is at least the value $I(\mathbf{x}')$ and can exceed it by one. Intuitively, since algorithm SEARCH uses the Laplace mechanism, the probabilities of returning a value min on $\mathbf{x}$ and $\mathbf{x}'$ are at most $e^{\pm\varepsilon'}$ apart. Formally, if $\mathrm{Lap}(1/\varepsilon') + I(\mathbf{x}) = \mathrm{Lap}(1/\varepsilon') + I(\mathbf{x})$ (where we consider two independent noises), then the algorithm returns the same value of min on both inputs. The lemma follows since for every set A:

$$e^{-\varepsilon'} \leq e^{-|I(\mathbf{x})-I(\mathbf{x}')|\varepsilon'} \leq \frac{\Pr[\mathrm{Lap}(1/\varepsilon') + I(\mathbf{x}) \in A]}{\Pr[\mathrm{Lap}(1/\varepsilon') + I(\mathbf{x}') \in A]} \leq e^{|I(\mathbf{x})-I(\mathbf{x}')|\varepsilon'} \leq e^{\varepsilon'}. \qquad \triangleleft$$

Thresholds in Random Motif Graphs

Michael Anastos
Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
http://www.math.cmu.edu/~manastos
manastos@andrew.cmu.edu

Peleg Michaeli
School of Mathematical Sciences, Tel Aviv University, Israel
http://www.math.tau.ac.il/~pelegm
peleg.michaeli@math.tau.ac.il

Samantha Petti
School of Mathematics, Georgia Institute of Technology, Atlanta, Georgia, USA
http://people.math.gatech.edu/~spetti3
spetti@gatech.edu

Abstract

We introduce a natural generalization of the Erdős-Rényi random graph model in which random instances of a fixed motif are added independently. The *binomial random motif graph* $G(H, n, p)$ is the random (multi)graph obtained by adding an instance of a fixed graph H on each of the copies of H in the complete graph on n vertices, independently with probability p. We establish that every monotone property has a threshold in this model, and determine the thresholds for connectivity, Hamiltonicity, the existence of a perfect matching, and subgraph appearance. Moreover, in the first three cases we give the analogous hitting time results; with high probability, the first graph in the random motif graph process that has minimum degree one (or two) is connected and contains a perfect matching (or Hamiltonian respectively).

2012 ACM Subject Classification Mathematics of computing → Random graphs; Mathematics of computing → Paths and connectivity problems

Keywords and phrases Random graph, Connectivity, Hamiltonicty, Small subgraphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.66

Category RANDOM

Funding *Peleg Michaeli*: This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement number 676970, RANDGEOM).
Samantha Petti: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1650044.

Acknowledgements We thank Alan Frieze for helpful discussions and for connecting the authors.

1 Introduction

In the late 1950's Gilbert [11] and Erdős and Rényi [6] introduced two of the most fundamental models for generating random graphs: the *binomial random graph* $G(n, p)$, generated by independently adding an edge between each pair of vertices in the complete graph on n vertices with probability p, and the the *uniform random graph* $G(n, m)$, which is a uniformly chosen graph from all graphs on n vertices with m edges. Since, the extensive study of these simple constructions has influenced a variety of fields including combinatorics, computer science, and statistical physics (see [9, 4, 12] for surveys).


© Michael Anastos, Peleg Michaeli, and Samantha Petti;
licensed under Creative Commons License CC-BY

Detailed analysis of the model has led to the development of plethora of new techniques in probability for analyzing random processes, and the model has been used to verify the existence of structures with certain properties [1]. In computer science, the model has been used to analyze the performance of algorithms on an "average" case, showing that NP complete problems may be easier random instances.

The rise of data in the form of graphs (e.g. internet connections, biological networks, social networks) has further fueled the study of random graphs. In practice, the comparison of real world networks to the Erdős-Rényi model is a popular technique for highlighting the non-random aspects of a network's structure [20, 2, 17, 14]. Moreover, the model has inspired many other models which are designed to mirror some characteristic of real-world networks (e.g. Watts-Strogatz graphs have small diameter [18], Barabási-Albert preferential attachment graph exhibit a power law degree distribution [3]).

In this paper we consider a natural generalization of the Erdős-Rényi model in which random motifs are added rather than random edges. A *motif* is a fixed small subgraph, such as a triangle. The motifs that are overrepresented in a network are correlated to the function of the network [20, 2, 17, 14]. Analyzing random graphs formed as the union of many instances of a particular motif H will give insight into the structural properties of networks with many copies of the motif H.

We define the ***binomial*** *random motif graph* $G(H,n,p)$ as the random (multi)graph obtained by adding an instance of H on each of the $\binom{n}{|V(H)|} \cdot |V(H)|!/\operatorname{aut}(H)$ copies of H in the complete graph on n vertices K_n, independently with probability p. Here by $\operatorname{aut}(H)$ we denote the number of automorphisms of H. Note that if H is an edge, then this is exactly $G(n,p)$. Similarly, the ***uniform*** *random motif graph* $\bar{G}(H,n,m)$ is the random (multi)graph obtained by taking the union of m uniformly chosen copies of H in K_n without replacement.

Closely related to $\bar{G}(H,n,m)$ is the *random motif graph process* $\bar{G}_0(H,n), \bar{G}_1(H,n), ...,$ $\bar{G}_N(H,n)$. $\bar{G}_0(H,n)$ is the empty graph on n vertices and for $0 \le i \le N = \binom{n}{|V(H)|}/\operatorname{aut}(H)$ the graph $\bar{G}_{i+1}(H,n)$ is generated by adding to $\bar{G}_i(H,n)$ a copy of H, H_{i+1}, chosen uniformly at random from all the copies of H except those in $\{H_1, H_2, ..., H_i\}$ i.e. those that have been added to $\bar{G}_0(H,n)$ so far. Clearly $\bar{G}_m(H,n)$ has the same law as $\bar{G}(H,n,m)$. In addition, by setting H to be an edge we retrieve the random graph process introduce by Erdős and Rényi [7]. By considering the random motif graph process in place of the uniform random motif graph model we can phrase results in a finer way (see for example Theorem 3).

In this work we show that every monotone graph property has a threshold in the binomial random motif graph $G(H,n,p)$. Then we determine the thresholds for connectivity, existence of a perfect matching, Hamiltoncity and subgraph appearance. In the first three cases we also show a hitting time result, according to which w.h.p.[1] the first graph in the random motif graph process that has minimum degree one (or two) is connected (or Hamiltonian respectively).

1.1 Notation

Throughout we assume the motif H has no isolated vertices. For an integer $r \ge 0$, denote by $m_r(H)$ the number of its copies in K_n which intersect the set $[r]$. For an integer $d \ge 0$ we define the quantities $\delta_d(H)$ and $p_d(H)$ by

$$\delta_d(H) := \lceil d/\delta(H) \rceil - 1 \qquad \text{and} \qquad p_d^{\pm}(H) := \frac{\ln n + \delta_d(H) \ln\ln n \pm x(n)}{m_1(H)},$$

[1] That is, with probability tending to 1 as n tends to infinity.

where $x(n)$ is any function of n satisfying $1 \ll x(n) \ll \ln \ln n$. Note that the expected number of added instances of H in $G(H, n, p_1^{\pm}(H))$ is $m_n(H) \cdot p_1^{\pm}(H)$, which only depends on n and on $|V(H)|$.

1.2 Results

A function $p^* = p^*(n)$ is a threshold for a monotone increasing property $\mathcal{P}$ in the random graph $G(H, n, p)$ if

$$\lim_{n\to\infty} \mathsf{Pr}[G(H,n,p) \in \mathcal{P}] = \begin{cases} 0 & \text{if } p/p^* \to 0, \\ 1 & \text{if } p/p^* \to \infty, \end{cases}$$

as $n \to \infty$. Our first result is a generalization of a theorem by Bollobás and Thomason [5].

▶ **Theorem 1.** *Every non-trivial monotone graph property has a threshold.*

Given Theorem 1, a natural goal is to find the thresholds for various monotone properties. The remaining results of this paper are dedicated towards this goal; we determine the threshold for connectivity, the existence of a perfect matchings, Hamiltonicity, and subgraph appearance.

A first such result, which generalizes a result in [6], shows, in particular, that the expected number of motifs needed to make the random motif graph connected depends only on the number of (non-isolated) vertices of the motif.

▶ **Theorem 2.** *Let H be a fixed graph. Then*

$$\lim_{n\to\infty} \mathit{Pr}[G(H,n,p) \text{ is connected }] = \begin{cases} 0 & p \le p_1^-(H), \\ 1 & p \ge p_1^+(H). \end{cases}$$

In fact, we show a hitting time result, according to which the hitting time of connectivity equals, w.h.p., the hitting time of minimum degree one. In other words, the random motif graph process becomes connected exactly when the last isolated vertex disappears, with high probability.

Fix an integer n and a graph H. Let $\tau_{\mathsf{c}} = \min\{i : \bar{G}_i(H,n) \text{ is connected}\}$, and for $d \ge 1$ denote $\tau_d = \min\{i : \delta(\bar{G}_i(H,n)) \ge d\}$.

▶ **Theorem 3.** *Let H be a fixed graph. Then w.h.p. $\tau_{\mathsf{c}} = \tau_1$.*

We remark that if the motif H is connected, every connectivity related question depends solely on the sets of vertices on which copies of H are added, and not on the way they are put there. Thus, we may model the question as a (binomial or uniform) random k-uniform hypergraph, where $k = |V(H)|$. In this case, Theorems 2 and 3 follow immediately from known results about (loose) connectivity in random hypergraphs (see, e.g.,[16]).

In the following two theorems we show that the existence of a perfect matching is also dependent on the number of non-isolated vertices of the motif.

▶ **Theorem 4.** *Let H be a fixed graph, and assume that n is even. Then,*

$$\lim_{n\to\infty} \mathit{Pr}[G(H,n,p) \text{ has a perfect matching }] = \begin{cases} 0 & p \le p_1^-(H), \\ 1 & p \ge p_1^+(H). \end{cases}$$

Let $\tau_{\mathsf{M}} = \min\{i : \bar{G}_i(H,n) \text{ has a perfect matching}\}$. The analogue hitting time result is also true.

▶ **Theorem 5.** *Let H be a fixed graph, and assume that n is even. Then w.h.p. $\tau_{\mathsf{M}} = \tau_1$.*

Theorem 6 establishes that the thresholds for minimum degree 2 and for Hamiltonicity are the same. Theorem 7 shows the hitting time version of that result.

▶ **Theorem 6.** *Let H be a fixed graph. Then*

$$\lim_{n\to\infty} \mathbf{Pr}[G(H,n,p) \text{ is Hamiltonian }] = \begin{cases} 0 & p \le p_2^-(H), \\ 1 & p \ge p_2^+(H). \end{cases}$$

Let $\tau_{\mathsf{H}} := \min\{i : \bar{G}_i(H,n) \text{ is Hamiltonian}\}$.

▶ **Theorem 7.** *Let H be a fixed graph. Then w.h.p. $\tau_{\mathsf{H}} = \tau_2$.*

Next, we describe the threshold for the appearance of a subgraph S. If S appears in a random motif graph, then S is a subgraph of some configuration of b copies of H whose union contains a vertices. For such an (a,b) covering of S, we call a subset of the covering containing b' copies of H whose union contains a' vertices an (a',b') subset. The threshold for the appearance of S depends on $\bar{\gamma}$, the maximum over all covering configurations of the minimum ratio a'/b' for all subsets of the covering configuration. Definition 15 formally describes $\bar{\gamma}$.

▶ **Theorem 8.** *Let H be a fixed graph, let S be a fixed graph, and set $v = |V(H)|$ and $\bar{\gamma} = \bar{\gamma}(S,H)$. Then*

$$\lim_{n\to\infty} \mathbf{Pr}\big[S \subseteq \bar{G}(H,n,m)\big] = \begin{cases} 0 & m \ll n^{v-\bar{\gamma}} \\ 1 & m \gg n^{v-\bar{\gamma}}. \end{cases}$$

The number of excess edges of a connected graph S, or simply its *excess*, is defined to be $\mathrm{exc}(S) = |E(S)| - |V(S)| + 1$. In particular, trees have excess 0. We say that S is *unicyclic* if its excess is 1, or *complex* if its excess is at least 2. The following theorem gives a simple description of $\bar{\gamma}$ when the motif H is a path, which allows us to deduce how the copies of H fit together to form a copy of S at the threshold when S first appears. If S is a tree, a minimal set of edge disjoint copies of H typically forms S. If S is complex, each copy of the path H typically contributes a single edge to S. If it is unicyclic, it may be formed by any edge disjoint configuration of paths H.

▶ **Theorem 9.** *Let H be a path of length $v-1$ and let S be a connected graph. Let β be the minimum number of edge-disjoint copies of H whose union contains S as a subgraph. Let $\eta = \min_{X\subseteq S} \frac{|V(X)|}{|E(X)|}$. Then*

$$\bar{\gamma} = \begin{cases} v - 1 + 1/\beta & \mathrm{exc}(S) = 0, \\ v - 1 & \mathrm{exc}(S) = 1, \\ v - 2 + \eta & \mathrm{exc}(S) \ge 2. \end{cases}$$

In the case where the motif is a long path, this result establishes a connection between the threshold for the appearance of subgraphs in random motif graphs and the threshold for the appearance of subgraphs in the trace of a random walk on the complete graph K_n (studied in [13]). Let S be a connected graph and β be the minimum number of paths in any edge-disjoint decomposition of S into paths. If H is longer than the maximum length path in such a minimum edge-disjoint path decomposition, then the threshold implied by Theorem 9 matches the threshold for the appearance of S in the trace of a random walk on the complete graph [13].

This should not come as a surprise; by noticing that when the motif is a long path, the random motif graph model approximates the trace model, in the following sense. One may sequentially "cut" the (lazy) simple random walk into chunks with buffers of length 1. We delete loops created by the trace of each chunk, and we enforce the condition that the remaining edges span a path of length ℓ (which is fixed but large). Hence the trace of each such chunk is an independent copy of a path of length ℓ. Thus we may couple the trace model and the random motif model such that the trace model will include the random motif model plus some loops plus a small number of buffer edges (which gets smaller as ℓ gets larger).

Viewing this analogy this way, we may use Theorems 8 and 9 to reprove the main theorems of [13] for the case where the base graph is complete.

2 Existence of thresholds for monotone properties

Proof of Theorem 1. Assume that $\mathcal{P}$ is a monotone increasing property and let $H_1, H_2, ..., H_{m_0(H)}$ be the copies of H that are spanned by K_n. Observe that

$$\mathsf{Pr}[G(H,n,p)\in\mathcal{P}] = \sum_{i=0}^{m_0(H)} \sum_{S\in\binom{m_0(H)}{i}} p^i(1-p)^{\binom{n}{|V(H)|}-i}\mathbb{I}\Big(\bigcup_{j\in S} H_j \in \mathcal{P}\Big)$$

is a polynomial in p. In addition, since $\mathcal{P}$ is increasing, it is increasing. Therefore we may define $p_{1/2}$ by

$$\mathsf{Pr}\big[G(H,n,p_{1/2})\in\mathcal{P}\big] = \frac{1}{2}.$$

We will show that $p_{1/2}$ is a threshold for $\mathcal{P}$. For two random graphs G, G' we write $G \subseteq G'$ if G, G' can be coupled such that G is a subgraph of G'.

First let $p = \omega(n)p_{1/2}$ where $\omega(n)\to\infty$ as $n\to\infty$ and let $k\in\mathbb{N}$. Let $G_i(H,n,p_{1/2})$ be distributed as a $G(H,n,p_{1/2})$ for $i\in[k]$. Then, by considering the probability of no appearance of a fixed copy of H, we have that the graph $\cup_{i\in[k]}G_i(H,n,p_{1/2})$ is distributed as $G(H,n,(1-(1-p_{1/2})^k))$. Thereafter $1-(1-p_{1/2})^k \le kp_{1/2}$ implies,

$$\bigcup_{i\in[k]} G_i(H,n,p_{1/2}) = G(H,n,(1-(1-p_{1/2})^k)) \subseteq G(H,n,kp_{1/2}).$$

Hence,

$$\begin{aligned}\mathsf{Pr}\big[G(H,n,\omega(n)p_{1/2})\in\mathcal{P}\big] &= 1-\mathsf{Pr}\big[G(H,n,\omega(n)p_{1/2})\notin\mathcal{P}\big]\\ &\ge \lim_{k\to\infty} 1-\mathsf{Pr}\big[G(H,n,kp_{1/2})\notin\mathcal{P}\big]\\ &\ge 1-\lim_{k\to\infty}\prod_{1=i}^{k}\mathsf{Pr}\big[G_i(H,n,p_{1/2})\notin\mathcal{P}\big]=1.\end{aligned}$$

Now assume that $p = p_{1/2}/\omega(n)$ for some $\omega(n)\to\infty$ as $n\to\infty$ and let $k\in\mathbb{N}$. Similarly to before, if we let $G_i(H,n,p_{1/2}/\omega(n))$ to be distributed as a $G(H,n,p_{1/2}/\omega(n))$ for $i\in[k]$ then, we have that

$$\begin{aligned}\bigcup_{i\in[k]} G_i(H,n,p_{1/2}/\omega(n)) &= G(H,n,(1-(1-p_{1/2}/\omega(n))^k))\\ &\subseteq G(H,n,kp_{1/2}/\omega(n)) \subseteq G(H,n,p_{1/2}).\end{aligned}$$

Hence,

$$\begin{aligned}\frac{1}{2} = \mathsf{Pr}\big[G(H,n,p_{1/2}) \in \mathcal{P}\big] &= 1 - \mathsf{Pr}\big[G(H,n,p_{1/2}) \notin \mathcal{P}\big] \\ &\geq \lim_{k\to\infty} 1 - \mathsf{Pr}\big[G(H,n,kp_{1/2}/\omega(n)) \notin \mathcal{P}\big] \\ &\geq 1 - \lim_{k\to\infty} \prod_{1=i}^{k} \mathsf{Pr}\big[G_i(H,n,p_{1/2}/\omega(n)) \notin \mathcal{P}\big] \\ &= 1 - \mathsf{Pr}\big[G_i(H,n,p_{1/2}/\omega(n)) \notin \mathcal{P}\big]^k.\end{aligned}$$

Rearranging the above gives,

$$\mathsf{Pr}\big[G_i(H,n,p_{1/2}/\omega(n)) \notin \mathcal{P}\big] \geq \lim_{k\to\infty} \left(\frac{1}{2}\right)^{1/k} = 1. \qquad \blacktriangleleft$$

3 Connectivity

Proof of Theorem 2. If $p \leq p_1^-(H)$ then by Theorem 19 the minimum degree of $G(H,n,p)$ is w.h.p. 0, hence it is not connected.

Suppose $p \geq p_1^+(H)$. In fact, for the argument below, we only assume that $p = (\ln n \pm o(\ln n))/m_1(H)$ (and the conclusion will follow by monotonicity). Let k denote the number of vertices of H. For $r = 1, \ldots, n/2$ denote by S_r the number of connected components of size r in $G(H,n,p)$. Note that for $r \geq k$, if a set of cardinality r is a connected component, then there exist $\lceil (r-1)/(k-1) \rceil$ copies of H inside the set which appear in $G(H,n,p)$, and there are no edges between it and its complement, so none of the $q = q_r(H)$ copies of H that intersect that set appear. By Lemma 17,

$$qp \sim r f_k(r/n) \cdot \ln n \geq (1 + o(1))k \ln n.$$

Let $\eta = k!/\operatorname{aut}(H)$ and suppose $r \geq k$. By Lemma 18 and by the union bound there exist constants $c, c', C > 0$ depending only on H such that

$$\begin{aligned}\mathsf{Pr}[S_r > 0] &\leq \binom{n}{r}\binom{\eta\binom{r}{k}}{\left\lceil \frac{r-1}{k-1} \right\rceil} p^{\left\lceil \frac{r-1}{k-1} \right\rceil} (1-p)^q \leq \left(\frac{en}{r}\right)^r \left(\frac{e\eta\binom{r}{k}p}{\left\lceil \frac{r-1}{k-1} \right\rceil}\right)^{\left\lceil \frac{r-1}{k-1} \right\rceil} e^{-qp} \\ &\leq \left[C \cdot \frac{n}{r} \cdot r \cdot p^{(r-1)/(r(k-1))} n^{-(1+o(1))k/r}\right]^r \\ &= \left[C \cdot \operatorname{polylog} n \cdot n^{1/r-(1+o(1))k/r}\right]^r = o(1).\end{aligned}$$

It follows that

$$\begin{aligned}\mathsf{Pr}[G(H,n,p) \text{ is not connected}] &\leq \sum_{r=1}^{n/2} \mathsf{Pr}[S_r > 0] \\ &= \mathsf{Pr}[S_1 > 0] + \sum_{r=k}^{n/2} \mathsf{Pr}[S_r > 0] = \mathsf{Pr}[S_1 > 0] + o(1),\end{aligned}$$

but according to Theorem 19 (for $p \geq p_1^+(H)$), there are no isolated vertices w.h.p., and the result follows. $\blacktriangleleft$

Note that a consequence of this proof is that for $p = (\ln n \pm o(\ln n))/m_1(H)$, with high probability, every connected component is of cardinality 1 or at least $n/2$. This means that w.h.p. there exists a unique "giant" component of linear size, and the rest of the vertices are isolated. The next lemma, whose proof uses a simple second moment argument, estimates the number of these isolated vertices for $p_- = (\ln n - \ln\ln n)/m_1(H)$.

▶ **Lemma 10.** *The number of isolated vertices in $G(H,n,p_-)$ is w.h.p. at most $2\ln n$.*

Proof. Let D_0 be the number of isolated vertices in $G(H,n,p_-)$. First,

$$\mathsf{E}[D_0] = n(1-p_-)^{m_1(H)} \sim ne^{-p_-\cdot m_1(H)} = ne^{-\ln n+\ln\ln n} = \ln n.$$

Moreover,

$$\mathsf{E}\left[D_0^2\right] = \mathsf{E}[D_0] + n(n-1)(1-p_-)^{m_2(H)}.$$

Denote $L := 2m_1(H) - m_2(H)$. Thus

$$\mathsf{E}\left[D_0^2\right] \le \mathsf{E}[D_0] + \mathsf{E}[D_0]^2(1-p_-)^{-L},$$

and since $(1-p_-)^{-L} - 1 \sim Lp_-$, we have that

$$\mathsf{Var}[D_0] \le \mathsf{E}[D_0] + \mathsf{E}[D_0]^2((1-p_-)^{-L}-1) \le \mathsf{E}[D_0] + (L+1)p_-\,\mathsf{E}[D_0]^2.$$

Thus, noting that $Lp_- = o(1)$,

$$\begin{aligned}\mathsf{Pr}[D_0 \ge 2\ln n] &= \mathsf{Pr}[|D_0 - \mathsf{E}[D_0]| \ge (1+o(1))\,\mathsf{E}[D_0]] \\ &\le (1+o(1))\left(\mathsf{E}[D_0]^{-1} + (L+1)p_-\right) = o(1).\end{aligned}$$ ◀

Proof of Theorem 3. Denote $p_\pm = (\ln n \pm \ln\ln n)/m_1(H)$ and $m_\pm = p_\pm \cdot m_n(H)$. By asymptotic equivalence of the binomial and the uniform models (see, e.g., [12]*Section 1.4) we have that w.h.p. $G(H,n,m_-)$ has a unique giant component, and the rest of the connected components are isolated vertices, whose number is at most $2\ln n$. Denote the set of these isolated vertices by V_0. Together with Theorem 2 we also conclude that w.h.p.

$$m_- \le \tau_1 \le \tau_{\mathsf{c}} \le m_+.$$

We may thus couple $\bar{G}(H,n,m_-)$, $\bar{G}(H,n,\tau_1)$, $\bar{G}(H,n,\tau_{\mathsf{c}})$ and $\bar{G}(H,n,m^+)$ such that

$$\bar{G}(H,n,m_-) \subseteq \bar{G}(H,n,\tau_1) \subseteq \bar{G}(H,n,\tau_{\mathsf{c}}) \subseteq \bar{G}(H,n,m_+),$$

by starting with $\bar{G}(H,n,m_-)$ and adding $M = m_+ - m_-$ random copies of H to create $\bar{G}(H,n,m_+)$. Note that if none of these M edges is fully contained in V_0 (and the coupling succeeds) then $\tau_1 = \tau_{\mathsf{c}}$. Thus, there exist positive constants C_1, C_2 such that,

$$\mathsf{Pr}[\tau_1 < \tau_{\mathsf{c}}] \le o(1) + M\cdot\frac{C_1\binom{|V_0|}{k}}{m_n(H)-m_+} \le o(1) + C_2\cdot\frac{m_n(H)\ln\ln n}{m_1(H)}\cdot\frac{\ln^2 n}{m_n(H)} = o(1).$$ ◀

4 Hamiltoncity and Perfect Matchings

The proof of Theorems 7 and 5 can be given in parallel, using the same techniques and tools. For clarity though, in this section we focus mainly on proving Theorem 7 and we give a sketch of the proof of Theorem 5 in the appendix.

For proving our Hamiltonicity result we use the standard technique of Posa's rotations. We define SMALL to be the vertices of significantly smaller degree than the expected one and we set LARGE to be the rest of the vertices. We first show that small to medium subsets of LARGE expand and that the vertices in SMALL are well spread. This is done in the context of Lemmas 11 and 12, 13 respectively. We use these properties of SMALL and LARGE in order to prove all the the ingredients needed to apply the Posa's rotations, which we gather in Lemma 14.

Let $p_0 := (\ln n - 2\ln\ln n)/m_1(H)$ and recall that $p_2^{\pm} = (\ln n + r_2 \ln\ln n \pm \omega(1))/m_1(H)$, $r_2 = \lfloor 2/\delta(H) - 1 \rfloor$. W.h.p. (see [9]) we can couple $G(H,n,p_0), G(H,n,p_2^-), \bar{G}(H,n,\tau_2)$ and $G(H,n,p_2^+)$ such that

(i) $G(H,n,p_0) \subset G(H,n,p_2^-) \subset \bar{G}(H,n,\tau_2) \subset G(H,n,p_2^+)$ and

(ii) there are $(1+o(1))(p_2^- - p_0)\frac{r!}{\mathrm{aut}(H)}\binom{n}{r} > n\ln\ln n/2r$ copies of H in $G(H,n,p_2^-)$, hence in $\bar{G}(H,n,\tau_2)$, that are not present in $G(H,n,p_0)$.

Observe that the above coupling and Theorem 7 imply Theorem 6. In addition a similar coupling and Theorem 5 imply Theorem 4.

We now define the sets SMALL, LARGE based on the degrees of the vertices in $G(H,n,p_0)$. Let LARGE $= \{v \in V : v$ intersects at least $\ln\ln n$ copies of H in $G(H,n,p_0)\}$ and SMALL $= V \setminus$ LARGE.

▶ **Lemma 11.** *W.h.p. every $S \subset$ LARGE of size at most $n/30r$ satisfies $|N(S)| \geq 10|S|$.*

▶ **Lemma 12.** *W.h.p. for every pair $u, v \in$ SMALL there do not exist $\ell \leq 6$ copies of H in $G(H,n,p_2^+)$ that span a connected subgraph containing both u, v. Hence w.h.p. every pair $u, v \in$ SMALL is at distance at least 7 in $G(H,n,p_2^+)$.*

▶ **Lemma 13.** *W.h.p. for every $v \in V$ there exists at most one copy of H in $G(H,n,p_2^+)$, hence in $\bar{G}(H,n,\tau_2)$, that intersect both $\{v\}$ and SMALL $\setminus \{v\}$.*

Now we generate $\bar{G}(H,n,\tau_2)$ as follows. We first generate $G_0' = G(H,n,p_0)$. Then we randomly permute the copies of H not appearing in G_0', let them be $H_1, H_2, ...$ We also let $S_0 = \emptyset$. We define the sequences $G_0', G_1', ...$ and $S_0, S_1, ...$ in the following way. At step $i \in \mathbb{N}$ we query H_i whether it is incident to a vertex in SMALL. If it is then we set $S_i = S_{i-1}$ and $G_i' = G_{i-1}' \cup H_i$. Otherwise we set $S_i = S_{i-1} \cup \{H_i\}$ and $G_i' = G_{i-1}'$. Let $t^* = \min\{i : \delta(G_i') = 2\}$ and $S_{t^*} = \{H_{i_1}, H_{i_2}, ..., H_{i_w}\}$.

Given the sequence $G_0', G_1', ..., G_{t^*}'$ and the set $S_{t^*} = \{H_{i_1}, H_{i_2}, ..., H_{i_w}\}$ we define the graph sequence $F_0, ..., F_w$ by $F_0 = G_{t^*}'$ and $F_j = F_{j-1} \cup H_{i_j}$ for $1 \leq j \leq w$. Observe that S_{t^*} consists of all copies of H in $\{H_1, ..., H_{t^*}\}$ that have not been added to G_0', equivalently the copies of H that are not incident to SMALL. Thus $F_w = G_{t^*}' \cup \big(\bigcup_{j=1}^{w} H_{i_j}\big) = G_0' \cup \big(\bigcup_{i=1}^{t^*} H_i\big) = \bar{G}(H,n,\tau_2)$.

▶ **Lemma 14.** *W.h.p. the following hold:*

- **i)** *$w \geq n\ln\ln n/2r - n$,*
- **ii)** *every $S \subset V$ of size at most $n/30r$ satisfies $|N(S)| \geq 2|S|$ in F_0,*
- **iii)** *F_0 is connected,*
- **iv)** *for every $1 \leq j \leq w$, $\epsilon > 0$, and every set Q_j consisting of ϵn^2 edges not present in F_j there exist a constant $C_\epsilon > 0$ such that the probability that Q_j intersects $E(H_{i_{j+1}})$ is at least C_ϵ.*

We are now ready to apply Posa's rotations . For that assume that F_j is not Hamiltonian and consider a longest path in F_j, P_j, $j \geq 0$. Let x, y be the end-vertices of P_j. Given yv where v is an interior vertex of P_j we can obtain a new longest path $P_j' = x..vy..w$ where w is the neighbor of v on P_j between v and y. In such a case we say that P_j' is obtained from P_j by a rotation with the end-vertex x being the fixed end-vertex.

Let $\textsc{End}_j(x;P_j)$ be the set of end-vertices of longest paths of F_j that can be obtained from P_j by a sequence of rotations that keep x as the fixed end-vertex. Thereafter for $z \in \textsc{End}_j(x;P_j)$ let $P_j(x,z)$ be a path that has end-vertices x,z and can be obtain form P_j by a sequence of rotations that keep x as the fixed end-vertex. Observe that for $z \in \textsc{End}_j(x;P_j)$ and $z' \in \textsc{End}_j(z;P_j(x,z))$ there exists a z-z' path $P_{z,z'}$ of length $|P_j|$ that can be obtained from P_j via a sequence of Posa rotations. Thus we can conclude that $\{z,z'\}$ does not belong to F_j. Indeed assume that $\{z,z'\} \in E(G_i)$. Then we can close $P_{z,z'}$ into a cycle $C_{z,z'}$ that is not Hamiltonian. Since F_j is connected there is an edge e spanned by $V(C_{z,z'}) \times V \setminus V(C_{z,z'})$. $E(C_{z,z'}) \cup \{e\}$ spans a path of length $|P_j|+2$ contradicting the maximality of P_j. Similarly if $\{z,z'\} \in E(H_{i_{j+1}})$ then F_{j+1} is either Hamiltonian or it contains a path that is longer than P_j. At the same time it follows (see [9]*Corollary 6.7) that

$$|N(\textsc{End}(x,P_j))| < 2|\textsc{End}(x,P_j)|.$$

Moreover for every $z \in \textsc{End}_j(x;P_j)$

$$|N(\textsc{End}(z,P_j(x,z)))| < 2|\textsc{End}(z,P_j(x,z))|.$$

As a consequence of Lemma 11, we have that $|\textsc{End}(x,P_j)| \geq n/30r$ and $|\textsc{End}(z,P_j(x,z))| \geq n/30r$ for every $z \in \textsc{End}_j(x;P_j)$. Let $E_j = \{\{z,z'\} : z \in \textsc{End}_j(x;P_j) \text{ and } z' \in \textsc{End}_j(z;P_j(x,z))\}$. Then $|E_j| \geq (n/30r)^2/2$.

Now let Y_j be the indicator of the event $\{E_j \cap E(H_{i_{j+1}}) \neq \emptyset\}$ and set $Z = \sum_{j=1}^{w} Y_i$. From Lemma 14 iv) we have $\mathsf{Pr}[Y_j = 1] \geq C_\epsilon$ (here $\epsilon = 1/2(30r)^2$). In the event that G_w is not Hamiltonian, $Z \leq n$ while Y_j is a $\mathsf{Bernoulli}(C_\epsilon)$ random variable for $1 \leq j \leq w$. Since $w \geq n \ln\ln n/2r - n$ we have $\mathsf{Pr}[\mathsf{Bin}(w,C_\epsilon) \leq n] = o(1)$. Hence w.h.p. $F_w = \bar{G}(H,n,\tau_2)$ is Hamiltonian and the hitting time for Hamiltonicity equals the hitting time for minimum degree 2.

5 Subgraph appearance

In $G(n,p)$ there is only one way for a specified subgraph to appear on a fixed set of vertices: all the edges in the subgraph must be present. In the case of random motif graphs, there are multiple ways to place motifs so that a specified subgraph appears on a fixed set of vertices. For example, in a random two-path graph, a triangle may appear on $\{1,2,3\}$ if (i) the paths $(1,2,3)$ and $(3,1,z)$ are present or (ii) the paths $(1,2,x)$, $(2,3,y)$ and $(3,1,z)$ are present. In order to pin down the threshold for subgraph appearance, it is necessary to understand the various motif configurations that cause the subgraph to appear and their relative probabilities. The following definition provides the notation to describe such configurations.

▶ **Definition 15.** *Let V be a set of vertices. Let S be a fixed graph on a subset of the vertices of V. Let $H_1, H_2, \dots H_b$ be copies H also defined on subsets of vertices of V.*

(a) *We say $\{H_1, H_2, \dots H_b\}$ is an (a,b) covering of S if (i) $S \subseteq \bigcup_{j=1}^{b} H_j$, (ii) $|V(\bigcup_{j=1}^{b} H_j)| = a$, and (iii) for each $\ell \in [b]$, $S \not\subseteq \bigcup_{j=1}^{b} H_j \setminus H_\ell$.*

(b) *Let $k(a,b)$ be the number of unique configurations of (a,b) coverings, i.e. the number of ways to place b copies of H on a vertices such that conditions (i)-(iii) of (a) hold. Enumerate the possible configurations of (a,b) coverings with values in $[k(a,b)]$. For $i \in [k(a,b)]$, an (a,b,i) covering of S is an (a,b) covering with configuration i.*

(c) *We say the set $\{F_1, F_2, \dots F_{b'}\}$ (with precisely b' elements) is an (a',b') subset of an (a,b,i) covering $\{H_1, H_2, \dots H_b\}$ if (i) $\{F_1, F_2, \dots F_{b'}\} \subseteq \{H_1, H_2, \dots H_b\}$, and (ii) $|V(\bigcup_{\ell=1}^{b'} F_\ell)| = a'$.*

(d) *Let* $\mathcal{I}(S,H) = \{(a,b,i) \mid$ *there exists an* (a,b) *covering of* S *by* H *and* $i \in [k(a,b)]\}$.

(e) *For* $(a,b,i) \in \mathcal{I}(S,H)$, *let*

$$\mathcal{D}(a,b,i) = \{(a',b') \mid \text{there exists an } (a',b') \text{ subset of the } (a,b,i) \text{ covering}\}.$$

(f) *For* $(a,b,i) \in \mathcal{I}(S,H)$, *let* $\gamma(a,b,i) = \min_{(a',b') \in \mathcal{D}(a,b,i)} \frac{a'}{b'}$ *and denote*

$$\bar{\gamma} = \max_{(a,b,i) \in \mathcal{I}(S,H)} \gamma(a,b,i).$$

Proof of Theorem 8. Let $G \sim \bar{G}(H,n,m)$. We say that an instance of the subgraph S in G is an (a,b,i) instance if the placed graphs $H_1, \dots H_b$ that contribute at least one edge to S form an (a,b,i) covering of S. Let X_i^{ab} denote the number of (a,b,i) instances of S in G. Let $Z = \sum_{(a,b,i) \in \mathcal{I}(S,H)} X_i^{ab}$ be the total number of instances of the subgraph S in G.

First we use the first moment method to show that if $m \ll n^{v-\bar{\gamma}}$, then the probability that S occurs as a subgraph is $o(1)$. It suffices to show that for all $(a,b,i) \in \mathcal{I}(S,H)$, $\mathsf{E}\left[X_i^{ab}\right] = o(1)$ since

$$\mathsf{Pr}[Z > 0] \leq \mathsf{E}[Z] = \sum_{(a,b,i) \in \mathcal{I}(S,H)} X_i^{ab},$$

and $|\mathcal{I}(S,H)|$ is a constant independent of n.

We now compute $\mathsf{E}\left[X_i^{ab}\right]$ for a fixed triple $(a,b,i) \in \mathcal{I}(S,H)$. Let $\{F_1, \dots F_{b'}\}$ be an (a',b') subset of the configuration (a,b,i) with $a'/b' = \gamma(a,b,i)$. Let Y be the number of instances of $F = \bigcup_{i=1}^{b'} F_{b'}$ in G formed by the configuration $\{F_1, \dots F_{b'}\}$. Since an (a,b,i) instance of S contains an instance of the configuration $\{F_1, \dots F_{b'}\}$, $X_i^{ab} \leq Y$. The number of ways to select a' vertices is at most $n^{a'}$. The probability that a labeled copy of H is placed on a specified set of vertices is m/n^v. We compute

$$\mathsf{E}\left[X_i^{ab}\right] \leq \mathsf{E}[Y] \leq cn^{a'} \left(\frac{m}{n^v}\right)^{b'} = c\left(n^{\gamma(a,b,i)-v} m\right)^{b'} \leq c\left(n^{\bar{\gamma}-v} m\right)^{b'},$$

where c is a constant depending only on the number of automorphisms of S and the number of automorphisms of the configuration $\{F_1, \dots F_{b'}\}$. It follows that for $m \ll n^{\bar{\gamma}-v}$, $\mathsf{E}\left[X_i^{ab}\right] = o(1)$, as desired.

Next we use the second moment method to show that if $m \gg n^{v-\bar{\gamma}}$ then S appears as a subgraph almost surely. It suffices to show that there exists some $(a,b,i) \in \mathcal{I}(S,H)$ such that X_i^{ab} is almost surely positive. Let (a,b,i) be such that $\bar{\gamma} = \gamma(a,b,i)$. We apply Corollary 4.3.5 of [1] to show that X_i^{ab} is almost surely positive. Let $X_i^{ab} = \sum_j A_j$ where A_j is an indicator random variable for the event that there is an (a,b,i) instance of S formed by a configuration of $H_1, H_2, \dots H_b$ each present on a specified set of vertices. Fix A_ℓ, and let

$$\Delta^* = \sum_{j \sim \ell} \mathsf{Pr}[A_j \mid A_\ell],$$

where $j \sim \ell$ indicates that A_j and A_ℓ are not independent. By 4.3.5 of [1], if $\mathsf{E}\left[X_i^{ab}\right] \to \infty$ and $\Delta^* = o(\mathsf{E}\left[X_i^{ab}\right])$, then $X_i^{ab} > 0$ almost surely.

First we show that $\mathsf{E}\left[X_i^{ab}\right] \to \infty$. We compute as above

$$\mathsf{E}\left[X_i^{ab}\right] \geq c' n^a \left(\frac{m}{n^v}\right)^b = c'\left(n^{a/b-v} m\right)^b \geq c'\left(n^{\bar{\gamma}-v} m\right)^b$$

where c' is a constant depending only on the number of automorphisms of S and the number of automorphisms of the configuration $\{H_1, \dots H_b\}$. It follows that if $m \gg n^{v-\bar{\gamma}}$ then $\mathsf{E}\left[X_i^{ab}\right] \to \infty$.

Finally, we show $\Delta^* = o(\mathsf{E}\left[X_i^{ab}\right])$. Observe that under the assumption $m \gg n^{v-\bar{\gamma}}$,

$$\Delta^* = \sum_{(a',b')\in\mathcal{D}(a,b,i)} cn^{a-a'}\left(\frac{m}{n^v}\right)^{b-b'} = \sum_{(a',b')\in\mathcal{D}(a,b,i)} c\,\mathsf{E}\left[X_i^{ab}\right]\left(n^{-a'/b'+v}m^{-1}\right)^{b'}$$
$$\leq c'\,\mathsf{E}\left[X_i^{ab}\right]\left(n^{-\gamma(a,b,i)+v}m^{-1}\right)^{b} = c'\,\mathsf{E}\left[X_i^{ab}\right]\left(n^{v-\bar{\gamma}}m^{-1}\right)^{b} = o\left(\mathsf{E}\left[X_i^{ab}\right]\right). \quad \blacktriangleleft$$

6 Conclusion

6.1 The value of the random motif model

The study of random motif graphs has the potential to strengthen the impact of the Erdős-Rényi construction. In the context of analyzing real-world networks with an overrepresented motif, random motif graphs may be a more insightful null hypothesis model to compare against to identify non-random structure. For instance by studying subgraphs counts of random H motif graphs one can determine if some larger motif pattern is a byproduct of having many copies of H or is itself some novel aspect of the network structure. Moreover, it is possible that a random motif graph may be used to establish the existence of a graph with some extremal property of interest. Finally, random motif graphs can be used as an alternate definition of average case for analyzing algorithms under the assumption that the input has some motif structure.

6.2 Future directions: understanding threshold behavior more broadly

We have established that random motif graphs behave similarly to traditional Erdős-Rényi random graphs with respect to thresholds and hitting times for monotone properties. Does similar behavior appear when we consider random graphs formed by randomly adding primitive subgraphs H whose size scales with n, the number of vertices of the random graph? Instead of taking H to be a fixed motif, H could be a path, cycle, matching or clique whose size depends on n, for example. Some of these cases were in fact studied in several contexts. For example, the union of $d \geq 3$ random perfect matchings is contiguous to the random d-regular graph, and is sometimes easier to analyze [19]. Moreover, we can consider the class of models where H itself is chosen from some probability distribution. In several cases, this has been studied as well. For instance, [10] and [8] consider the case when H is the uniform spanning tree, and [15] considers the case when H is an Erdős-Rényi random graph with constant density and size dependent on n. Further study of these models is a first step toward delineating a larger family of random graphs that exhibit Erdős-Rényi like threshold and hitting time behaviors.

References

1 Noga Alon and Joel H. Spencer. *The probabilistic method*. Wiley Series in Discrete Mathematics and Optimization. John Wiley & Sons, Inc., Hoboken, NJ, fourth edition, 2016.

2 Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461, 2007.

3 Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999. doi:10.1126/science.286.5439.509.

4 Béla Bollobás. *Random graphs*. Academic Press, Inc. [Harcourt Brace Jovanovich, Publishers], London, 1985.

5 Béla Bollobás and Andrew Thomason. Hereditary and monotone properties of graphs. In *The mathematics of Paul Erdős, II*, volume 14 of *Algorithms Combin.*, pages 70–78. Springer, Berlin, 1997. doi:10.1007/978-3-642-60406-5_7.

6 Paul Erdős and Alfréd Rényi. On random graphs. I. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

7 Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Magyar Tud. Akad. Mat. Kutató Int. Közl.*, 5:17–61, 1960.

8 Alan Frieze, Navin Goyal, Luis Rademacher, and Santosh Vempala. Expanders via random spanning trees. *SIAM Journal on Computing*, 43(2):497–513, 2014. doi:10.1137/120890971.

9 Alan Frieze and Michał Karoński. *Introduction to random graphs*. Cambridge University Press, Cambridge, 2016. doi:10.1017/CBO9781316339831.

10 Alan Frieze, Michał Karoński, and Luboš Thoma. On perfect matchings and Hamilton cycles in sums of random trees. *SIAM Journal on Discrete Mathematics*, 12(2):208–216, 1999. doi:10.1137/S0895480196313790.

11 Edgar N. Gilbert. Random graphs. *Annals of Mathematical Statistics*, 30:1141–1144, 1959. doi:10.1214/aoms/1177706098.

12 Svante Janson, Tomasz Łuczak, and Andrzej Rucinski. *Random graphs*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience, New York, 2000. doi:10.1002/9781118032718.

13 Michael Krivelevich and Peleg Michaeli. Small subgraphs in the trace of a random walk. *Electronic Journal of Combinatorics*, 24(1):Paper 1.28, 22, 2017.

14 Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri B. Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

15 Samantha Petti and Santosh S. Vempala. Approximating Sparse Graphs: The Random Overlapping Communities Model. *arXiv e-prints*, February 2018. arXiv:1802.03652.

16 Daniel Poole. On the strength of connectedness of a random hypergraph. *Electronic Journal of Combinatorics*, 22(1):Paper 1.69, 16, 2015.

17 Sen Song, Per Jesper Sjöström, Markus Reigl, Sacha B. Nelson, and Dmitri B. Chklovskii. Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS biology*, 3(3):e68, 2005.

18 Duncan J. Watts and Steven H. Strogatz. Collective dynamics of “small-world" networks. *Nature*, 393(6684):440–442, 1998. doi:10.1038/30918.

19 Nicholas C. Wormald. Models of random regular graphs. In *Surveys in combinatorics, 1999 (Canterbury)*, volume 267 of *London Math. Soc. Lecture Note Ser.*, pages 239–298. Cambridge Univ. Press, Cambridge, 1999.

20 Esti Yeger-Lotem, Shmuel Sattath, Nadav Kashtan, Shalev Itzkovitz, Ron Milo, Ron Y. Pinter, Uri Alon, and Hanah Margalit. Network motifs in integrated cellular networks of transcription-regulation and protein-protein interaction. *Proceedings of the National Academy of Sciences of the United States of America*, 101(16):5934–5939, 2004.

A Estimates for useful functions

▶ **Lemma 16.** *For $r = r(n)$, if $k = |V(H)|$ and $\alpha = r/n$ then $m_r(H) \sim rm_1(H) \cdot \frac{1-(1-\alpha)^k}{k\alpha}$.*

Proof. Observe that for $r \geq 0$,

$$m_r(H) = \left(\binom{n}{k} - \binom{n-r}{k}\right) \cdot \frac{k!}{\mathrm{aut}(H)},$$

thus

$$\frac{m_r(H)}{rm_1(H)} = \frac{\binom{n}{k} - \binom{n-r}{k}}{r\left(\binom{n}{k} - \binom{n-1}{k}\right)} \sim \frac{n^k - (n-r)^k}{r(n^k - (n-1)^k)} = \frac{1-(1-\alpha)^k}{r(1-(1-n^{-1})^k)} \sim \frac{1-(1-\alpha)^k}{k\alpha}. \quad \blacktriangleleft$$

For $r \geq 1$, denote by $q_r(H)$ the number of copies of H that intersect $[r]$ but that are not contained in $[r]$.

▶ **Lemma 17.** *For $r = r(n)$, if $k = |V(H)|$ and $\alpha = r/n$ then $q_r(H) \sim rm_1(H) \cdot \frac{1-(1-\alpha)^k-\alpha^k}{k\alpha}$.*

Proof. Observe that for $r \geq 0$,

$$q_r(H) = \left(\binom{n}{k} - \binom{n-r}{k} - \binom{r}{k}\right) \cdot \frac{k!}{\mathrm{aut}(H)},$$

thus

$$\frac{q_r(H)}{rm_1(H)} = \frac{\binom{n}{k} - \binom{n-r}{k} - \binom{r}{k}}{r\left(\binom{n}{k} - \binom{n-1}{k}\right)} \sim \frac{n^k - (n-r)^k - r^k}{r(n^k - (n-1)^k)}$$
$$= \frac{1-(1-\alpha)^k-\alpha^k}{r(1-(1-n^{-1})^k)} \sim \frac{1-(1-\alpha)^k-\alpha^k}{k\alpha}. \quad \blacktriangleleft$$

For convenience we define for $\alpha \in [0,1]$ and $k \geq 1$,

$$f_k(\alpha) = \frac{1-(1-\alpha)^k-\alpha^k}{k\alpha}.$$

▶ **Lemma 18.** *For $2 \leq k \leq r$ we have that $rf_k(r/n) \geq (1+o(1))k$.*

Proof. Write $g_k(\alpha) = f_k(\alpha) \cdot k\alpha = 1-(1-\alpha)^k-\alpha^k$. Observe that it is strictly increasing in $(0, 1/2)$. Note also that

$$n \cdot g_k\left(\frac{k}{n}\right) = n - ne^{-k^2/n} - o(1) \sim k^2.$$

It follows that

$$\frac{kr}{n} \cdot f_k\left(\frac{r}{n}\right) = g_k\left(\frac{r}{n}\right) \geq g_k\left(\frac{k}{n}\right) \sim \frac{k^2}{n},$$

so $rf_k(r/n) \geq (1+o(1))k$. $\blacktriangleleft$

B Minimum degree

▶ **Theorem 19.** *With high probability*

$$\delta(G(H,n,p_d^-)) < d \qquad \text{and} \qquad \delta(G(H,n,p_d^+)) \geq d.$$

Proof. Let $\delta = \delta(H)$. It suffices to show that with high probability for $\ell \in \mathbb{Z}_{\geq 0}$

$$\mathsf{Pr}\left[\delta(G(H,n,p_{\ell\cdot\delta}^-)) > (\ell-1)\delta\right] = o(1) \tag{1}$$

and

$$\mathsf{Pr}\left[\delta(G(H,n,p_{\ell\cdot\delta}^+)) < \ell\delta\right] = o(1). \tag{2}$$

Proof of (1): Let $p = p_{\ell\cdot\delta}^-$. For $v \in V$ let $I_v = \mathbb{I}\{d(v) = (\ell-1)\delta\}$ and $Z = \sum_{v\in V} I_v$.

$$\mathsf{E}[Z] \geq (1-o(1))n\binom{n-1}{v_H-1}^{\ell-1} p^{\ell-1}(1-p)^{m_1(H)-\ell+1}$$
$$\geq C_1 n(pn^{(v_H-1)})^{\ell-1} e^{-(p+4p^2)(m_1(H)-\ell+1)}$$
$$\geq C_2 n(\log n)^{\ell-1} e^{-\log n-(\ell-1)\log\log n+\omega(1)} \geq e^{\omega(1)/2}.$$

In addition,

$$\begin{aligned}
\mathsf{E}\left[Z^2\right] &= \sum_{u,v\in V} \mathsf{Pr}[I_v \wedge I_u] \\
&\leq \mathsf{E}[Z]^2 + \sum_{u\neq v\in V} \mathsf{Pr}[I_u \wedge I_v \wedge \{u,v \text{ lie on the same copy of } H\}] \\
&\leq \mathsf{E}[Z]^2 + \binom{n}{2}\binom{n-2}{r-2}\frac{r!}{\mathrm{aut}(H)}p(1-p)^{(1-o(1))2m_1} \\
&= \mathsf{E}[Z]^2 + nm_1p(1-p)^{m_1-1}C_3(1-p_2^-)^{(1-o(1))m_1} = \mathsf{E}[Z]^2 + o(1)\,\mathsf{E}[Z] \\
&= (1+o(1))\,\mathsf{E}[Z]^2.
\end{aligned}$$

Chebyshev's inequality give us,

$$\mathsf{Pr}[|Z - \mathsf{E}[Z]| \geq \mathsf{E}[Z]/2] \leq \frac{\mathsf{E}\left[Z^2\right] - \mathsf{E}[Z]^2}{0.25\,\mathsf{E}[Z]^2} = o(1).$$

Hence with high probability there exist vertices of degree $(\ell-1)\delta$.

Proof of (2): Let $p = p^+_{\ell\cdot\delta}$. Let $\mathcal{E}_1$ be the event that in $G(H,n,p)$ there exists a vertex of degree $d \leq \ell\delta$ that lies on more than ℓ copies of H. In the event $\mathcal{E}_1$ there exists a vertex v and a vertex set S of size d such that all the neighbors of v lie in S and at least $\ell+1$ copies of H intersect $S\cup\{v\}$, each in at least $\delta+1$ vertices. Therefore,

$$\begin{aligned}
\mathsf{Pr}[\mathcal{E}_1] &\leq n\binom{n}{d}[1-p]^{\binom{n-d-1}{v_H-1}}\left(\binom{d+1}{\delta+1}\binom{n-\delta-1}{v_H-\delta-1}\right)^{\ell+1}p^{\ell+1} \\
&\leq e^{-p\cdot\binom{n-d-1}{v_H-1}}n^{d+1-\delta(\ell+1)}[n^{v_H-1}p]^{\ell+1} \\
&\leq e^{-(1+o(1))p\cdot m_1(H)}(\log^2 n)^{\delta_d(H)+1} = o(1).
\end{aligned}$$

In the event $\neg\mathcal{E}_1$ the number of vertices of degree less than $\ell\delta$ is bounded by the number of vertices that are covered by at most $\ell-1$ copies of H. Thus

$$\begin{aligned}
\mathsf{Pr}\left[\delta(G(H,n,p^+_{\ell\cdot\delta})) < \ell\delta\right] &\leq \mathsf{Pr}[\mathcal{E}_1] + n\sum_{i=0}^{\ell-1}\binom{m_1(H)}{i}p^i(1-p)^{m_1(H)-i} \\
&\leq \ell n(m_1(H)p)^{\ell-1}e^{-pm_1(H)+p\ell} + o(1) \\
&\leq \ell n[2\log n]^{\ell-1}e^{-\log n-(\ell-1)\log\log n-\omega(1)} + o(1) = o(1).
\end{aligned}$$

◀

C Proofs of lemmas for Hamiltoncity

Proof of Lemma 11. If there exists $S \subset$ LARGE of size $n^{19/20} \leq |S| \leq n/30r$ such that $|N(S)| < 10|S|$ then there exist sets A, B of size $n^{19/20} \leq s \leq n/30r$ and $n-11s$ respectively such that no copy of H, H' satisfies $|A\cap H'| = 1$ and $|B\cap H'| = r-1$ (take $S = A$ and B to be any subset of $V\setminus(S\cup N(S))$ of size $n-11s$). The probability of such event occurring is bounded above by

$$\begin{aligned}
&\sum_{s=n^{19/20}}^{n/30r}\binom{n}{s}\binom{n-s}{10s}(1-p_0)^{\frac{r!}{\mathrm{aut}(H)}\cdot s\binom{n-11s}{r-1}} \\
&\leq \sum_{s=n^{19/20}}^{n/30r}\left[\frac{en}{s}\cdot\left(\frac{en}{10s}\right)^{10}e^{-p_0\frac{r!}{\mathrm{aut}(H)}\cdot\binom{n-11s}{r-1}}\right]^s
\end{aligned}$$

$$\begin{aligned}\dots &\leq \sum_{s=n^{19/20}}^{n/30r}\left[\left(\frac{n}{s}\right)^{11} e^{-\frac{\ln n-2\ln\ln n}{\binom{n-1}{r-1}}\cdot\binom{n-11s}{r-1}}\right]^s\\ &\leq \sum_{s=n^{19/20}}^{n/30r}\left[\left(\frac{n}{s}\right)^{11}\left(\frac{\ln^2 n}{n}\right)^{(1-\frac{11s}{n})\cdots(1-\frac{11s-r+2}{n-r+2})}\right]^s\\ &\leq \sum_{s=n^{19/20}}^{n/30r}\left[\left(\frac{n}{s}\right)^{11}\left(\frac{\ln^2 n}{n}\right)^{1-\frac{12sr}{n}}\right]^s \leq \sum_{s=n^{19/20}}^{n/30r}\left[n^{11/20}\left(\frac{\ln^2 n}{n}\right)^{18/30}\right]^s = o(1).\end{aligned}$$

Now assume that there exists a set $S \subset \textsc{Large}$ of size at most $n^{19/20}$ that satisfies $|N(S)| < 10|S|$. Since every vertex in S is in at least $\ln\ln n$ copies of H and every copy of H covers r vertices we have that S intersects at least $|S|\ln\ln n/11$ copies of H. Each of those copies is spanned by $S \cup N(S)$. Therefore there exists a set $W \supseteq S \cup N(S)$ of size $w = |W| = 11|S| \leq 11n^{19/20}$ that intersects at least $\frac{|W|\ln\ln n}{11r}$ copies of H each, in at least 2 vertices. Since every vertex in $\textsc{Large}$ has $\ln\ln n$ neighbors $|W| \geq \ln\ln n$. The probability that such a set exists is bounded by

$$\begin{aligned}&\sum_{w=\ln\ln n}^{11n^{19/20}}\binom{n}{w}\binom{r!\binom{w}{2}\binom{n}{r-2}}{w\ln\ln n/11r}p_0^{w\ln\ln n/11r}\\ &\leq \sum_{w=\ln\ln n}^{11n^{19/20}} n^w\left(\frac{11er^3wn^{r-2}}{\ln\ln n}\right)^{w\ln\ln n/11r}p_0^{w\ln\ln n/11r}\\ &\leq \sum_{w=\ln\ln n}^{11n^{19/20}}\left[n^{11r/\ln\ln n}\cdot\frac{11er^3wn^{r-2}}{\ln\ln n}\cdot p_0\right]^{w\ln\ln n/11r}\\ &\leq \sum_{w=\ln\ln n}^{11n^{19/20}}\left(n^{11r/\ln\ln n}\cdot\frac{w\log n}{n}\right)^{w\ln\ln n/11r} = o(1).\end{aligned}$$

◀

Proof of Lemma 12. For $u \in V$ and $Q \subset V$ let $S(u,Q)$ be the event that in $G(H,n,p_0)$ u intersects at most $\ln\ln n$ copies of H that do not intersect Q. For $0 \leq |Q| \leq 6$,

$$\mathsf{Pr}[S(u,Q)] \leq \mathsf{Pr}\left[\mathsf{Bin}\left(\frac{r!}{\mathrm{aut}(H)}\binom{n-7}{r-1},p_0\right) \leq \ln\ln n\right] \leq n^{-0.9}.$$

Let $\mathcal{B}$ be the event that for some $u,v \in \textsc{Small}$ there exist $\ell \leq 6$ copies of H in $G(H,n,p_2^+)$ that span a connected subgraph containing both u,v. If $\mathcal{B}$ occurs then we can find a set $Q = \{v = v_0, v_1, \dots, v_{\ell-1}, v_\ell = u\}$ such that i) the events $S(v, Q\setminus\{v\})$, $S(u, Q\setminus\{u\})$ occur and ii) there exist $H_1,\dots,H_\ell$ in $G(H,n,p_2^+)$ such that $H_i \cap Q = \{v_{i-1}, v_i\}$. Since all the aforementioned events are independent

$$\begin{aligned}\mathsf{Pr}[\mathcal{B}] &\leq \sum_{\ell=1}^{6}\sum_{Q=\{v_0,v_1,\dots,v_\ell\}}\mathsf{Pr}[S(v_0,Q\setminus\{v_0\}]\cdot\left(\binom{n-2}{r-2}\frac{r!}{\mathrm{aut}(H)}p_2^+\right)^\ell\cdot\mathsf{Pr}[S(v_\ell,Q\setminus\{v_\ell\}]\\ &\leq \sum_{\ell=1}^{6} n^{\ell+1}\cdot n^{-0.9}\cdot\left(\frac{C_3\ln n}{n}\right)^\ell\cdot n^{-0.9} = o(1).\end{aligned}$$

◀

Proof of Lemma 13. Lemma 12 implies that w.h.p. there do not exist $v \in V$ and $u, w \in \textsc{Small}$, $u \neq w$ such that in $G(H, n, p_2^+)$ v and u are in a copy of H and v and w are in a copy of H. The probability that there exist $v \in V$, $u \in \textsc{Small} \setminus \{v\}$ that are both contained in more than one copy of H in $G(H, n, p_2^+)$ is bounded by

$$\sum_{v,u \in V} \mathsf{Pr}[S(u, \{v\})] \left(\binom{n-2}{r-2} \frac{r!}{\mathrm{aut}(H)} p_2^+ \right)^2 \leq C_4 n^{-0.9} \log^2 n = o(1). \qquad \blacktriangleleft$$

Proof of Lemma 14.

1. Recall that we can couple $G(H, n, p_0), \bar{G}(H, n, \tau_2)$ such that $G(H, n, p_0) \subset \bar{G}(H, n, \tau_2)$ w.h.p. and there are at least $n \ln \ln n / 2r$ copies of H in $\bar{G}(H, n, \tau_2)$ that are not present in $G(H, n, p_0)$. From Lemma 13 it follows that w.h.p. each of those copies that spans a vertex in $\textsc{Small}$ also spans a unique vertex in $V \setminus \textsc{Small}$. Hence $w \geq n \ln \ln n / 2r - n$.
2. Let $S \subset V$, $|S| \leq n/30r$ and set $S_s = S \cap \textsc{Small}$, $S_L = S \cap \textsc{Large}$. Lemma 11 implies that $|N(S_L)| \geq 10|S_L|$. In the case $|S_L| \geq |S_s|$ we have

   $$|N(S)| \geq |N(S_L) \setminus S_s| \geq 10|S_L| - |N(S_L) \cap S_s| \geq 10|S_L| - |S_s| \geq 9|S_L| \geq 2|S|.$$

   Next assume $|S_L| < |S_s|$. Lemma 12 implies that no two vertices in $\textsc{Small}$ are within distance 2 in $G(H, n, p_2^+)$, hence their neighborhoods are disjoint. Also F_0 has minimum degree 2. Therefore $|N(S_s)| \geq 2|S_s|$. Now let $S_L = S_1 \cup S_2$ where S_2 consists of all the vertices in S_L that are within distance 2 from S_s and $S_1 = S_L \setminus S_2$. If $|S_1| \geq |S_2|$ then since S_s and S_1 have disjoint neighborhoods we have that

   $$|N(S)| \geq |N(S_s) \setminus S_2| + |N(S_1) \setminus S_2| \geq 2|S_s| + 10|S_1| - 2|S_2| \geq 2|S|.$$

   Otherwise $|S_s| > |S_L|$ and $|S_2| > |S_1|$. For $v \in S_s$ let $N_{S_2}(v)$ be the set of vertices in S_2 that are within distance 2 from v, hence $\cup_{v \in S_s} N_{S_2}(v) = |S_2|$. Lemma 12 states that no two vertices in $\textsc{Small}$ are within distance 6, thus for $v, u \in S_s, v \neq u$ the sets $N(N_{S_2}(v)), N(N_{S_2}(u))$ are disjoint. In addition since $N_{S_2} \subset S_L$ and $|S_L| \leq |S| \leq n/30r$, Lemma 11 implies that $|N(N_{S_2}(v))| \geq 10|N_{S_2}(v)|$ for all $v \in S_s$. Thus

   $$\begin{aligned} |N(S)| &\geq \sum_{v \in S_s} |N(N_{S_2}(v) \cup \{v\})| \\ &\geq \sum_{v \in S_s} [10|N_{S_2}(v)| - |\{v\}|] \cdot \mathbb{I}_{N_{S_2}(v) \neq \emptyset} + |N(v)| \mathbb{I}_{N_{S_2}(v) = \emptyset} \\ &\geq \sum_{v \in S_s} 2 = 2|S_s| \geq |S|. \end{aligned}$$

3. Assume that there exists a set $S \subset V$ such that S is a connected component of F_0 and let $s = |S|$. F_0 has minimum degree 2 therefore $s \geq 3$. Let $S_L = S \cap \textsc{Large}$ and $S_s = S \cap \textsc{Small}$. Lemma 13 implies that every vertex in S_L can be adjacent to at most 1 vertex in $\textsc{Small}$ hence $|S_L| \geq |S_s|$. Thereafter Lemma 11 implies that $|S| > n/30r$ since otherwise

   $$|N(S)| \geq |N(S_L)| - |S_s| \geq 10|S_L| - |S_L| > 0.$$

   Finally the probability that there exists a connected component of size $n/30r \leq s \leq n/2$ in $G(H, n, p_0) \subset F_0$ is bounded by

   $$\sum_{s=n/30r}^{0.5n} \binom{n}{s} (1 - p_0)^{\frac{r!}{\mathrm{aut}(H)} \cdot s \binom{n-s}{r-1}} \leq \sum_{s=n/30r}^{0.5n} \left[\frac{en}{s} \cdot e^{-C_5 \ln n} \right]^s = o(1).$$

4. First we show that w.h.p. $|\textsc{Small}| \leq n^{0.1}$. Indeed by Markov's inequality,

$$\mathsf{Pr}\left[|\textsc{Small}| > n^{0.1}\right] \leq n^{-0.1} \cdot n\,\mathsf{Pr}\left[\mathsf{Bin}\left(\frac{r!}{\mathrm{aut}(H)}\binom{n-1}{r-1}, p_0\right) \leq \ln\ln n\right] = o(1).$$

Now let Q_j be a set of ϵn^2 edges not present in F_j and Q'_j be the subset of Q_j consisting of the edges that are not incident to $\textsc{Small}$. Then w.h.p. $|Q'_j| = (1+o(1))\epsilon n^2$. Every edge in Q'_j belongs to $C_6 n^{r-2}$ copies of H that are no present in F_j and every copy of H may cover at most $\binom{r}{2}$ edges in Q'_j. Therefore there exists a set W_i consisting of at least $C_6 n^{r-2} \cdot (1+o(1))\epsilon n^2/\binom{r}{2}$ distinct copies of H that intersect Q'_j. $H_{i_{j+1}}$ is uniformly distributed among the copies of H that are not present in F_j and are not incident to a vertex in $\textsc{Small}$. Thus

$$\mathsf{Pr}[iv] = \mathsf{Pr}[H_i \in W_i] \geq \frac{C_6 n^{r-2} \cdot (1+o(1))\epsilon n^2/\binom{r}{2}}{n^r} \geq C_7\epsilon = C_\epsilon. \qquad \blacktriangleleft$$

D Proof sketch of Theorems 4 and 5

To prove Theorem 5 we first indicate the edge set Q_1, consisting of the edges that are incident to vertices of degree 1. Then we delete these edges and the vertex set U_1 consisting of the vertices incident to them. Thereafter we use exactly the same techniques as above in order to find a Hamilton cycle in the remaining graph. We use half of the edges of that cycle and the edges in Q_1 to form a perfect matching.

Given the above, the only substantial difference is that while generating $\bar{G}(H,n,\tau_1)$ (in place of $\bar{G}(H,n,\tau_2)$) we stop at time $t^* = \min\{i : \delta(G'_i) = 1\}$. The proofs of all Lemmas with exception the proof of Lemma 14, follow in exactly the same way. For the proof of Lemma 14 we have to be slightly more cautious as we want to prove the corresponding statements for the subgraph that is spanned by $V \setminus U_1$. Thus we have to use $\textsc{Small} \setminus U_1$ and $\textsc{Large} \setminus U_1$ in place of $\textsc{Small}$ and $\textsc{Large}$ respectively.

E Proof of Theorem 9

Before proving Theorem 9, we derive an expression for a'/b' and establish the following upper bound on $\gamma(a,b,i)$.

▶ **Lemma 20.** *Consider an (a,b,i) covering of S by a path of length $v-1$ and an (a',b') subcovering with c' connected components. Let S_j be the subgraph of S covered by j^{th} connected component of the (a',b') subcovering. Let $f_j = |E(S_j)| - |V(S_j)| + 1$ and $f' = \sum_{j=1}^{c'} f_j$. Let k be the number of duplicate edges in the (a',b') subcovering, i.e. k is the smallest integer such that removing k edges from multigraph union of b' copies of H can yield a simple graph. Then*

$$\frac{a'}{b'} = v - 1 + \frac{c' - f' - k}{b'} \tag{3}$$

and

$$\gamma(a,b,i) \leq \begin{cases} v - 1 + \frac{1-f}{b} & (a,b,i) \text{ is edge-disjoint} \\ v - 1 - \frac{f}{b} & (a,b,i) \text{ is not edge-disjoint} \end{cases} \tag{4}$$

Proof. We compute a'. Note that each of the b' copies of H contributes v vertices, however vertices may be counted multiple times. We compute

$$a' = b'v - \left(b' - \sum_{j=1}^{c'} 1 - f_j\right) - k = b'(v-1) + c' - f' - k,$$

where the first term subtracted corresponds to doubling counting vertices in each connected component and subtracting k corresponds to removing double counting for vertices adjacent to edges of S that are covered multiple times.

By definition, $\gamma(a,b,i) \le a/b$. For the $(a',b') = (a,b)$ subcover that is the entire (a,b,i) cover, $c' = 1$, $f' = f$ and $k = 0$ if (a,b,i) is edge-disjoint and $k \ge 1$ if (a,b,i) is not edge-disjoint. Thus, Equation (4) follows directly from Equation (3). ◀

Proof of Theorem 9. We consider each case separately.

Case: $f = 0$. Consider an (a,b,i) covering. If (a,b,i) is edge-disjoint, then $b \ge \gamma$. It follows from Equation (4) that

$$\gamma(a,b,i) \le \begin{cases} v - 1 + \frac{1}{\beta} & (a,b,i) \text{ is edge-disjoint} \\ v - 1 & (a,b,i) \text{ is not edge-disjoint.} \end{cases}$$

Thus $\bar{\gamma} = \max_{(a,b,i)\in\mathcal{I}(S,H)} \gamma(a,b,i) \le v - 1 + 1/\beta$.
Next consider an edge-disjoint cover of S by β copies of H, (a,β,i). By Equation (3), for any (a',b') subcover of the (a,β,i) cover,

$$\frac{a'}{b'} = v - 1 + \frac{c'}{b'}.$$

This value is minimized with $c' = 1$ and $b' = \beta$, which is achieved by the (a,β) subcover which is the whole cover. Thus $\gamma(a,\beta,i) = v - 1 + 1/\beta$, and so $\bar{\gamma} \ge v - 1 + 1/\beta$.

Case: $f = 1$. By Equation (4), $\gamma(a,b,i) \le v - 1$ for all (a,b,i) and so it follows that $\bar{\gamma} \le v - 1$.
Next consider an edge-disjoint cover of S, (a,b,i). By Equation (3), for any (a',b') subcover of the (a,β,i) cover,

$$\frac{a'}{b'} = v - 1 + \frac{c' - 1}{b'}.$$

This value is minimized with $c' = 1$, which is achieved by the (a,b) subcover which is the whole cover. Thus $\gamma(a,b,i) = v - 1$, and so $\bar{\gamma} \ge v - 1$.

Case: $f \ge 2$. Consider an (a,b,i) cover. By Equation (3),

$$\gamma(a,b,i) = \min_{(a',b')\in\mathcal{D}(a,b,i)} \frac{a'}{b'} = \min_{a',b',c',k} v - 1 + \frac{c' - f' - k'}{b'}.$$

Let t' and e' be the number of edges and vertices of S covered by the subcover, so $e' = t' - c' + f' + k$. It follows

$$\gamma(a,b,i) = \min_{t',e',b'} v - 1 + \frac{t' - e'}{b'}. \tag{5}$$

To give an upper bound on $\gamma(a,b,i)$, we construct a subcover of the (a,b,i) cover as follows. Let X be a subgraph of S with t^* vertices and e^* edges such that $t^*/e^* = \eta$. Let t', e', b' correspond to the subcover that minimally covers X, and let C be the subgraph of S covered by this subcover (so X is a subgraph of C).

We claim that $t'-e' \leq t^*-e^*$. Note that $t'-t^* = |V(C)\setminus V(X)|$ and $e'-e^* = |E(C)\setminus E(X)|$. In each component of $C \setminus E(X)$, at least one vertex is included in $V(X)$. Since the number of vertices in a connected component is at least the number of edges in the connected component minus one, and at least one vertex in each connected component is not included in $V(C) \setminus V(X)$, it follows that $|V(C) \setminus V(X)| \geq |E(C) \setminus E(X)|$. Thus $t' - t^* \leq e' - e^*$ and the claim follows.
By considering this subcover with parameters t', e', b', we obtain

$$\gamma(a,b,i) \leq v - 1 + \frac{t'-e'}{b'} \leq v - 1 + \frac{t^*-e^*}{e^*} = v - 2 + \eta$$

since $b' \leq e^*$ and $t^* - e^* \leq 0$. It follows that $\bar{\gamma} \leq v - 2 + \eta$.
Finally to lower bound $\bar{\gamma}$ we consider a cover in which there are $b = |E(S)|$ copies of H and each copy covers precisely one edge of S. In this case in all subcovers $b' = e'$. By Equation (5)

$$\gamma(a,b,i) = \min_{t',e',b'} v - 1 + \frac{t'-e'}{b'} = \min_{t',e'} v - 2 + \frac{t'}{e'} = v - 2 - \eta.$$

Thus $\bar{\gamma} \geq v - 2 + \eta$. ◀

Random-Cluster Dynamics in $\mathbb{Z}^2$: Rapid Mixing with General Boundary Conditions

Antonio Blanca
Department of Computer Science and Engineering, Pennsylvania State University, USA
ablanca@cse.psu.edu

Reza Gheissari
Courant Institute of Mathematical Sciences, New York University, USA
reza@cims.nyu.edu

Eric Vigoda
School of Computer Science, Georgia Institute of Technology, USA
vigoda@cc.gatech.edu

Abstract

The random-cluster (FK) model is a key tool for the study of phase transitions and for the design of efficient Markov chain Monte Carlo (MCMC) sampling algorithms for the Ising/Potts model. It is well-known that in the high-temperature region $\beta < \beta_c(q)$ of the q-state Ising/Potts model on an $n \times n$ box Λ_n of the integer lattice $\mathbb{Z}^2$, spin correlations decay exponentially fast; this property holds even arbitrarily close to the boundary of Λ_n and uniformly over all boundary conditions. A direct consequence of this property is that the corresponding single-site update Markov chain, known as the Glauber dynamics, mixes in optimal $O(n^2 \log n)$ steps on Λ_n for all choices of boundary conditions. We study the effect of boundary conditions on the FK-dynamics, the analogous Glauber dynamics for the random-cluster model.

On Λ_n the random-cluster model with parameters (p, q) has a sharp phase transition at $p = p_c(q)$. Unlike the Ising/Potts model, the random-cluster model has non-local interactions which can be forced by boundary conditions: external wirings of boundary vertices of Λ_n. We consider the broad and natural class of boundary conditions that are *realizable* as a configuration on $\mathbb{Z}^2 \setminus \Lambda_n$. Such boundary conditions can have many macroscopic wirings and impose long-range correlations even at very high temperatures ($p \ll p_c(q)$). In this paper, we prove that when $q > 1$ and $p \neq p_c(q)$ the mixing time of the FK-dynamics is polynomial in n for *every* realizable boundary condition. Previously, for boundary conditions that do not carry long-range information (namely wired and free), Blanca and Sinclair (2017) had proved that the FK-dynamics in the same setting mixes in optimal $O(n^2 \log n)$ time. To illustrate the difficulties introduced by general boundary conditions, we also construct a class of non-realizable boundary conditions that induce slow (stretched-exponential) convergence at high temperatures.

2012 ACM Subject Classification Mathematics of computing → Markov-chain Monte Carlo convergence measures; Theory of computation → Random walks and Markov chains

Keywords and phrases Markov chain, mixing time, random-cluster model, Glauber dynamics, spatial mixing

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.67

Category RANDOM

Related Version A full version of the paper is available at `https://arxiv.org/abs/1807.08722`.

Funding *Antonio Blanca*: Research supported in part by NSF grants CCF-1617306 and CCF-1563838.
Eric Vigoda: Research supported in part by NSF grants CCF-1617306 and CCF-1563838.

Acknowledgements The authors thank the anonymous referees for their helpful suggestions.



© Antonio Blanca, Reza Gheissari, and Eric Vigoda;
licensed under Creative Commons License CC-BY
Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh; Article No. 67; pp. 67:1–67:19



Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Statistical physics models are designed to study physical phase transitions where a small change in a parameter which controls the local interactions, such as temperature, causes abrupt changes in the macroscopic behavior of the system. Phase transitions are captured by the onset of long-range correlations between vertices in the underlying graph, the infinite 2-dimensional integer lattice graph $\mathbb{Z}^2$ being a widely considered example. These long-range correlations manifest in the asymptotic effect of "boundary conditions" in large finite volumes. For example, if we take an $n \times n$ box Λ_n of $\mathbb{Z}^2$ and fix a configuration on the boundary of this box, as we formalize momentarily, this fixed boundary condition may affect the static (equilibrium state) and dynamic (approach to equilibrium) properties of the system.

The most notable and well-studied statistical physics model is the Ising/Potts model of ferromagnetism. The (ferromagnetic) Ising/Potts model on a (finite) graph, say the $n \times n$ box $\Lambda_n \subset \mathbb{Z}^2$ with nearest-neighbor edges $E(\Lambda_n)$, is defined on the set of spin assignments $\{1, \dots, q\}^{\Lambda_n}$. The probability of a configuration $\sigma \in \{1, \dots, q\}^{\Lambda_n}$ in the associated Gibbs distribution μ_{Λ_n} is proportional to $\exp(\beta H(\sigma))$, where $H(\sigma)$ is the number of edges of Λ_n whose endpoints are assigned the same spin in σ; the parameter $\beta > 0$ corresponds to the inverse temperature and controls the strength of the nearest-neighbor interactions.

An Ising/Potts boundary condition τ is a fixed assignment of spins to $\partial\Lambda_n$, the (inner) boundary of Λ_n; i.e., those vertices in Λ_n that are adjacent to vertices in $\mathbb{Z}^2 \setminus \Lambda_n$. The Gibbs distribution on Λ_n conditioned on the fixed assignment τ to $\partial\Lambda_n$, denoted $\mu^\tau_{\Lambda_n}$, is used for example to define the infinite Ising/Potts Gibbs measures on $\mathbb{Z}^2$. These are obtained as the limits of the distributions on finite boxes for distinct boundary conditions τ; i.e, $\lim_{n\to\infty} \mu^\tau_{\Lambda_n}$ for different τ.

On $\mathbb{Z}^2$, it is known that the Ising/Potts model undergoes a sharp phase transition at a critical point $\beta = \beta_c(q) = \ln(1 + \sqrt{q})$ [31, 2]. This phase transition marks the onset of long-range correlations and can also be understood as a transition in the number of (unique vs. multiple) infinite-volume Gibbs measures. In finite regions of $\mathbb{Z}^2$ such as Λ_n, this phase transition corresponds to whether an arbitrary boundary condition τ on $\partial\Lambda_n$ may have macroscopic effects on the Gibbs distribution. For instance, in the low-temperature region $\beta > \beta_c(q)$, if τ is the all "1" configuration on $\partial\Lambda_n$, the spins of all vertices, even those near the center of Λ_n will prefer the spin "1" and thus align with the boundary. In contrast, in the high-temperature region $\beta < \beta_c(q)$, there is exponential decay (with distance) of spin correlations: crucially this holds uniformly over all boundary conditions and over all vertices (i.e., even for those near the boundary); this property is known as *strong spatial mixing (SSM)*.

This phase transition also exhibits itself in the dynamic properties of the system, e.g., through the speed of convergence to stationarity of natural Markov chains for the Ising/Potts model. The classical Glauber dynamics, for example, which in each step updates the spin of a random vertex according to the spins of its neighbors, is known to converge in $\Theta(n^2 \log n)$ steps when $\beta < \beta_c(q)$ [28, 8, 2, 1]; this bound relies on the SSM property described above and, as such, it holds for *every* fixed boundary condition. In contrast, when $\beta > \beta_c(q)$ the speed of convergence of the Glauber dynamics is expected to depend crucially on the boundary condition and understanding its behavior for general boundaries is a long-standing open problem. At the moment, it is known that Glauber dynamics requires exponentially (in n) many steps to converge for free (no boundary) and periodic (toroidal) boundary conditions [34, 7, 16] and, in the special case of the Ising model ($q = 2$), sub-exponentially many steps for uniform (e.g., all "1") boundaries [25, 29].

Our focus here is the random-cluster (FK) model [13], which is a random graph model intimately connected to the Ising/Potts model. Indeed, it has been central to the study of the Ising/Potts phase transition (see, e.g., the recent breakthroughs on $\mathbb{Z}^2$ [2, 12, 11]) and plays an indispensable role in the design of efficient Markov Chain Monte Carlo (MCMC) algorithms for the Ising/Potts model (e.g., the Swendsen-Wang dynamics [33, 22]). We study the effects of boundary conditions on Λ_n on the speed of convergence of the FK-dynamics, the analog of the Ising/Potts Glauber dynamics for the random-cluster model. Despite the close connection between these models, the boundary effects are fundamentally different. Whereas the SSM property of the Ising/Potts model at $\beta < \beta_c(q)$ is uniform over the choice of boundary condition, in the random-cluster setting, SSM is limited to only a few select choices of boundary conditions.

We seek to understand the dynamics in situations where spatial mixing is destroyed near the boundary by the boundary condition. First we establish that for all *realizable* FK boundary conditions (those which are consistent with the planarity of $\mathbb{Z}^2$), the FK-dynamics converges in polynomially many (in n) steps, both at high and low temperatures. To illustrate the difficulties introduced by general boundary conditions, we also construct a class of non-realizable boundary conditions that induce slow (stretched-exponential) convergence at high temperatures.

The random-cluster model. For a graph $G = (V, E)$ and parameters $p \in (0, 1)$ and $q > 0$, random-cluster configurations are subsets of edges in $\Omega = \{S \subseteq E\}$, with the probability of $S \subset E$ given by

$$\pi_{G,p,q}(S) = \frac{1}{Z} p^{|S|}(1-p)^{|E\setminus S|} q^{c(S)}\,, \tag{1}$$

where $c(S)$ is the number of connected components (including isolated vertices) in the subgraph (V, S), and $Z = Z_{G,p,q}$ is the normalizing constant that makes $\pi_{G,p,q}$ a probability measure.

For *integer* $q \geq 2$ connectivities in the random-cluster model correspond to spin correlations in the Ising/Potts setting, and it is consequently viewed as a generalization of the ferromagnetic Ising/Potts model to non-integer values of q. The random-cluster model, however, is not a spin system in the usual sense, as the weight of a configuration S is not a function of local interactions between edges in G, but instead of global connectivity properties. This non-local structure is a crucial feature of the model but significantly complicates its analysis; for example, it allows boundary conditions to induce long-range connections in G.

We consider the random-cluster model on the $n \times n$ box Λ_n of $\mathbb{Z}^2$, where, for $q \geq 1$, the model is also known to exhibit a phase transition corresponding to the emergence of long-range correlations in the form of large connected components [2]. That is, there exists a critical value $p = p_c(q) = \sqrt{q}/(\sqrt{q}+1)$ such that, with high probability, when $p < p_c(q)$ all connected components are of size $O(\log n)$ whereas when $p > p_c(q)$ there exists a "giant" component of size $\Theta(n^2)$ [2].

A random-cluster boundary condition ξ on $\partial\Lambda_n$ is a partition $\{\xi_1, \xi_2, ...\}$ of the boundary vertices such that all vertices in ξ_i are connected via "ghost" (or external) wirings; these connections are considered in the counting of $c(S)$ in (1) and can therefore impose highly non-local interactions. Of particular interest are boundary conditions where the partition is induced by the connectivity components of a random-cluster configuration on $E(\mathbb{Z}^2) \setminus E(\Lambda_n)$. We call such boundary conditions *realizable*. In fact, many works, including the standard text [21], often restrict attention to realizable boundary conditions, but non-realizable boundary conditions are still relevant in some cases.

The FK-dynamics. In this paper we study the *mixing time* of the FK-dynamics in the presence of boundary conditions. (The mixing time is the number of steps until a Markov chain is close to its stationary distribution in total variation distance, starting from the worst possible initial configuration.) For a configuration $S_t \subseteq E(\Lambda_n)$, a transition $S_t \to S_{t+1} \subseteq E(\Lambda_n)$ of the FK-dynamics is defined as follows:

1. Choose an edge $e \in E(\Lambda_n)$ uniformly at random;
2. let $S_{t+1} = S_t \cup \{e\}$ with probability

$$\frac{\pi_{\Lambda_n,p,q}(S_t \cup \{e\})}{\pi_{\Lambda_n,p,q}(S_t \cup \{e\}) + \pi_{\Lambda_n,p,q}(S_t \setminus \{e\})} = \begin{cases} \frac{p}{q(1-p)+p} & \text{if } e \text{ is a "cut-edge" in } (\Lambda_n, S_t); \\ p & \text{otherwise;} \end{cases}$$

3. else let $S_{t+1} = S_t \setminus \{e\}$.

We say e is a *cut-edge* in (Λ_n, S_t) if the number of connected components in $S_t \cup \{e\}$ and $S_t \setminus \{e\}$ differ. Under a boundary condition ξ, the property of e being a cut-edge is defined with respect to the augmented graph (Λ_n, S_t^ξ). The FK-dynamics converges to (1) by construction, and we study its mixing time. We say the dynamics is *rapidly mixing* if the mixing time is polynomial in $|V|$, and *torpidly mixing* when the mixing time is exponential in $|V|^\varepsilon$ for some $\varepsilon > 0$.

Results. The FK-dynamics is quite powerful since it is known to mix in $\Theta(n^2 \log n)$ steps for all $q > 1$ both at high and low temperatures (i.e., for all $p \neq p_c(q)$) for certain "nice" boundary conditions that do not carry information about random-cluster connectivities in non-local ways: namely, configurations in different regions of Λ_n do not interact through these boundaries [5]. (In comparison, the Ising/Potts Glauber dynamics is torpidly mixing in the low-temperature regime.) Specifically, the tight mixing time bound in [5] holds under boundary conditions that are free (no boundary condition), wired (all boundary vertices are connected to one another) or periodic (the torus). More recently, [15] examined the cutoff phenomenon in the FK-dynamics at $p \ll p_c(q)$; they also restricted attention to periodic boundaries. At the critical $p = p_c(q)$ the FK-dynamics may exhibit torpid mixing depending on the "order" (i.e., the continuity) of the phase transition [16, 18]; notably, when $q \gg 1$ and $p = p_c(q)$, the mixing time may be exponential or sub-exponential depending on the choice of boundary conditions [17].

The stability of the FK-dynamics to the choice of boundary conditions remained unclear at $p \neq p_c(q)$; we show that the FK-dynamics is in fact rapidly mixing for *all* realizable boundary conditions at $p \neq p_c(q)$.

▶ **Theorem 1.1.** *For every $q > 1$, $p \neq p_c(q)$, there exists a constant $C > 0$ such that the mixing time of the FK-dynamics on the $n \times n$ box $\Lambda_n \subset \mathbb{Z}^2$ with any realizable boundary condition is $O(n^C)$.*

We pause to comment on the proof of Theorem 1.1. The proofs of fast mixing when $p \neq p_c(q)$ have relied crucially on a strong spatial mixing property, which in the random-cluster model would say that correlations between edges (even near $\partial\Lambda_n$) decay exponentially in the graph distance between them. It is easy to construct examples of realizable boundary conditions where this correlation does not decay at all, even if $p \ll p_c(q)$, as the boundary can enforce long-range interactions. Since the exponential decay of correlations does hold for edges at distance $\Theta(\log n)$ away from $\partial\Lambda_n$, we are able to reduce the proof of Theorem 1.1 to proving a polynomial upper bound for the mixing time of the FK-dynamics on thin rectangles of dimension $n \times \Theta(\log n)$ with realizable boundary conditions. This reduction is a byproduct

of a more general framework we describe in Section 3 for deriving mixing time estimates from spatial mixing properties. The analysis of the FK-dynamics on thin rectangles is then the key technical challenge for us; see Theorem 4.1 and Section 4.1 for a detailed outline of its proof and the novelties therein.

Theorem 1.1 shows a polynomial upper bound on the mixing time, uniformly over all realizable boundary conditions. Utilizing this theorem we can prove near-optimal $\tilde{O}(n^2)$ mixing time for "typical" boundaries. The notion of typicality should be understood as with high probability under some distribution over realizable boundary conditions, with a natural choice being the marginal of the infinite-volume random-cluster measure $\pi_{\mathbb{Z}^2,p,q}$ on $E(\mathbb{Z}^2) \setminus E(\Lambda_n)$ (when $p \neq p_c(q)$ this measure is unique: see [21]).

▶ **Theorem 1.2.** *Let $q > 1$, $p \neq p_c(q)$ and suppose ω is a random-cluster configuration sampled from $\pi_{\mathbb{Z}^2,p,q}$. Let ξ_ω be the boundary condition on $\partial\Lambda_n$ induced by the edges of ω in $\mathbb{Z}^2 \setminus \Lambda_n$. Then, there exists a constant $C > 0$ such that with probability $1 - o(1)$ the mixing time of the FK-dynamics on the $n \times n$ box Λ_n with boundary condition ξ_ω is $O(n^2(\log n)^C)$.*

The proof of Theorem 1.2 uses Theorem 1.1 in a crucial way. Typical boundary conditions do not exhibit the strong spatial mixing property from [5]; however, for such boundary conditions we are able to prove that correlations between edges near the boundary decay exponentially in their graph distance divided by a $\Theta(\log n)$ factor. Using this spatial mixing bound, together with the aforementioned general framework in Section 3, we reduce bounding the mixing time on Λ_n with typical boundaries to bounding the mixing time on $\Theta((\log n)^2) \times \Theta((\log n)^2)$ rectangles with arbitrary realizable boundary conditions. Theorem 1.1 implies that the mixing time of the FK-dynamics in these smaller rectangles is at most poly-logarithmic in n. Similar classes of typical boundary conditions were considered in [18] at $p = p_c(q)$.

Given that our rapid mixing result for realizable boundaries relies heavily on the planarity of the boundary connections in $\mathbb{Z}^2 \setminus \Lambda_n$, one may wonder whether rapid mixing holds for all possible FK boundary conditions (including those not realizable as configurations on $\mathbb{Z}^2 \setminus \Lambda_n$). We answer this in the negative, showing that there exist (non-realizable) boundaries for which the FK-dynamics is torpidly mixing even while $p \neq p_c(q)$. In fact, this torpid mixing holds at $p \ll p_c(q)$, which may sound especially surprising as correlations in the Gibbs measure $\pi_{\Lambda_n,p,q}$ die off faster as p decreases.

▶ **Theorem 1.3.** *Let $q > 2$. For every $\alpha \in (0, \frac{1}{2}]$ and $\lambda > 0$ there exists a boundary condition ξ, such that when $p = \lambda n^{-\alpha}$ the mixing time of the FK-dynamics on the $n \times n$ box Λ_n with boundary condition ξ is $\exp(\Omega(n^\alpha))$.*

Our proof of this theorem is constructive: we take any graph G on m edges for which torpid mixing of the FK-dynamics is known at some value of $p(m) < p_c(q)$ and show how to embed G into the boundary of Λ_n. We then develop a procedure to transfer mixing time bounds from G to Λ_n. The high-level idea is that for sufficiently small $p(m)$ the effect of the configuration away from the boundary is negligible, and the mixing time of the FK-dynamics on G completely governs the mixing time of FK-dynamics near the boundary $\partial\Lambda_n$. Therefore, we can use known torpid mixing results for the mean-field random-cluster model (the case where G is the complete graph) in its critical window at $q > 2$ [20, 4, 14, 19].

We remark that the requirement $q > 2$ appears to be sharp for Theorem 1.3, since it was recently shown that the mixing time of FK-dynamics when $q = 2$ is at most polynomial in the number of vertices on *any* graph and at every $p \in (0, 1)$ [22]. It is expected that this rapid mixing holds for all $q \leq 2$. We believe that our torpid mixing result may extend to small, but $\Omega(1)$ values of $p < p_c(q)$, though our current proof does not allow for this. In

principle, one would want to embed a bounded degree graph into $\partial\Lambda_n$, so that the value of p at which it exhibits slow mixing is $\Omega(1)$. There are already several examples of bounded degree graphs where torpid mixing is known [10, 6, 7, 16, 17].

Finally, we remark that by slight adaptations of the comparison results in [35, 36, 4], our theorems translate (up to polynomial factors in n) to bounds for the mixing times of popular non-local dynamics like the Chayes-Machta dynamics [9] and the Swendsen-Wang dynamics on FK configurations [35, 36, 4].

The paper is organized as follows. In Section 2, we define various preliminary notions that are used in our proofs. In Section 3, we introduce a general framework to deduce mixing time estimates on Λ_n from spatial and local mixing properties. We then present our key rapid mixing result for thin rectangles (Theorem 4.1) in Section 4, before completing the proof of Theorem 1.1 in Section 5. The proofs from Section 4 are deferred to Section 6, and those of Theorems 1.2 and 1.3 are included in the full manuscript [3].

2 Preliminaries: the random-cluster model in $\mathbb{Z}^2$

In this section we introduce a number of definitions, notation, and background results that we will refer to repeatedly. More details and proofs can be found in the books [21, 24]. We will be considering the random-cluster model on rectangular subsets of $\mathbb{Z}^2$ of the form $\Lambda_{n,l} = \{0, ..., n\} \times \{0, ..., l\} = [\![0, n]\!] \times [\![0, l]\!]$. When $n = l$, we use Λ_n for $\Lambda_{n,n}$. For simplicity, in this preliminary section we shall focus on the $n = l$ case, but everything stated here holds more generally for rectangular subsets with $n \neq l$.

Abusing notation, we will also use Λ_n for the graph $(\Lambda_n, E(\Lambda_n))$ where $E(\Lambda_n)$ consists of all nearest neighbor pairs of vertices in Λ_n. We denote by $\partial\Lambda_n$ the (inner) boundary of Λ_n; that is the vertex set consisting of all vertices in Λ_n adjacent to vertices in $\mathbb{Z}^2 \setminus \Lambda_n$. A *boundary condition* ξ of Λ_n is a partition of the vertices in $\partial\Lambda_n$. When $u, v \in \partial\Lambda_n$ are in the same element of ξ, we say that they are *wired* in ξ. If there exists a random-cluster configuration ω on $E(\mathbb{Z}^2) \setminus E(\Lambda_n)$ such that $u, v \in \partial\Lambda_n$ are connected in ω if and only if they are wired in ξ, we say that the boundary condition ξ is *realizable*.

For $p \in (0, 1)$ and $q > 0$, the *random-cluster model* on Λ_n with boundary conditions ξ is the probability measure $\pi^{\xi}_{\Lambda_n,p,q}$ over the subsets $S \subseteq E(\Lambda_n)$ given by (1) with the $q^{c(S)}$ term replaced by $q^{c(S;\xi)}$, where $c(S;\xi)$ corresponds to the number of connected components in the augmented graph (Λ_n, S^{ξ}) and S^{ξ} adds auxiliary edges between all pairs of vertices in $\partial\Lambda_n$ that are in the same element of ξ. Every random-cluster configuration $S \subseteq E(\Lambda_n)$, can be identified with some $\omega : E(\Lambda_n) \to \{0, 1\}$ via $\omega(e) = 1$ if $e \in S$ (e is *open*) and $\omega(e) = 0$ if $e \notin S$ (e is *closed*). We sometimes interchange vertex sets with the subgraph they induce; e.g., the random-cluster configuration on a set $B \subset \mathbb{Z}^2$ corresponds to the configuration in the subgraph induced by B. We omit the subscripts p, q when understood from context.

Exponential decay of connectivities (EDC). A consequence of the results in [1, 2] is that for every $q > 1$ and $p < p_c(q)$, there is a $c = c(p, q) > 0$ such that for every boundary condition ξ and all $u, v \in \Lambda_n$,

$$\pi^{\xi}_{\Lambda_n,p,q}(u \xleftrightarrow{\Lambda_n} v) \leq \mathrm{e}^{-cd(u,v)}, \tag{2}$$

where $d(u, v)$ is the graph distance between u, v in $\mathbb{Z}^2$ and $u \xleftrightarrow{\Lambda_n} v$ denotes that there is an open path between u and v in the FK configuration on $E(\Lambda_n)$ (not using the connections of ξ).

Monotonicity. Define a partial order over boundary conditions by $\xi \leq \eta$ if the partition corresponding to ξ is *finer* than that of η. The extremal boundary conditions then, are the *free* boundary where $\xi = \{\{v\} : v \in \partial\Lambda_n\}$, which we denote by $\xi = 0$, and the *wired* boundary where $\xi = \{\partial\Lambda_n\}$, denoted by $\xi = 1$. When $q > 1$, the random-cluster model satisfies the following monotonicity in boundary conditions: if ξ, η are two boundary conditions on $\partial\Lambda_n$ with $\xi \leq \eta$, then $\pi^{\xi}_{\Lambda_n} \preceq \pi^{\eta}_{\Lambda_n}$, which is to say that $\pi^{\eta}_{\Lambda_n}$ stochastically dominates $\pi^{\xi}_{\Lambda_n}$ with respect to the natural partial order on FK configurations.

Planar duality. Let $\Lambda_n^* = (\Lambda_n^*, E(\Lambda_n^*))$ denote the planar dual of Λ_n. That is, Λ_n^* corresponds to the set of faces of Λ_n, and for each $e \in E(\Lambda_n)$, there is a dual edge $e^* \in E(\Lambda_n^*)$ connecting the two faces bordering e. The random-cluster distribution satisfies $\pi_{\Lambda_n,p,q}(S) = \pi_{\Lambda_n^*,p^*,q}(S^*)$, where S^* is the dual configuration to $S \subseteq E$ (i.e., $e^* \in S^*$ iff $e \notin S$), and $p^* = q(1-p)/(q(1-p)+p)$. Notice that the infinite graph $\mathbb{Z}^2$ is isomorphic to its dual. The unique value of p satisfying $p = p^*$, denoted $p_{sd}(q)$, is called the *self-dual point* and [2] established that $p_c(q) = p_{sd}(q)$; recall that $p_c(q)$ is the critical point for the phase transition in $\mathbb{Z}^2$.

Mixing time and spectral gap. Consider an ergodic (i.e., irreducible and aperiodic) Markov chain $\mathcal{M}$ with finite state space Ω, transition matrix P and stationary distribution μ. The *mixing time* of $\mathcal{M}$ is given by $t_{\text{MIX}} := t_{\text{MIX}}(1/4)$, where $t_{\text{MIX}}(\varepsilon) = \min\{t : \max_{X_0 \in \Omega} \|P^t(X_0, \cdot) - \mu\|_{\text{TV}} \leq \varepsilon\}$ and $\|\cdot\|_{\text{TV}}$ denotes total-variation distance. For any positive $\varepsilon < 1/2$, we have $t_{\text{MIX}}(\varepsilon) \leq \lceil \log_2 \varepsilon^{-1} \rceil \, t_{\text{MIX}}$. We use $t_{\text{MIX}}(\Lambda_n^{\xi})$ to denote the mixing time of the FK-dynamics on $\Lambda_n \subset \mathbb{Z}^2$ with boundary condition ξ. If P is irreducible and reversible with respect to μ, then it has real eigenvalues $1 = \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_{|\Omega|} \geq -1$. The *spectral gap* of P is defined by $\text{gap}(P) = 1 - \max\{|\lambda_2|, |\lambda_{|\Omega|}|\}$, and the inverse of the spectral gap captures the mixing time up to a $O(\log(\mu_{\min}^{-1}))$ factor, where $\mu_{\min} := \min_{\omega \in \Omega} \mu(\omega)$. For the various dynamics consider in this paper this factor is $\text{poly}(n)$.

FK-dynamics and duality. Each run of the FK-dynamics on Λ_n, with realizable boundary conditions ξ and parameters p, q, determines a valid run of the FK-dynamics on the dual graph Λ_n^* with boundary conditions ξ^* and parameters p^*, q. (Simply identify the FK configuration in each step with its dual configuration; it can be straightforwardly verified that the transitions of the FK-dynamics on the dual graph occur with the correct probabilities.) Hence, the two dynamics have the same mixing times.

▶ Remark 2.1. The edge-set of the dual graph Λ_n^* is not exactly in correspondence with the edge-set of a rectangle $\Lambda^* = \{-\frac{1}{2}, ..., n+\frac{1}{2}\} \times \{-\frac{1}{2}, ..., n+\frac{1}{2}\}$ as it does not include any edges that are between boundary vertices of Λ^*. All the proofs in the paper carry through, only with the natural minor geometric modifications, to the case of rectangles Λ_n with modified edge-set that only contains edges edges with at least one endpoint in $\Lambda_n \setminus \partial\Lambda_n$. The dual of this modified graph is then a $(n-1) \times (n-1)$ rectangle with all nearest-neighbor edges. With these considerations, it often suffices for us to prove our theorems for $p < p_c(q)$. For example, it is sufficient to prove Theorem 1.1 for $p < p_c(q)$.

3 Mixing time upper bounds: a general framework

In this section we introduce a general framework for bounding the mixing time of the FK-dynamics on $\Lambda_n = (\Lambda_n, E(\Lambda_n))$ by its mixing times on certain smaller subsets. In [5] it was shown that a strong form of spatial mixing (encoding exponential decay of correlations uniformly over subsets of Λ_n) implies optimal mixing of the FK-dynamics. However, this

notion, known as *strong spatial mixing (SSM)* and described in Remark 3.2, does not hold for many boundary conditions for which fast mixing of the FK-dynamics is still expected. To circumvent this, we introduce a weaker notion, which we call *moderate spatial mixing (MSM)*.

We introduce some notation first. For a set $R \subseteq \Lambda_n$, let $E(R) \subseteq E_n$ be the set of edges of $E(\Lambda_n)$ with both endpoints in R. We will denote by R^c the vertex set $\Lambda_n \setminus R$ and by $E^c(R)$ the edge-complement of R; i.e., $E^c(R) := E(\Lambda_n) \setminus E(R)$. For a configuration $\omega : E(\Lambda_n) \to \{0,1\}$, we will use $\omega(R)$, or alternatively $\omega(E(R))$, for the configuration of ω on $E(R)$. With a slight abuse of notation, for an edge set $F \subseteq E(\Lambda_n)$, we use $\{F = \omega\}$ for the event that the configuration on F is given by ω; when ω is the all free or the all wired configuration, we simply use $\{F = 0\}$ and $\{F = 1\}$, respectively.

▶ **Definition 3.1.** *Let $\mathcal{B} = \{B_1, B_2, \ldots, B_k\}$ be a collection of subsets of $\Lambda_n = (\Lambda_n, E(\Lambda_n))$ and let ξ be a boundary condition on Λ_n. We say that* moderate spatial mixing (MSM) *holds on Λ_n for ξ, $\mathcal{B}$ and $\delta > 0$ if for all $e \in E(\Lambda_n)$, there exists $B_j \in \mathcal{B}$ such that*

$$\left|\pi^{\xi}_{\Lambda_n,p,q}(\, e = 1 \mid E(B_j^c) = 1\,) - \pi^{\xi}_{\Lambda_n,p,q}(\, e = 1 \mid E(B_j^c) = 0\,)\right| \;\leq\; \delta\,. \tag{3}$$

In words, MSM holds for $\mathcal{B}$ if for every edge $e \in E(\Lambda_n)$ we can find B_j such that $e \in B_j$ and the "influence" of the configuration on $\Lambda_n \setminus B_j$ on the state of e is bounded by δ.

▶ Remark 3.2. SSM as defined in [5] holds when

MSM holds for a specific sequence of collections of subsets: if $\mathcal{B}_r$ is the set of subsets containing all the square boxes of side length $2r$ centered at each $e \in E(\Lambda_n)$ (intersected with $E(\Lambda_n)$), then SSM holds if MSM holds for $\mathcal{B}_r$ for every $r \geq 1$ with $\delta = \exp(-\Omega(r))$.

MSM does not capture the fast mixing of the FK-dynamics the way SSM does; it is easy to find collections of subsets for which MSM holds for *all* boundary conditions, including those boundary conditions for which we later prove slow mixing (Theorem 1.3). However, if, for a collection $\mathcal{B} = \{B_1, \ldots, B_k\}$, we also bound the mixing time of the FK-dynamics on every B_j, we can deduce a mixing time bound for the FK-dynamics on Λ_n. Let $t_{\text{MIX}}(B^\tau)$ denote the mixing time of the FK-dynamics on $B \subseteq \Lambda_n$ with boundary condition τ.

▶ **Definition 3.3.** *Let $\mathcal{B} = \{B_1, B_2, \ldots, B_k\}$ be a collection of subsets of $\Lambda_n = (\Lambda_n, E(\Lambda_n))$ and let ξ be a boundary condition on Λ_n. We say that* local mixing (LM) *holds for $\mathcal{B}$ and $T > 0$, if*

$$t_{\text{MIX}}\big(B_j^{(1,\xi)}\big) \leq T \quad \textit{and} \quad t_{\text{MIX}}\big(B_j^{(0,\xi)}\big) \leq T \qquad \textit{for all } j = 1, ..., k$$

where $(1,\xi)$ (resp., $(0,\xi)$) denotes the boundary condition on B_j induced by the event $\{E(B_j^c) = 1\}$ (resp. $\{E(B_j^c) = 0\}$) and the boundary condition ξ.

▶ Remark 3.4. When $B_j \cap \partial\Lambda_n = \emptyset$, $(1,\xi)$ and $(0,\xi)$ are simply the wired and free boundary condition on B_j, respectively. When $B_j \cap \partial\Lambda_n \neq \emptyset$, ξ could also induce some connections in $(1,\xi)$ and $(0,\xi)$.

Our next theorem, roughly speaking, establishes the following implication:

$$\text{MSM} + \text{LM} \implies \text{upper bound for mixing time of FK-dynamics,}$$

with the quality of the bound depending on the T for which LM holds. A similar (and inspiring) implication for the Glauber dynamics of the Ising model in graphs of bounded degree was established by Mossel and Sly in [30]; there, the notion of MSM is replaced by a form of spatial mixing which is stronger than SSM. The proof of this theorem is provided in the full version of this paper [3].

▶ **Theorem 3.5.** *Let ξ be a boundary condition on $\Lambda_n = (\Lambda_n, E(\Lambda_n))$ and let $\mathcal{B} = \{B_1, B_2, \ldots, B_k\}$ with $B_j \subset \Lambda_n$ for all $j = 1, \ldots, k$. If for ξ and $\mathcal{B}$, moderate spatial mixing holds for some $\delta \leq 1/(12|E(\Lambda_n)|)$ and local mixing holds for some $T > 0$, then $t_{\text{MIX}}(\Lambda_n^\xi) = O(Tn^2 \log n)$.*

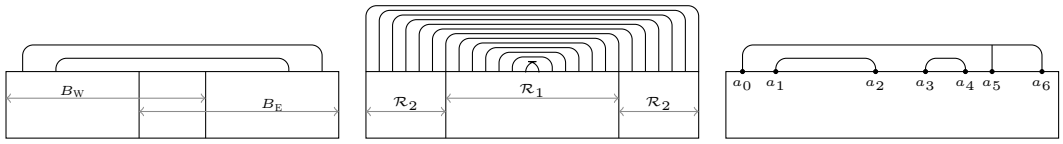


Figure 1 **(a)** A boundary condition for which no configuration in $B_{\mathrm{W}} \cap B_{\mathrm{E}}$ isolates $B_{\mathrm{W}} \setminus B_{\mathrm{E}}$ from $B_{\mathrm{E}} \setminus B_{\mathrm{W}}$. **(b)** A boundary condition ξ where every pair of overlapping rectangles must interact through ξ; the two *groups of rectangles* $\mathcal{R}_1, \mathcal{R}_2$ do not interact through ξ. **(c)** A boundary condition with disconnecting intervals: $[\![a_1, a_4]\!]$ of free-type; $[\![a_1, a_2]\!]$, $[\![a_3, a_4]\!]$, $[\![a_0, a_6]\!]$ of free-wired-type; and $[\![a_0, a_5]\!]$, $[\![a_5, a_6]\!]$ of wired-type.

4 Fast mixing on thin rectangles

The main difficulty in proving Theorem 1.1 using the general framework from Section 3 is obtaining mixing time estimates for the FK-dynamics on thin rectangles of dimension $\Theta(n) \times \Theta(\log n)$ with realizable boundary conditions. To motivate this we notice that in Λ_n, when $p \neq p_c(q)$, the influence of the boundary condition is lost with high probability at a distance $\Theta(\log n)$ from $\partial\Lambda_n$. (This is a consequence of the EDC property when $p < p_c(q)$, or the corresponding dual property when $p > p_c(q)$.) Consequently, the main challenge will be to establish the mixing time of the FK-dynamics in the annulus of width $\Theta(\log n)$ with realizable boundary conditions on the outside. The key ingredient in the proof of Theorem 1.1 will be the following mixing time estimate on thin rectangles. For an $n \times l$ rectangle $\Lambda_{n,l} = [\![0, n]\!] \times [\![0, l]\!]$, let $\partial_{\mathrm{N}}\Lambda_{n,l}$, $\partial_{\mathrm{E}}\Lambda_{n,l}$, $\partial_{\mathrm{S}}\Lambda_{n,l}$ and $\partial_{\mathrm{W}}\Lambda_{n,l}$ denote its north, east, south and west boundaries respectively.

▶ **Theorem 4.1.** *Consider $\Lambda_{n,l} = (\Lambda_{n,l}, E(\Lambda_{n,l}))$ for $l \leq n$ with an arbitrary* realizable *boundary condition ξ that is either free or wired on $\partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l} \cup \partial_{\mathrm{S}}\Lambda_{n,l}$. Then, for every $q > 1$ and $p \neq p_c(q)$, the mixing time of the FK-dynamics on $\Lambda_{n,l}$ is at most $\exp(O(l + \log n))$.*

When $l = O(\log n)$, this implies the mixing time is $n^{O(1)}$, which will be the setting of interest in our proofs. Moreover, we note that it suffices to prove Theorem 4.1 for the set of realizable boundary conditions ξ that are free on $\partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l} \cup \partial_{\mathrm{S}}\Lambda_{n,l}$ for all $p \neq p_c(q)$, as the set of boundary conditions dual to these are exactly the set of realizable boundary conditions that are wired on $\partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l} \cup \partial_{\mathrm{S}}\Lambda_{n,l}$; see Remark 2.1.

4.1 Proof of Theorem 4.1

Remarks about previous methods. To motivate our proof approach, we first mention some obstructions that FK boundary conditions present if we tried to adapt methods for the analogous problems in the context of spin systems. A traditional technique to proving mixing time bounds for thin rectangles is the canonical paths method ([26, 27, 23, 32]), which gives an upper bound that is exponential in the shorter side length; however, this approach relies on bounding the *cut-width* of $\Lambda_{n,l}$ which can be significantly distorted in the augmented graph $\Lambda_{n,l}^{\xi}$ by the FK boundary conditions ξ.

A sharper technique is an inductive scheme [8, 27, 18], whereby, the mixing time of the FK-dynamics on the rectangle $\Lambda_{n\times l}$ is bounded by the mixing times in two smaller (overlapping) rectangular subsets, e.g., the left two-thirds $B_{\mathrm{W}} = [\![0, \frac{2}{3}n]\!] \times [\![0, l]\!]$ and the right two-thirds $B_{\mathrm{E}} = [\![\frac{1}{3}n, n]\!] \times [\![0, l]\!]$; see Figure 1(a). This approach requires bounding the spectral gap of a *block dynamics*, whose updates consist of resampling the configuration on a random block $B_i \in \{B_{\mathrm{W}}, B_{\mathrm{E}}\}$ from $\pi^{\xi}_{\Lambda_{n,l}}$ conditional on the configuration on $E^c(B_i)$.

It follows from classical results that the spectral gap of the FK-dynamics on $\Lambda_{n,l}$ is bounded from below by the spectral of the block dynamics times the worst spectral gap of the FK-dynamics in any block B_i, assuming a worst-case configuration on $E^c(B_i)$; see, e.g., Proposition 3.4 in [27]. With the choice of blocks $B_{\mathrm{W}}, B_{\mathrm{E}}$, applying this recursively, the spectral gap of the FK-dynamics on $\Lambda_{n,l}$ is bounded from below by the gap of the block dynamics raised to a $\Theta(\log n)$ power. Therefore, establishing Theorem 4.1 requires an $\Omega(1)$ lower bound on the spectral gap of the block dynamics.

The spectral gap of the block dynamics is typically bounded by showing that after the first block update in either B_{W} or B_{E}, the configuration in $B_{\mathrm{W}} \cap B_{\mathrm{E}}$ disconnects the influence of the configuration on $B_{\mathrm{W}} \setminus B_{\mathrm{E}}$ from $B_{\mathrm{E}} \setminus B_{\mathrm{W}}$ with probability $\Omega(1)$. This would then allow a standard coupling argument to lower bound the spectral gap by $\Omega(1)$. In the presence of long-range boundary connections, however, it could be that no configuration on $B_{\mathrm{W}} \cap B_{\mathrm{E}}$ would disconnect the two sides from one another and facilitate coupling; see Figure 1(b) for such an example. As such, our choices of blocks will depend on the boundary conditions and will be chosen to allow for the block dynamics to couple in $O(1)$ time, while ensuring that the blocks are still at most a fraction of the size of the original rectangle, so that after $O(\log n)$ recursive steps we arrive at a sufficiently small base scale.

Definitions and main results for thin rectangles. As Figure 1(b) demonstrates, there are realizable boundary conditions that would force the blocks for the block dynamics to not be single rectangles, but rather unions of rectangular subsets of $\Lambda_{n,l}$ of the form $R = [\![a, b]\!] \times [\![0, l]\!]$ with $0 \le a < b \le n$; for ease of notation, let $[\![a, b]\!]^c = [\![0, n]\!] \setminus [\![a, b]\!]$. Our recursive argument will proceed instead on *groups of rectangles*.

▶ **Definition 4.2.** *Let $m = C_\star \log l$ where $C_\star$ is a suitably large constant. A* group of rectangles *$\mathcal{R} = \bigcup_{i=1}^{N(\mathcal{R})} R_i$ is the union of $N(\mathcal{R})$ disjoint rectangular subsets $R_i = [\![a_i, b_i]\!] \times [\![0, l]\!]$ of $\Lambda_{n,l}$ such that $W(R_i) := b_i - a_i \ge 2m$ for every $i = 1, ..., N(\mathcal{R})$.*

The requirement that the width $W(R_i)$ of every constituent rectangle R_i is at least $2m$, is so that the interior of the R_i's are can be isolated from the configuration on $E(\Lambda_{n,l}) \setminus E(\mathcal{R})$. Indeed, the constant $C_\star$ is chosen so that $C_\star > c^{-1}$ with c being the constant from the EDC property (2).

We show that for every group of rectangle $\mathcal{R}$ there is a choice of two suitable blocks, which in turn will be group of rectangles, for the block dynamics. By suitable we mean two group of rectangles whose width are a constant fraction of that of $\mathcal{R}$ and that are sufficiently isolated from one another in ξ; see Proposition 4.6. (The width of a group of rectangles $\mathcal{R} = \bigcup_{i=1}^{N(\mathcal{R})} R_i$, denoted $W(\mathcal{R})$, is simply the sum of the width of its constituent rectangles; that is $W(\mathcal{R}) = \sum_{i=1}^{N(\mathcal{R})} W(R_i)$.)

For this, we introduce the key notions of *disconnecting intervals* of a boundary condition ξ and *compatibility* of a group of rectangles $\mathcal{R} \subset \Lambda_{n,l}$ with ξ. These allow us to manage the unwieldy interactions that may be induced by the realizable boundary condition ξ. Roughly speaking, a disconnecting interval is a segment $[\![a, b]\!] \times \ell$ of $\partial_{\mathrm{N}}\Lambda_{n,l}$ that has no interaction through ξ with the remaining vertices in $\partial_{\mathrm{N}}\Lambda_{n,l}$.

▶ **Definition 4.3.** *For a* realizable *boundary condition ξ on $\Lambda_{n,l}$ that is free on $\partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{S}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$, an interval $[\![a, b]\!] \subset [\![0, n]\!]$ is called* disconnecting *of*

1. free-type*: if there are no boundary connections in ξ between $[\![a, b]\!] \times \{l\}$ and $[\![a, b]\!]^c \times \{l\}$.*
2. wired-type*: if there is a boundary component in ξ that contains both vertices (a, l) and (b, l).*

Observe that an interval can be both of free-type and of wired-type if (a,l) and (b,l) are connected through ξ but are not connected to any boundary vertex in $[\![a,b]\!]^c \times [\![0,l]\!]$; in this case, we may refer to the interval as being of *free-wired-type*; see Figure 1(c) for several examples.

We say a group of rectangles $\mathcal{R}$ is compatible with ξ if the boundary interactions between the rectangular subsets of $\mathcal{R}$ are limited in the following way.

▶ **Definition 4.4.** *Let $\mathcal{R} = \bigcup_{i=1}^{N(\mathcal{R})} R_i$ be a group of rectangles with $R_i = [\![a_i, b_i]\!] \times [\![0,l]\!]$ and $a_1 < b_1 < \ldots < a_{N(\mathcal{R})} < b_{N(\mathcal{R})}$. Let ξ be a realizable boundary condition on $\Lambda_{n,l}$ that is free on $\partial_{\mathrm{S}}\Lambda_{n,l} \cup \partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$, and free in all vertices in $\partial_{\mathrm{N}}\Lambda_{n,l}$ at distance at most m from $\partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$.*

We say $\mathcal{R}$ is compatible *with ξ, if*

1. *Between every two consecutive rectangles $R_i = [\![a_i, b_i]\!] \times [\![0,l]\!]$ and $R_{i+1} = [\![a_{i+1}, b_{i+1}]\!] \times [\![0,l]\!]$ the interval $[\![b_i - m, a_{i+1} + m]\!]$ is a disconnecting interval; and*
2. *The interval $[\![a_1 + m, b_{N(\mathcal{R})} - m]\!]$ is also a disconnecting interval.*

It is clear from the definition that $\Lambda_{n,l}$ is compatible with ξ: the first condition is vacuous, while the second is satisfied by the additional assumption that all vertices a distance at most m from $\partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$ are free (i.e., they appear as singletons in the corresponding boundary partition)

With the definition of group of rectangles, disconnecting intervals and compatibility in hand, we can now design a "splitting" algorithm for picking two blocks $\mathcal{R}_{\mathrm{INT}}, \mathcal{R}_{\mathrm{EXT}}$ for the block dynamics with the desired properties. The following lemma, proved in Section 6, provides the basis of such an algorithm.

▶ **Lemma 4.5.** *Let ξ be a realizable boundary condition on $\partial\Lambda_{n,l}$ that is free on $\partial_{\mathrm{S}}\Lambda_{n,l} \cup \partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$ and free in all vertices in $\partial_{\mathrm{N}}\Lambda_{n,l}$ at distance at most m from $\partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$. For every group of rectangles $\mathcal{R} = \bigcup_{i=1}^{N(\mathcal{R})} R_i$ compatible with ξ, with $W(\mathcal{R}) \geq 100m$, there exists a disconnecting interval $[\![c_\star, d_\star]\!]$ such that $(c_\star, l), (d_\star, l) \in \partial_{\mathrm{N}}\mathcal{R}$, are distance at least m from the vertical sides $\bigcup_{i=1}^{N(\mathcal{R})} \partial_{\mathrm{W}}R_i \cup \partial_{\mathrm{E}}R_i$ of $\mathcal{R}$, and*

$$\frac{1}{4}W(\mathcal{R}) \leq W(\mathcal{R} \cap ([\![c_\star, d_\star]\!] \times [\![0,l]\!])) \leq \frac{3}{4}W(\mathcal{R})\,.$$

With the disconnecting interval $[\![c_\star, d_\star]\!]$ from Lemma 4.5, we define $\mathcal{A}_{\mathrm{INT}} = \mathcal{R} \cap ([\![c_\star, d_\star]\!] \times [\![0,l]\!])$ and $\mathcal{A}_{\mathrm{EXT}} = \mathcal{R} \cap ([\![c_\star, d_\star]\!]^c \times [\![0,l]\!])$. Their enlargements by m will form the blocks $\mathcal{R}_{\mathrm{INT}}$ and $\mathcal{R}_{\mathrm{EXT}}$:

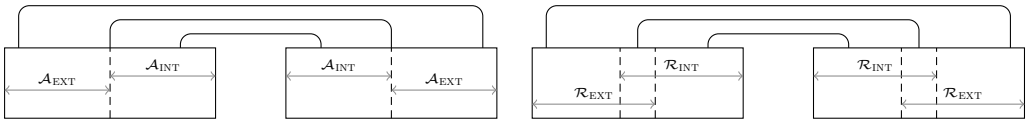
$$\mathcal{R}_{\mathrm{INT}} = \mathcal{R} \cap ([\![c_\star - m, d_\star + m]\!] \times [\![0,l]\!]) \quad \text{and} \quad \mathcal{R}_{\mathrm{EXT}} = \mathcal{R} \cap (([\![0, c_\star + m]\!] \cup [\![d_\star - m, n]\!]) \times [\![0,l]\!])\,;$$

These sets are depicted in Figure 2(a)–(b). The requirement that the corners of $[\![c_\star, d_\star]\!] \times [\![0,l]\!]$ are a distance at least m from the vertical sides of $\mathcal{R}$ is so that when we enlarge the sets $\mathcal{A}_{\mathrm{INT}}, \mathcal{A}_{\mathrm{EXT}}$ by m, we do not overflow beyond the rectangles containing $(c_\star, l)$ and $(d_\star, l)$. Crucially, our ability to pick disconnecting segments that satisfy this requirement is guaranteed by the compatibility of $\mathcal{R}$ with ξ.

It follows from Lemma 4.5, and the definitions of disconnecting interval and compatibility, that $\mathcal{R}_{\mathrm{INT}}$ and $\mathcal{R}_{\mathrm{EXT}}$ have the following properties, which will facilitate our recursive argument to prove Theorem 4.1.

▶ **Proposition 4.6.** *If $\mathcal{R}$ is a group of rectangles compatible with ξ, and moreover, $W(\mathcal{R}) \geq 100m$, then the sets $\mathcal{R}_{\mathrm{INT}}$ and $\mathcal{R}_{\mathrm{EXT}}$ are groups of rectangles satisfying the following properties:*

1. *$\frac{1}{5}W(\mathcal{R}) \leq W(\mathcal{R}_{\mathrm{INT}}) \leq \frac{4}{5}W(\mathcal{R})$ and likewise $\frac{1}{5}W(\mathcal{R}) \leq W(\mathcal{R}_{\mathrm{EXT}}) \leq \frac{4}{5}W(\mathcal{R})$;*
2. *Both $\mathcal{R}_{\mathrm{INT}}$ and $\mathcal{R}_{\mathrm{EXT}}$ are compatible with ξ.*



(a) **(b)**

Figure 2 **(a)** The cores $\mathcal{A}_{\text{INT}}$ and $\mathcal{A}_{\text{EXT}}$ and **(b)** the blocks $\mathcal{R}_{\text{INT}}$ and $\mathcal{R}_{\text{EXT}}$. The blocks $\mathcal{R}_{\text{INT}}$ and $\mathcal{R}_{\text{EXT}}$ are the enlargements of $\mathcal{A}_{\text{INT}}$ and $\mathcal{A}_{\text{EXT}}$ by exactly m, and are thus, themselves, groups of rectangles.

Finally, we consider the spectral gap of the block dynamics $\{X_t\}$ on $\mathcal{R}$ with blocks $\mathscr{B} = \{\mathcal{R}_{\text{INT}}, \mathcal{R}_{\text{EXT}}\}$. In this case, $\{X_t\}$ is the discrete-time Markov chain that in each step picks i uniformly at random from $\{\text{INT}, \text{EXT}\}$ and updates the configuration in $E(\mathcal{R}_i)$ with a sample from the stationary distribution of the chain conditional on the configuration on $E^c(\mathcal{R}_i)$. Let $\text{gap}(\mathcal{R}^\zeta; \mathscr{B})$ be the spectral gap of this block dynamics on the group of rectangle $\mathcal{R}$ with boundary condition ζ induced on $\mathcal{R}$ by ξ and a fixed random-cluster configuration $\omega_{\mathcal{R}^c}$ on $E^c(\mathcal{R}) = E(\Lambda_{n,l}) \setminus E(\mathcal{R})$; hence, we may identify ζ with the pair $(\xi, \omega_{\mathcal{R}^c})$.

▶ **Lemma 4.7.** *Let ξ be a realizable boundary condition on $\partial\Lambda_{n,l}$ that is free on $\partial_{\text{S}}\Lambda_{n,l} \cup \partial_{\text{E}}\Lambda_{n,l} \cup \partial_{\text{W}}\Lambda_{n,l}$ and free on vertices in $\partial_{\text{N}}\Lambda_{n,l}$ at distance at most m from $\partial_{\text{E}}\Lambda_{n,l} \cup \partial_{\text{W}}\Lambda_{n,l}$. For every $q > 1$ and $p \neq p_c(q)$, there exists $K = K(p,q) \geq 1$ such that for every group of rectangles $\mathcal{R}$ compatible with ξ, and every configuration $\omega_{\mathcal{R}^c}$ on $E^c(\mathcal{R})$, we have $\text{gap}(\mathcal{R}^{(\xi,\omega(\mathcal{R}^c))}; \mathscr{B}) \geq K^{-1}$.*

The proof of Lemma 4.7 is deferred to Section 6. We are now ready to prove Theorem 4.1.

Proof of Theorem 4.1. Fix $q > 1$, $p \neq p_c(q)$ and $\Lambda_{n,l}$ with a realizable boundary condition ξ' that is *free* on $\partial_{\text{E}}\Lambda_{n,l} \cup \partial_{\text{S}}\Lambda_{n,l} \cup \partial_{\text{W}}\Lambda_{n,l}$. We modify ξ' to a boundary condition ξ that is also free on all vertices a distance at most $m = C_\star \log l$ from $\partial_{\text{E}}\Lambda_{n,l} \cup \partial_{\text{W}}\Lambda_{n,l}$ at a cost of an exponential in m factor in the mixing time of the FK-dynamics (see Lemma 2.3 in [3]). Let ξ be the resulting realizable boundary condition.

Let $\mathcal{R} \subset \Lambda_{n,l}$ be a group of rectangles that is compatible with ξ and has $W(\mathcal{R}) = s$ for $100m \leq s \leq n$. Let $\mathcal{R}_{\text{INT}}$ and $\mathcal{R}_{\text{EXT}}$ be the group rectangles defined earlier and consider the block dynamics with respect to these blocks. Recall that we use $\text{gap}(\mathcal{R}^\zeta)$ and $\text{gap}(\mathcal{R}^\zeta; \mathscr{B})$ for the spectral gaps of the FK-dynamics and the blocks dynamics with respect to $\mathscr{B} = \{\mathcal{R}_{\text{INT}}, \mathcal{R}_{\text{EXT}}\}$ respectively. As discussed earlier, for any boundary condition $\zeta = (\xi, \omega^c_{\mathcal{R}})$, Proposition 3.4 from [27] implies that for a suitable constant $\gamma \in (0,1)$

$$\text{gap}\left(\mathcal{R}^\zeta\right) \geq \gamma \cdot \text{gap}\left(\mathcal{R}^\zeta; \mathscr{B}\right) \cdot \min_{\substack{i \in \{\text{INT},\text{EXT}\} \\ \omega \in \Omega(R_i^c)}} \text{gap}\left(\mathcal{R}_i^{(\xi,\omega)}\right) \geq \frac{\gamma}{K} \cdot \min_{\substack{i \in \{\text{INT},\text{EXT}\} \\ \omega \in \Omega(R_i^c)}} \text{gap}\left(\mathcal{R}_i^{(\xi,\omega)}\right), \quad (4)$$

where the second inequality follows from Lemma 4.7 and $\Omega(R_i^c)$ is the set of FK configurations on $E(R_i^c)$. Observe that Proposition 4.6 implies that $\max\{W(\mathcal{R}_{\text{INT}}), W(\mathcal{R}_{\text{EXT}})\} \leq 4s/5$. Therefore, applying (4) $O(\log n)$ times, we deduce that $\text{gap}(\mathcal{R}^\zeta) \geq \exp(\Omega(-\log n)) \cdot \text{gap}(\mathcal{R}_0^{\zeta_0})$, where $\mathcal{R}_0$ is a group of rectangles with $W(\mathcal{R}_0) \leq 100m$ and $\zeta_0 = (\xi, \omega_0)$ is an arbitrary boundary condition for $\mathcal{R}_0$.

Finally, since $|\partial\mathcal{R}_0| = O(m + l) = O(l)$, the lower bound for $\text{gap}(\mathcal{R}_0^{\zeta_0})$ follows from the following crude argument. Observe that we can first modify the boundary condition ζ_0 to be all free on all of $\partial\mathcal{R}_0$, incurring a cost of a $q^{-\Omega(l)}$ factor in the spectral gap; see Lemma 2.3 in [3]. The fast mixing result from [5] for the free boundary condition then implies that $\text{gap}(\mathcal{R}_0^{\zeta_0}) \geq \exp(-\Omega(l))$ and so the result follows. ◀

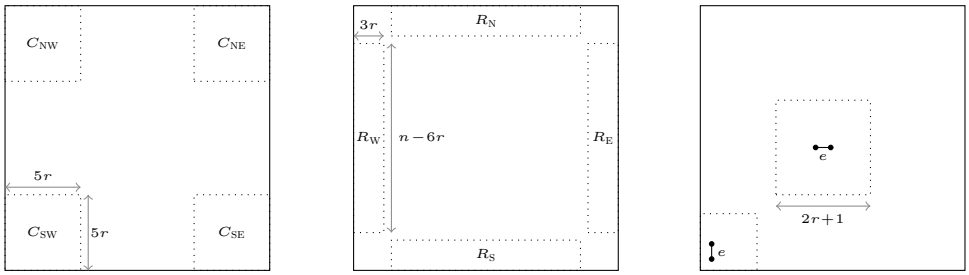


Figure 3 Subsets (a) C_{NE}, C_{NW}, C_{SE}, and C_{SW} and (b) R_{N}, R_{E}, R_{W} and R_{S}. **(c)** $B(e,r)$ for two edges e of Λ_n.

5 Polynomial mixing time for realizable boundary conditions

In this section we finalize the proof of Theorem 1.1 for $p < p_c(q)$ using the technology introduced in Section 3; namely, we construct a collection of subsets $\mathcal{B}$ for which we can establish LM and MSM; see Definitions 3.1–3.3. To establish LM we crucially use Theorem 4.1. The results for $p > p_c(q)$ follow from the self-duality of the model and of realizable boundary conditions, as explained in Section 2.

For general realizable boundary conditions, proving LM for a collection of subsets $\mathcal{B}$ for which MSM holds is the main challenge. This is because, for MSM to hold for a collection $\mathcal{B}$ for all realizable boundary conditions, a subset in $\mathcal{B}$ needs to contain $\Omega(n)$ edges. In particular, some element of $\mathcal{B}$ must include most (or all) edges near $\partial\Lambda_n$, as otherwise it is easy to construct examples of realizable boundary conditions for which MSM does not hold. Thus, a trivial (exponential in the perimeter) upper bound for the mixing time on those subsets with $\Omega(n)$ edges would be unhelpful, and we ought to use Theorem 4.1 instead.

We define a collection of blocks for which we can establish both LM and MSM. Let $r \in \mathbb{N}$ and let $C_{\mathrm{NE}}, C_{\mathrm{NW}}, C_{\mathrm{SE}}, C_{\mathrm{SW}} \subset \Lambda_n$ be the four square boxes of side length $5r$ with a corner that coincides with a corner of Λ_n; see Figure 3(a). Let $R_{\mathrm{N}} \subset \Lambda_n$ be the $(n-6r) \times 2r$ rectangle at distance $3r$ from both $\partial_{\mathrm{W}}\Lambda_n$, $\partial_{\mathrm{E}}\Lambda_n$ whose top boundary is contained in $\partial_{\mathrm{N}}\Lambda_n$ and let $R_{\mathrm{E}}, R_{\mathrm{W}}, R_{\mathrm{S}}$ be defined analogously; see Figure 3(b). Let $R = R_{\mathrm{N}} \cup R_{\mathrm{E}} \cup R_{\mathrm{W}} \cup R_{\mathrm{S}}$. Now, for $e \in E(\Lambda_n)$, let $B(e,r) \subset \Lambda_n$ be the set of vertices in the minimal square box around e such that $d(\{e\}, \Lambda_n \setminus B(e,r)) \geq r$. If $d(\{e\}, \partial\Lambda_n) > r$, then $B(e,r)$ is just a square box of side length $2r+1$ centered at e; otherwise $B(e,r)$ intersects $\partial\Lambda_n$; see Figure 3(c). Finally, let

$$\mathcal{B}_r = \{C_{\mathrm{NE}}, C_{\mathrm{NW}}, C_{\mathrm{SE}}, C_{\mathrm{SW}}, R\} \cup \{B(e,r) : e \in E(\Lambda_n), d(\{e\}, \partial\Lambda_n) > r\}. \tag{5}$$

We claim that LM holds for $\mathcal{B}_r$ with $r = \Theta(\log n)$ and $T = O(n^C)$ for some constant $C > 0$.

▶ **Theorem 5.1.** *Let $q \geq 1$, $p < p_c(q)$ and $r = c_0 \log n$ with $c_0 > 0$ independent of n. There exists a constant $C > 0$ such that LM holds for every realizable boundary condition ξ and $\mathcal{B}_r$ with $T = O(n^C)$.*

The subsets $B(e,r)$ in $\mathcal{B}_r$ and the corner boxes C_{NE}, C_{NW}, C_{SE} and C_{SW} are small enough that crude bounds for their mixing times are sufficient. As mentioned earlier, the main challenge for proving local mixing for $\mathcal{B}_r$ is to derive a mixing time bound for $R = R_{\mathrm{N}} \cup R_{\mathrm{E}} \cup R_{\mathrm{W}} \cup R_{\mathrm{S}}$ as it intersects the boundary of Λ_n and contains $\Omega(n)$ vertices. To establish such a bound we rely on Theorem 4.1. In particular, we relate the mixing time of the FK-dynamics on R

to that of the FK-dynamics on a single thin rectangle by concatenating the four rectangles constituting R, one after another, such that the union of their outer boundaries make up the northern boundary of the new rectangle.

The final ingredient of the proof is establishing MSM for the collection $\mathcal{B}_r$. We show that MSM holds for $\mathcal{B}_r$ with $r = \Theta(\log n)$ for all realizable boundary conditions ξ where the vertices in $\partial\Lambda_n$ at distance $5r$ from the corners of Λ_n are free in ξ. This is sufficient since any realizable boundary condition can be turned into a realizable boundary condition with this property by simply removing all connections in ξ involving vertices near the corners of Λ_n; this modification can change the mixing time of the FK-dynamics by a factor of at most $\exp(O(r))$; see Lemma 2.3 in [3].

▶ **Theorem 5.2.** *Let $q \geq 1$, $p < p_c(q)$ and $r = c_0 \log n$ with $c_0 > 0$ independent of n. Let ξ be a realizable boundary condition with the property that every vertex $v \in \partial\Lambda_n$ at distance at most $5r$ from a corner of Λ_n is free in ξ. Then, for all sufficiently large $c_0 > 0$, MSM holds for ξ and $\mathcal{B}_r$ with $\delta < 1/(12|E(\Lambda_n)|)$.*

Theorem 1.1 follows from Theorems 3.5 and 5.1–5.2. Their proofs are found in the full version [3].

6 Proofs from Section 4

We prove here the two key results from Section 4: Lemmas 4.5 and 4.7. We begin with the proof of Lemma 4.5 which describes how to find an appropriate disconnecting interval for the block dynamics in a group of rectangles $\mathcal{R}$. The proof uses the following important geometric observations regarding disconnecting intervals. For more details we refer to [3].

▶ **Lemma 6.1.** *Let ξ be a realizable boundary condition on $\Lambda_{n,l}$ that is free on $\partial_{\mathrm{S}}\Lambda_{n,l} \cup \partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$ and let $a < b < c$. If both $[\![a, b]\!]$ and $[\![b, c]\!]$ are disconnecting intervals of wired-type, then so is $[\![a, c]\!]$. If both $[\![a, b]\!]$ and $[\![b+1, c]\!]$ are disconnecting intervals of free-type, then so is $[\![a, c]\!]$.*

▶ **Lemma 6.2.** *Let ξ be a realizable boundary condition on $\Lambda_{n,l}$ that is free on $\partial_{\mathrm{S}}\Lambda_{n,l} \cup \partial_{\mathrm{E}}\Lambda_{n,l} \cup \partial_{\mathrm{W}}\Lambda_{n,l}$. Suppose there exist $a < b < c < d$ such that $[\![a, c]\!]$ and $[\![b, d]\!]$ are disconnecting intervals; then either both are of free-type or both are of wired-type. Moreover, if both are*

1. *of* wired-type*: then $[\![a, b]\!], [\![b, c]\!], [\![c, d]\!]$ and $[\![a, d]\!]$ are all disconnecting intervals of wired-type.*
2. *of* free-type*: then $[\![a, b-1]\!], [\![b, c]\!], [\![c+1, d]\!]$ and $[\![a, d]\!]$ are all disconnecting intervals of free-type.*

Proof of Lemma 4.5. We find a candidate disconnecting interval $[\![c, d]\!]$ with $(c, l), (d, l) \in \partial_{\mathrm{N}}\mathcal{R}$ satisfying:

$$\tfrac{1}{3}W(\mathcal{R}) \leq W(\mathcal{R} \cap ([\![c, d]\!] \times [\![0, l]\!])) \leq \tfrac{2}{3}W(\mathcal{R})\,. \tag{6}$$

In the second part of the proof we show how to modify $[\![c, d]\!]$ to obtain a disconnecting interval $[\![c_\star, d_\star]\!]$ with the added property that both $(c_\star, l)$ and $(d_\star, l)$ are distance at least m from $\partial_{\|}\mathcal{R} := \bigcup_{i=1}^{N(\mathcal{R})} \partial_{\mathrm{W}}R_i \cup \partial_{\mathrm{E}}R_i$.

If there exist vertices $(x, l), (y, l) \in \partial_{\mathrm{N}}\mathcal{R}$ such that $\frac{1}{3}W(\mathcal{R}) \leq W(\mathcal{R} \cap ([\![x, y]\!] \times [\![0, l]\!])) \leq \frac{2}{3}W(\mathcal{R})$ with (x, l) connected to (y, l) through ξ, then we take $c = x$, $d = y$ and use $[\![c, d]\!] = [\![x, y]\!]$ as our candidate disconnecting interval. Suppose otherwise that there does not

exist any such boundary connection: then every pair $(x,l),(y,l) \in \partial_{\mathrm{N}}\mathcal{R}$ connected through ξ is such that

$$W(\mathcal{R} \cap (\llbracket x, y \rrbracket \times \llbracket 0, l \rrbracket)) < \tfrac{1}{3} W(\mathcal{R})\,, \qquad \text{or} \qquad W(\mathcal{R} \cap (\llbracket x, y \rrbracket \times \llbracket 0, l \rrbracket)) > \tfrac{2}{3} W(\mathcal{R})\,. \tag{7}$$

If the latter holds, then there is a pair, say $(x_0,l),(y_0,l) \in \partial_{\mathrm{N}}\mathcal{R}$, for which the latter holds with a minimal width. Consequently, all other connections through ξ between vertices $(x_1,l),(y_1,l) \in \partial_{\mathrm{N}}\mathcal{R} \cap (\llbracket x_0+1, y_0-1 \rrbracket \times \llbracket 0, l \rrbracket)$ will be such that $W(\mathcal{R} \cap (\llbracket x_1, y_1 \rrbracket \times \llbracket 0, l \rrbracket)) < \frac{1}{3} W(\mathcal{R})$. We can then partition the vertices of $\partial_{\mathrm{N}}\mathcal{R} \cap (\llbracket x_0+1, y_0-1 \rrbracket \times \{l\})$ into disjoint disconnecting intervals of free-wired-type as follows:

1. Let $\rho = \{C_1, \ldots, C_k\}$ be the partition of $\partial_{\mathrm{N}}\mathcal{R} \cap (\llbracket x_0+1, y_0-1 \rrbracket \times \{l\})$ induced by ξ;
2. For each C_i, consider the disconnecting interval L_i of free-wired-type determined by the left-most and right-most vertices of C_i in $\partial_{\mathrm{N}}\mathcal{R} \cap (\llbracket x_0+1, y_0-1 \rrbracket \times \{l\})$ (n.b. these may be singletons);
3. Let $\{L_{i_1}, ..., L_{i_\ell}\}$ be those which are maximal, i.e., there does not exist j and k such that $L_{i_j} \subset L_k$.

The set of disconnecting intervals $\{L_{i_1}, \ldots, L_{i_\ell}\}$ partitions $\llbracket x_0+1, y_0-1 \rrbracket$ into disjoint disconnecting intervals of free-wired-type with the property that $W(\mathcal{R} \cap (L_{i_j} \times \llbracket 0, l \rrbracket)) \leq \frac{1}{3} W(\mathcal{R})$ for every $j \in \{1, \ldots, \ell\}$. We can then use Lemma 6.1 to merge adjacent disconnecting intervals until we obtain a candidate disconnecting interval $\llbracket c, d \rrbracket \subset \llbracket x_0, y_0 \rrbracket$ (of free-type), having width $W(\mathcal{R} \cap (\llbracket c, d \rrbracket \times \llbracket 0, l \rrbracket) \in [\frac{1}{3} W(\mathcal{R}), \frac{2}{3} W(\mathcal{R})]$.

Now that we have found a candidate disconnecting interval $\llbracket c, d \rrbracket$ satisfying (6), we modify it to obtain a disconnecting interval $\llbracket c_\star, d_\star \rrbracket$ with the property that both $(c_\star, l)$, $(d_\star, l)$ are distance at least m from $\partial_{\|}\mathcal{R}$.

If (c,l) is at distance at least m from $\partial_{\|}\mathcal{R}$, set $c_\star = c$, and similarly if (d,l) is at distance at least m from $\partial_{\|}\mathcal{R}$, then set $d_\star = d$. Otherwise, suppose (c,l) is at distance less than m from $\partial_{\mathrm{W}} R_i$ for some constituent rectangular subset $R_i = \llbracket a_i, b_i \rrbracket \times \llbracket 0, l \rrbracket$ of $\mathcal{R}$. Since $\mathcal{R}$ is compatible with ξ, the interval $\mathcal{I}_c = \llbracket b_{i-1} - m, a_i + m \rrbracket$ is a disconnecting interval, and we set

$$c_\star = \begin{cases} a_i + m\,, & \text{if } \mathcal{I}_c \text{ is of wired-type, or } i = 1, \text{ or } W(R_i) = 2m; \\ a_i + m + 1\,, & \text{if } \mathcal{I}_c \text{ is only of free-type, and } W(R_i) > 2m; \end{cases} .$$

When (c,l) is at distance less than m from $\partial_{\mathrm{E}} R_i$ for some i, then we simply set $c_\star = b_i - m$. Symmetrically, if (d,l) is at distance less than m from $\partial_{\mathrm{E}} R_i$ for $R_i = \llbracket a_i, b_i \rrbracket \times \llbracket 0, l \rrbracket$, let $\mathcal{I}_d = \llbracket b_i - m, a_{i+1} + m \rrbracket$

$$d_\star = \begin{cases} b_i - m\,, & \text{if } \mathcal{I}_d \text{ is of wired-type, or } i = N(\mathcal{R}), \text{ or } W(R_i) = 2m; \\ b_i - m - 1\,, & \text{if } \mathcal{I}_d \text{ is only of free-type, and } W(R_i) > 2m. \end{cases}$$

When (d,l) is at distance less than m from $\partial_{\mathrm{W}} R_i$, let $d_\star = a_i + m$. Since $W(R_i) \geq 2m$ for every i, the points $(c,l),(d,l)$ cannot be both less than m away from $\partial_{\mathrm{E}} R_i$ and less than m away from $\partial_{\mathrm{W}} R_i$.

One can check via a case analysis, exploiting the compatibility of $\mathcal{R}$ with ξ and using Lemma 6.2, that in all of these cases the interval $\llbracket c_\star, d_\star \rrbracket$ is a disconnecting interval; we defer these details to the full manuscript [3]. The fact that $(c_\star, l), (d_\star, l) \in \partial_{\mathrm{N}}\mathcal{R}$ are a distance at least m away from $\partial_{\|}\mathcal{R}$ follows directly from the construction. Finally, we claim that in all such situations, $\llbracket c_\star, d_\star \rrbracket$ satisfies

$$\tfrac{1}{4} W(\mathcal{R}) \leq W(\mathcal{R} \cap (\llbracket c_\star, d_\star \rrbracket \times \llbracket 0, l \rrbracket)) \leq \tfrac{3}{4} W(\mathcal{R})\,.$$

This follows from the facts that $W(\mathcal{R}) \geq 100m$, $|c - c_\star| \leq m$ and $|d - d_\star| \leq m$. ◀



Figure 4 **(a)** The block $\mathcal{R}_{\text{INT}}$ with its subsets $\mathcal{A}_{\text{INT}}$, Q_{W} and Q_{E}. **(b)** The block $\mathcal{R}_{\text{INT}}$ with the dual-paths (dotted) of a configuration in Γ allowing coupling inside $\mathcal{A}_{\text{INT}}$.

We proceed with the proof of Lemma 4.7, where we establish a lower bound for the spectral gap of the block dynamics in the group of rectangles $\mathcal{R}$ with blocks $\mathcal{R}_{\text{INT}}$ and $\mathcal{R}_{\text{EXT}}$ as defined in Section 4.1.

Proof of Lemma 4.7. We consider the $p < p_c(q)$ case; the case of $p > p_c(q)$ follows from a similar (dual) argument which we defer to the full manuscript [3]. Let $\{X_t\}, \{Y_t\}$ be two instances of the block dynamics on $\mathcal{R}$ with boundary condition $\zeta = (\xi, \omega_{\mathcal{R}^c})$ started from initial configurations X_0, Y_0. It suffices to construct a coupling $\mathbb{P}$ of the steps of $\{X_t\}, \{Y_t\}$ such that $\min_{X_0,Y_0} \mathbb{P}(X_2 = Y_2) = \Omega(1)$; see [24].

With probability $1/4$ the first block to be updated is $\mathcal{R}_{\text{INT}}$ and the second is $\mathcal{R}_{\text{EXT}}$. Suppose this is the case and let us consider the update on $\mathcal{R}_{\text{INT}}$. Let θ_ω be the boundary condition on $\partial\mathcal{R}_{\text{INT}}$ induced by ζ and the restriction of a configuration ω to $E(\mathcal{R}) \setminus E(\mathcal{R}_{\text{INT}})$, and let π^{θ_ω} be the FK distribution on $\mathcal{R}_{\text{INT}}$ with boundary conditions θ_ω. As $X_1(\mathcal{R}_{\text{INT}}), Y_1(\mathcal{R}_{\text{INT}})$ have laws $\pi^{\theta_{X_0}}$, $\pi^{\theta_{Y_0}}$, respectively, a coupling for $\pi^{\theta_{X_0}}$ and $\pi^{\theta_{Y_0}}$ yields a coupling for X_1 and Y_1.

We describe next such a coupling for $\pi^{\theta_{X_0}}$ and $\pi^{\theta_{Y_0}}$. Let $Q_{\text{W}}, Q_{\text{E}} \subset \mathcal{R}_{\text{INT}}$ be the two rectangles of width m that contain all the vertices in $\mathcal{R}_{\text{INT}} \setminus \mathcal{A}_{\text{INT}}$; i.e., $Q_{\text{W}} \cup \mathcal{A}_{\text{INT}} \cup Q_{\text{E}} = \mathcal{R}_{\text{INT}}$, $Q_{\text{W}} \cap \mathcal{A}_{\text{INT}} = \emptyset$ and $Q_{\text{E}} \cap \mathcal{A}_{\text{INT}} = \emptyset$ (see Figure 4(a)). Let $\partial E(Q_{\text{W}})$ be the set of edges with one endpoint in Q_{W} and the other in $\mathcal{A}_{\text{INT}}$, and similarly define $\partial E(Q_{\text{E}})$. Let Γ_{W} be the set of configurations in $\mathcal{R}_{\text{INT}}$ that have a *dual-path* in $E(Q_{\text{W}}) \cup \partial E(Q_{\text{W}})$ connecting the top-most edge in $\partial E(Q_{\text{W}})$ to an edge in $\partial_{\text{S}} Q_{\text{W}}$, and similarly define Γ_{E} as the set of configurations in $\mathcal{R}_{\text{INT}}$ that have a dual-path in $E(Q_{\text{E}}) \cup \partial E(Q_{\text{E}})$ from the top-most edge in $\partial E(Q_{\text{E}})$ to an edge in $\partial_{\text{S}} Q_{\text{E}}$. (A dual-path is an open path in the dual configuration.) Let $\Gamma = \Gamma_{\text{E}} \cap \Gamma_{\text{W}}$; see Figure 4(b). Let θ_1 be the boundary condition on $\partial\mathcal{R}_{\text{INT}}$ induced by ζ and the wired configuration on $E(\mathcal{R}) \setminus E(\mathcal{R}_{\text{INT}})$. The following lemma supplies the desired coupling.

▶ **Lemma 6.3.** *Let $q > 1$ and $p < p_c(q)$. There exists a coupling $\mathbb{P}_1$ of the distributions $\pi^{\theta_{X_0}}$, $\pi^{\theta_{Y_0}}$, π^{θ_1} such that if $(\omega^{\theta_X}, \omega^{\theta_Y}, \omega^{\theta_1})$ is sampled from $\mathbb{P}_1$, the following hold:*

1. $\mathbb{P}_1(\omega^{\theta_X}(\mathcal{A}_{\text{INT}}) = \omega^{\theta_Y}(\mathcal{A}_{\text{INT}}) \mid \omega^{\theta_1} \in \Gamma) = 1$;
2. *There exists a constant $\rho = \rho(p, q) > 0$ such that $\mathbb{P}_1(\omega^{\theta_1} \in \Gamma) \geq \rho$.*

If we use the coupling $\mathbb{P}_1$ from Lemma 6.3 to couple the first step of the chains, then X_1 and Y_1 will agree on $E(\mathcal{A}_{\text{INT}})$ with probability at least $\rho > 0$. If this occurs, then we can couple the update on $\mathcal{R}_{\text{EXT}}$ in the second step so that $X_2 = Y_2$ with probability one. This is because $X_1(E(\mathcal{A}_{\text{INT}})) = Y_1(E(\mathcal{A}_{\text{INT}}))$ implies $X_1(E(\mathcal{R}) \setminus E(\mathcal{R}_{\text{EXT}})) = Y_1(E(\mathcal{R}) \setminus E(\mathcal{R}_{\text{EXT}}))$, and thus the boundary conditions induced by the two instances of the chain on $\mathcal{R}_{\text{EXT}}$ are identical. As a consequence, we obtain that for any X_0, Y_0, $\mathbb{P}(X_2 = Y_2) \geq \rho/4$, which concludes the proof for $p < p_c(q)$. ◀

We conclude this section with the proof of Lemma 6.3.

Proof of Lemma 6.3. Let $L = \partial_{\mathrm{W}} Q_{\mathrm{W}} \cup \partial_{\mathrm{N}} Q_{\mathrm{W}} \cup \partial_{\mathrm{E}} Q_{\mathrm{E}} \cup \partial_{\mathrm{N}} Q_{\mathrm{E}}$. For a configuration ω on $\mathcal{R}_{\mathrm{INT}}$ let $F(\omega) := \mathcal{R}_{\mathrm{INT}} \setminus \bigcup_{v \in L} C(v, \omega)$, where $C(v, \omega)$ is the vertex set of the connected component of v in ω, ignoring the boundary connections. Note that $\omega \in \Gamma$ if and only if the vertices in the boundary components of L, i.e., $\bigcup_{v \in L} C(v, \omega)$, are confined to $Q_{\mathrm{W}} \cup Q_{\mathrm{E}}$, in which case $\mathcal{A}_{\mathrm{INT}} \subseteq F(\omega)$.

Clearly, $\pi^{\theta_1} \succeq \pi^{\theta_X}$ and $\pi^{\theta_1} \succeq \pi^{\theta_Y}$ and thus there exist monotone couplings $\mathbb{P}_X$ (resp., $\mathbb{P}_Y$) for π^{θ_X} and π^{θ_1} (resp., π^{θ_Y} and π^{θ_1}). The coupling $\mathbb{P}_1$ is defined as follows. First sample $(\omega^{\theta_X}, \omega^{\theta_1})$ from $\mathbb{P}_X$ and ω^{θ_Y} from $\mathbb{P}_Y(\,\cdot \mid \omega^{\theta_1})$. If $\mathcal{A}_{\mathrm{INT}} \subseteq F(\omega^{\theta_1})$, then re-sample the configuration on $E(F(\omega^{\theta_1}))$ in ω^{θ_1} and update $\omega^{\theta_X}(F(\omega^{\theta_1}))$ and $\omega^{\theta_Y}(F(\omega^{\theta_1}))$ such that $\omega^{\theta_1}(F(\omega^{\theta_1})) = \omega^{\theta_X}(F(\omega^{\theta_1})) = \omega^{\theta_Y}(F(\omega^{\theta_1}))$.

To deduce part 1, it now suffices to show that if $\mathcal{A}_{\mathrm{INT}} \subseteq F(\omega^{\theta_1})$ the three boundary conditions η_1, η_X, η_Y induced on $\partial F(\omega^{\theta_1})$ by the configurations of ω^{θ_X}, ω^{θ_Y}, ω^{θ_1} on $E(\mathcal{R}_{\mathrm{INT}}) \setminus E(F(\omega^{\theta_1}))$, respectively, and the corresponding boundary conditions θ_X, θ_Y, θ_1 are identical; if this is the case part 1 follows from the domain Markov property (see [21]).

First observe that the boundary condition on $\partial_{\mathrm{S}} \mathcal{A}_{\mathrm{INT}}$ is always free by assumption. Also from the definition of $F(\omega^{\theta_1})$ every edge of $E(\mathcal{R}_{\mathrm{INT}}) \setminus E(F(\omega^{\theta_1}))$ incident to $\partial F(\omega^{\theta_1})$ is closed in ω^{θ_1}, so by the monotonicity of the coupling, the same holds for ω^{θ_X} and ω^{θ_Y}. The remaining portion of $\partial F(\omega^{\theta_1})$ is precisely the set of vertices $(\partial \mathcal{A}_{\mathrm{INT}} \cap \partial \mathcal{R}) \setminus \partial_{\mathrm{S}} \mathcal{R}$. In order for the boundary conditions η_1, η_x, η_Y to differ on this set, there must be at least two distinct boundary components in $\zeta = (\xi, \omega_{\mathcal{R}^c})$ between $(\partial \mathcal{A}_{\mathrm{INT}} \cap \partial \mathcal{R}) \setminus \partial_{\mathrm{S}} \mathcal{R}$ and $[\![c_\star, d_\star]\!]^c \times \{l\}$; this cannot happen because $[\![c_\star, d_\star]\!]$ is disconnecting.

Part 2 of the lemma is a straightforward consequence of the EDC property of the random-cluster model at $p < p_c(q)$; see (2). Namely, since the width of Q_{W} is $m = C_\star \log l$, (2) implies that when $C_\star$ is large enough there is a constant $\rho_{\mathrm{W}}(p, q) > 0$ such that $\pi^{\theta_1}(\Gamma_{\mathrm{W}}^{\mathrm{c}}) \le \pi^1_\Delta(L \overset{\Delta}{\longleftrightarrow} \partial_{\mathrm{W}} \mathcal{A}_{\mathrm{INT}}) \le 1 - \rho_{\mathrm{W}}$, where Δ is the subgraph induced by the edges in $E(\mathcal{R}_{\mathrm{INT}}) \setminus E(\mathcal{A}_{\mathrm{INT}})$. A matching bound holds for $\Gamma_{\mathrm{E}}^{\mathrm{c}}$. Since Γ_{E}, Γ_{W} are both decreasing events, by the FKG inequality (see [21]), $\pi^{\theta_1}(\Gamma) \ge \rho_{\mathrm{W}} \rho_{\mathrm{E}} =: \rho$, concluding the proof. ◀

References

1 Kenneth S. Alexander. On weak mixing in lattice models. *Probab. Theory Related Fields*, 110(4):441–471, 1998. doi:10.1007/s004400050155.

2 Vincent Beffara and Hugo Duminil-Copin. The self-dual point of the two-dimensional random-cluster model is critical for $q \ge 1$. *Probab. Theory Related Fields*, 153(3-4):511–542, 2012. doi:10.1007/s00440-011-0353-8.

3 Antonio Blanca, Reza Gheissari, and Eric Vigoda. Random-cluster dynamics in $\mathbb{Z}^2$: rapid mixing with general boundary conditions. *preprint available at* arXiv:1807.08722., 2018.

4 Antonio Blanca and Alistair Sinclair. Dynamics for the Mean-field Random-cluster Model. In *Proc. of the 19th International Workshop on Randomization and Computation (RANDOM 2015)*, pages 528–543, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.528.

5 Antonio Blanca and Alistair Sinclair. Random-Cluster Dynamics in $\mathbb{Z}^2$. *Probab. Theory Related Fields*, 2016. Extended abstract appeared in Proc. of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2016), pp. 498–513. doi:10.1007/s00440-016-0725-1.

6 Christian Borgs, Jennifer T. Chayes, Alan Frieze, Jeong H. Kim, Prasad Tetali, Eric Vigoda, and Van H. Vu. Torpid mixing of some Monte Carlo Markov chain algorithms in statistical physics. In *Proc. of the 40th Annual Symposium on Foundations of Computer Science (FOCS 1999)*, pages 218–229, 1999. doi:10.1109/SFFCS.1999.814594.

7 Christian Borgs, Jennifer T. Chayes, and Prasad Tetali. Tight bounds for mixing of the Swendsen-Wang algorithm at the Potts transition point. *Probab. Theory Related Fields*, 152(3-4):509–557, 2012. doi:10.1007/s00440-010-0329-0.

8 Filippo Cesi. Quasi-factorization of the entropy and logarithmic Sobolev inequalities for Gibbs random fields. *Probability Theory and Related Fields*, 120(4):569–584, August 2001. doi:10.1007/PL00008792.

9 Lincoln Chayes and Jon Machta. Graphical representations and cluster algorithms I. Discrete spin systems. *Physica A: Statistical Mechanics and its Applications*, 239(4):542–601, 1997.

10 Colin Cooper and Alan M. Frieze. Mixing properties of the Swendsen–Wang process on classes of graphs. *Random Structures and Algorithms*, 15:242–261, 1999.

11 Hugo Duminil Copin, Maxim Gagnebin, Matan Harel, Ioan Manolescu, and Vincent Tassion. Discontinuity of the phase transition for the planar random-cluster and Potts models with $q > 4$. *CoRR*, 2016. arXiv:1611.09877.

12 Hugo Duminil-Copin, Vladas Sidoravicius, and Vincent Tassion. Continuity of the Phase Transition for Planar Random-Cluster and Potts Models with $1 \leq q \leq 4$. *Communications in Mathematical Physics*, 349(1):47–107, January 2017. doi:10.1007/s00220-016-2759-8.

13 Cornelius M. Fortuin and Pieter W. Kasteleyn. On the random-cluster model. I. Introduction and relation to other models. *Physica*, 57:536–564, 1972.

14 Andreas Galanis, Daniel Štefankovic, and Eric Vigoda. Swendsen-Wang Algorithm on the Mean-Field Potts Model. In *Proc. of the 19th International Workshop on Randomization and Computation (RANDOM 2015)*, pages 815–828, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.815.

15 Shirshendu Ganguly and Insuk Seo. Information Percolation and Cutoff for the Random-Cluster Model. *CoRR*, 2018. arXiv:1812.01538.

16 Reza Gheissari and Eyal Lubetzky. Mixing Times of Critical Two-Dimensional Potts Models. *Comm. Pure Appl. Math*, 71(5):994–1046, 2018.

17 Reza Gheissari and Eyal Lubetzky. The effect of boundary conditions on mixing of 2D Potts models at discontinuous phase transitions. *Electron. J. Probab.*, 23:30 pp., 2018. doi:10.1214/18-EJP180.

18 Reza Gheissari and Eyal Lubetzky. Quasi-polynomial mixing of critical two-dimensional random cluster models. *Random Structures and Algorithms*, 2019. doi:10.1002/rsa.20868.

19 Reza Gheissari, Eyal Lubetzky, and Yuval Peres. Exponentially slow mixing in the mean-field Swendsen–Wang dynamics. *Annales de l'Institut Henri Poincare (B)*, 2019. to appear. Extended abstract appeared in Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2018), pp. 1981–1988.

20 Vivek K. Gore and Mark R. Jerrum. The Swendsen-Wang process does not always mix rapidly. *J. Statist. Phys.*, 97(1-2):67–86, 1999. doi:10.1023/A:1004610900745.

21 Geoffrey Grimmett. The random-cluster model. In *Probability on discrete structures*, volume 110 of *Encyclopaedia Math. Sci.*, pages 73–123. Springer, Berlin, 2004. doi:10.1007/978-3-662-09444-0_2.

22 Heng Guo and Mark Jerrum. Random cluster dynamics for the Ising model is rapidly mixing. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1818–1827, 2017. doi:10.1137/1.9781611974782.118.

23 Mark Jerrum and Alistair Sinclair. Approximating the permanent. *SIAM J. Comput.*, 18(6):1149–1178, 1989. doi:10.1137/0218077.

24 David A. Levin, Malwina J. Luczak, and Yuval Peres. Glauber dynamics for the mean-field Ising model: cut-off, critical power law, and metastability. *Probab. Theory Related Fields*, 146(1-2):223–265, 2010. doi:10.1007/s00440-008-0189-z.

25 Eyal Lubetzky, Fabio Martinelli, Allan Sly, and Fabio Lucio Toninelli. Quasi-polynomial mixing of the 2D stochastic Ising model with "plus" boundary up to criticality. *J. Eur. Math. Soc. (JEMS)*, 15(2):339–386, 2013. doi:10.4171/JEMS/363.

26 Fabio Martinelli. On the two-dimensional dynamical Ising model in the phase coexistence region. *J. Statist. Phys.*, 76(5-6):1179–1246, 1994. doi:10.1007/BF02187060.

27 Fabio Martinelli. Lectures on Glauber dynamics for discrete spin models. In *Lectures on probability theory and statistics (Saint-Flour, 1997)*, volume 1717 of *Lecture Notes in Math.*, pages 93–191. Springer, Berlin, 1999. doi:10.1007/978-3-540-48115-7_2.

28 Fabio Martinelli, Enzo Olivieri, and Roberto H. Schonmann. For 2-D lattice spin systems weak mixing implies strong mixing. *Comm. Math. Phys.*, 165(1):33–47, 1994. URL: http://projecteuclid.org/euclid.cmp/1104271032.

29 Fabio Martinelli and Fabio Lucio Toninelli. On the mixing time of the 2D stochastic Ising model with "plus" boundary conditions at low temperature. *Comm. Math. Phys.*, 296(1):175–213, 2010. doi:10.1007/s00220-009-0963-5.

30 Elchanan Mossel and Allan Sly. Exact thresholds for Ising–Gibbs samplers on general graphs. *Ann. Probab.*, 41(1):294–328, January 2013. doi:10.1214/11-AOP737.

31 Lars Onsager. Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition. *Phys. Rev.*, 65:117–149, February 1944. doi:10.1103/PhysRev.65.117.

32 Alistair Sinclair. Improved bounds for mixing rates of Markov chains and multicommodity flow. *Combin. Probab. Comput.*, 1(4):351–370, 1992. doi:10.1017/S0963548300000390.

33 Robert H. Swendsen and Jian-Sheng Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Phys. Rev. Lett.*, 58:86–88, January 1987. doi:10.1103/PhysRevLett.58.86.

34 Lawrence E. Thomas. Bound on the mass gap for finite volume stochastic Ising models at low temperature. *Comm. Math. Phys.*, 126(1):1–11, 1989. URL: http://projecteuclid.org/euclid.cmp/1104179720.

35 Mario Ullrich. Comparison of Swendsen-Wang and heat-bath dynamics. *Random Structures and Algorithms*, 42(4):520–535, 2013. doi:10.1002/rsa.20431.

36 Mario Ullrich. Rapid mixing of Swendsen-Wang dynamics in two dimensions. *Dissertationes Math. (Rozprawy Mat.)*, 502:64, 2014. doi:10.4064/dm502-0-1.

On List Recovery of High-Rate Tensor Codes

Swastik Kopparty
Department of Mathematics and Department of Computer Science, Rutgers University, NJ, USA
swastik.kopparty@gmail.com

Nicolas Resch
Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
nresch@cs.cmu.edu

Noga Ron-Zewi
Department of Computer Science, University of Haifa, Israel
noga@cs.haifa.ac.il

Shubhangi Saraf
Department of Mathematics and Department of Computer Science, Rutgers University, NJ, USA
shubhangi.saraf@gmail.com

Shashwat Silas
Department of Computer Science, Stanford University, CA, USA
silas@stanford.edu

Abstract

We continue the study of list recovery properties of high-rate tensor codes, initiated by Hemenway, Ron-Zewi, and Wootters (FOCS'17). In that work it was shown that the tensor product of an efficient (poly-time) high-rate globally list recoverable code is *approximately* locally list recoverable, as well as globally list recoverable in *probabilistic* near-linear time. This was used in turn to give the first capacity-achieving list decodable codes with (1) local list decoding algorithms, and with (2) *probabilistic* near-linear time global list decoding algorithms. This also yielded constant-rate codes approaching the Gilbert-Varshamov bound with *probabilistic* near-linear time global unique decoding algorithms.

In the current work we obtain the following results:

1. The tensor product of an efficient (poly-time) high-rate globally list recoverable code is globally list recoverable in *deterministic* near-linear time. This yields in turn the first capacity-achieving list decodable codes with *deterministic* near-linear time global list decoding algorithms. It also gives constant-rate codes approaching the Gilbert-Varshamov bound with *deterministic* near-linear time global unique decoding algorithms.
2. If the base code is additionally locally correctable, then the tensor product is (genuinely) locally list recoverable. This yields in turn (non-explicit) constant-rate codes approaching the Gilbert-Varshamov bound that are *locally correctable* with query complexity and running time $N^{o(1)}$. This improves over prior work by Gopi et. al. (SODA'17; IEEE Transactions on Information Theory'18) that only gave query complexity N^{ε} with rate that is exponentially small in $1/\varepsilon$.
3. A nearly-tight combinatorial lower bound on output list size for list recovering high-rate tensor codes. This bound implies in turn a nearly-tight lower bound of $N^{\Omega(1/\log\log N)}$ on the product of query complexity and output list size for locally list recovering high-rate tensor codes.

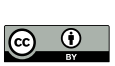
2012 ACM Subject Classification Mathematics of computing → Coding theory; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Coding theory, Tensor codes, List-decoding and recovery, Local codes

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.68

Category RANDOM

Related Version https://eccc.weizmann.ac.il/report/2019/080/



© Swastik Kopparty, Nicolas Resch, Noga Ron-Zewi, Shubhangi Saraf, and Shashwat Silas;
licensed under Creative Commons License CC-BY

Funding *Swastik Kopparty*: Research supported in part by NSF grants CCF-1253886, CCF-1540634, CCF-1814409 and CCF-1412958, and BSF grant 2014359. Some of this research was done while visiting the Institute for Advanced Study.
Nicolas Resch: Research supported in part by NSF-BSF grant CCF-1814629 and 2017732, NSERC grant CGSD2-502898, NSF grants CCF- 1422045, CCF-1527110, CCF-1618280, CCF-1814603, CCF-1910588, NSF CAREER award CCF-1750808 and a Sloan Research Fellowship.
Noga Ron-Zewi: Research supported in part by NSF-BSF grant CCF-1814629 and 2017732.
Shubhangi Saraf: Research supported in part by NSF grants CCF-1350572, CCF-1540634 and CCF-1412958, BSF grant 2014359, a Sloan research fellowship and the Simons Collaboration on Algorithms and Geometry. Some of this research was done while visiting the Institute for Advanced Study.
Shashwat Silas: Research supported in part by NSF-BSF grant CCF-1814629 and 2017732 and a Google Fellowship in the School of Engineering at Stanford.

1 Introduction

Error-correcting codes enable protection of data from errors. They allow one to encode a message so that even after some symbols of the encoding get changed, the original message can still be recovered.

Formally, an *error-correcting code* of *blocklength* n over a finite alphabet Σ is a subset $C \subseteq \Sigma^n$. If k is such that $|C| = |\Sigma|^k$, then a k symbol *message* can be encoded using this code. The redundancy of the code is measured by the *rate* $\rho = k/n$ (so that $|C| = |\Sigma|^{\rho n}$). The robustness to errors is measured by its *relative distance* δ, defined to be the minimum, over all distinct $x, y \in C$, of the relative Hamming distance $\mathrm{dist}(x, y)$. A basic but important observation is that for codes with relative distance δ, for every $w \in \Sigma^n$, there is at most one codeword $c \in C$ for which $\mathrm{dist}(w, c) < \delta/2$. Finding this codeword given w is the algorithmic problem of *unique decoding* C upto half the minimum distance.

Given this setup, we now state some central goals of coding theory. First, we would like to understand the best possible *tradeoffs* for ρ and δ that are achievable. Next, we would like to have *explicit constructions* of codes that achieve this best possible tradeoff. Finally, we would like *efficient algorithms* for decoding such optimal codes upto half their minimum distance – this would give codes correcting the maximum possible fraction of (worst-case) errors for their rate.

For the case of $|\Sigma| = 2$ (the binary alphabet), the *Gilbert-Varshamov bound* states that for all $\delta \leq 1/2$ and $\gamma > 0$ there exist codes with $n \to \infty$ for which[1] $\rho \geq 1 - H_2(\delta) - \gamma$. In fact, a random linear code satisfies this with high probability. The Gilbert-Varshamov bound is the best known tradeoff in the setting where $\delta = \Omega(1)$, and surprisingly, it is not known to be tight. Furthermore, despite their abundance, we do not know how to explicitly construct codes achieving the Gilbert-Varshamov bound.

For growing alphabets, $|\Sigma| = \omega(1)$, the picture is almost completely understood. We know that the best tradeoff achievable is $\rho = 1 - \delta - \gamma$, and furthermore we know how to explicitly construct codes achieving this tradeoff that can be efficiently unique decoded upto half their minimum distance.

[1] Here H_2 is the binary entropy function.

1.1 The cast

In recent years, several important variations of the problem of unique decoding have been considered. We will need many of these, so we give below a quick and gentle introduction (without formal definitions).

List decoding

In list decoding we attempt to decode from an even larger fraction α of errors than $\delta/2$ – now there may be more than one nearby codeword, and our goal is to find the *list* of all of them. A basic limitation is that efficient list decoding is only possible if the number of nearby codewords is guaranteed to be polynomially bounded.

Unlike the case of unique decoding, the optimal tradeoff between the rate ρ and the *list decoding radius* α (for polynomial-size lists) is known for all alphabet sizes. The optimal rate for a given α is known as the *list decoding capacity*. For $|\Sigma| = 2$, the list decoding capacity is $\rho = 1 - H_2(\alpha) - \gamma$, while for $|\Sigma| = \omega(1)$, the list decoding capacity is $\rho = 1 - \alpha - \gamma$. Over large alphabets, this tradeoff can be achieved by explicit codes with efficient list decoding algorithms [21] (see also [27] for the state of the art). Over binary alphabet, we do not know how to explicitly construct codes achieving list decoding capacity.

List recovery

List recovery is a generalization of list decoding where we are given a small list of candidate alphabet symbols at each coordinate (these lists are called the *input lists*) and the goal is to find the *output list* of all codewords that are consistent with many of these input lists. In other words, we want all codewords such that for a $(1 - \alpha)$-fraction of coordinates, the symbol of the codeword at that coordinate lies within the input list for that coordinate (we call these the "nearby codewords"). When the input list size is 1, then list recovery is the same as list decoding.

Local decoding

In local decoding, we want to unique decode in sublinear time. Standard decoding has linear output size, so we need to aim lower. For a given $w \in \Sigma^n$ and a given message coordinate $i \in [k]$, we only ask to recover symbol i of the message underlying the codeword c near w. We would like to run in sublinear time (and hence use only a sublinear number of queries to w), so we allow the algorithm to use randomness and allow a small probability of error.

Local correction is a variation of local decoding where one is required to recover *codeword* symbols as opposed to message symbols. In *approximate* local decoding (local correction, resp.) one is only required to recover correctly *most* of the message (codeword, resp.) coordinates.

Local list decoding

Local list decoding combines the notions of local decoding and list decoding. We are given some $w \in \Sigma^n$, and the goal is that for any nearby codeword, one can in sublinear time recover the ith symbol of the message corresponding to the codeword for any $i \in [k]$. In order to make this precise, the local list decoding algorithm first does some preprocessing and then produces as output a collection of algorithms A_j. For any nearby codeword c, with high

probability one of these algorithms corresponds to it.[2] These algorithms then behave like local decoding algorithms. On input $i \in [k]$, if the algorithm corresponded to a codeword c, then by making queries to only a sublinear number of coordinates, the algorithm with high probability outputs the correct value of the ith symbol of the message corresponding to c.

The above definition of local list decoding can be extended to *local list recovery* in a straightforward way where now the algorithms A_j correspond to all codewords that agree with most of the input lists. As above, we can also define a local correction version of local list decoding (or local list recovery) where the algorithms A_j are required to recover codeword symbols as opposed to message symbols. Finally, we can also define approximate local list decoding (or local list recovery) where the algorithms A_j are only required to recover correctly most of the message (or codeword in the local correction version) coordinates.

The context

The starting point for this paper is the recent result of [23] on high-rate list recoverable tensor codes, and its corollaries. Tensoring is a natural operation on codes that significantly enhances their local properties [5, 34, 9, 10, 15, 6, 7, 37, 28, 36, 26].

The main technical result of [23] was that the tensor product of an efficient (poly-time) high-rate globally list recoverable code is *approximately* locally list recoverable (in either the local decoding or local correction version). They then observed that the "approximately" modifier can be eliminated by pre-encoding the tensor product with a locally decodable code. This gave the first construction of codes with rate arbitrarily close to 1 that are locally list recoverable from an $\Omega(1)$ fraction of errors (however, only in the local decoding version). Finally, using the expander-based distance amplification method of [2, 3] (specialized to the setting of local list recovery [18, 17]), this gave the first capacity-achieving locally list recoverable (and in particular, list decodable) codes with sublinear (and in fact $N^{\tilde{O}(1/\log\log N)}$) query complexity and running time (once more, in the local decoding version).

The above result also yielded further consequences for global decoding. Specifically, [23] observed that the approximate local list recovery algorithm for tensor codes naturally gives a *probabilistic* near-linear time global list recovery algorithm. Once more, using the expander-based distance amplification method of [2, 3, 18], this gave the first capacity-achieving list recoverable (and in particular, list decodable) codes with *probabilistic* near-linear time global list recovery algorithms. Finally, via the random concatenation method of [33, 19], this yielded in turn a (randomized) construction of constant-rate binary codes approaching the Gilbert-Varshamov bound with a *probabilistic* near-linear time algorithm for global unique decoding upto half the minimum distance.

One could potentially hope (following [17] which implemented a local version of [33, 19]) for an analogous result that would give constant-rate codes approaching the Gilbert-Varshamov bound that are locally correctable (or locally decodable) with query complexity and running time $N^{o(1)}$. However, what prevented [23] from obtaining such a result was the fact that their capacity-achieving locally list recoverable codes only worked in the local decoding version (i.e., they were only able to recover message coordinates).

[2] Some of these algorithms A_j might not correspond to any codeword and might output garbage. Later in the paper we define local list decoding to not allow these garbage producing A_j's. Eliminating the garbage can be easily done if the underlying code is also *locally testable*, and in this case the stronger notion can be achieved.

1.2 Results

We revisit the technique of [23] and show the following.

- The tensor product of an efficient (poly-time) high-rate globally list recoverable code is globally list recoverable in *deterministic* near-linear time. Plugging this into the machinery of [2, 3, 18], we get the first capacity-achieving list recoverable (and in particular, list decodable) codes with *deterministic* near-linear time global list recovery algorithms. Plugging this into the machinery of [33, 19], yields in turn constant-rate binary codes (with a randomized construction) approaching the Gilbert-Varshamov bound with *deterministic* near-linear time global unique decoding algorithms.

  Our deterministic global list recovery algorithm is obtained by derandomizing the random choices of the [23] algorithm using appropriate samplers.

- An instantiation of the base code to produce tensor product codes which are themselves genuinely locally list recoverable (i.e., not just approximately locally list recoverable) in the *local correction version*. Once more, plugging this into the machinery of [2, 3, 17], we get capacity-achieving locally list recoverable codes, but now in the *local correction version*. This now plugs in turn into the machinery of [33, 19, 17] to give constant-rate binary codes (with a randomized construction) approaching the Gilbert-Varshamov bound that are locally decodable with query complexity and running time $N^{o(1)}$. This improves over prior work [17] that only gave query complexity N^{ε} with rate that is exponentially small in $1/\varepsilon$.

  We obtain our result by taking the base code to be the *intersection* of an efficient (poly-time) high-rate globally list recoverable code and a high-rate locally correctable code. Assuming both codes are linear, we have that the intersection is a high-rate code that is both! The result of [23] already guarantees that this tensor product is approximately locally list recoverable (in the local correction version), and we use the fact that the tensor product of a locally correctable codes is also locally correctable [37] to remove the "approximately" modifier.[3]

- A combinatorial lower bound showing the limitations on the list recoverability of high-rate tensor codes. Specifically, we show that when the rate of the base code is high, every t-wise tensor product of this code has output list size *doubly-exponential* in t. This means that taking t to be more than $\log \log N$ leads to superpolynomial output list size, precluding the possibility of efficient list recovery.

  Instantiating this appropriately, this implies in turn that there is a base code such that for every tensor power with block length N, the product of the query complexity and output list size for local list recovery is at least $N^{\Omega(1/\log \log N)}$. We note that in contrast, it could be that for every base code, there is a tensor power with block length N for which local correction can be done with query complexity $O(1)$.

  A key observation that we use is that a high-rate code has many codewords with pairwise-disjoint supports. We combine this along with other linear-algebraic arguments to design a list recovery instance for the tensor product of a high-rate code which has many codewords that are consistent with it.

Below we give formal statements of our results. For formal definitions of the various notions of decoding in the following theorem statements, see Section 2.

[3] To eliminate "garbage" we also use the fact that the tensor product is locally testable [37].

1.2.1 Deterministic near-linear time global list recovery

Our first main result shows that the tensor product of an efficient (poly-time) high-rate globally list recoverable code is globally list recoverable in *deterministic* near-linear time. In the theorem statement, one should think of all parameters δ, α, L, t, and consequently also s, as constants (or more generally, as slowly increasing/decreasing functions of n). In that case, the theorem says that if $C \subseteq \mathbb{F}^n$ is (α, ℓ, L)-globally list recoverable deterministically in time $T = \mathrm{poly}(n)$, then the t-iterated tensor product $C^{\otimes t}$ of length $N := n^t$ is $(\Omega(\alpha), \ell, L^{O(1)})$-globally list recoverable deterministically in time $O(n^t \cdot T) = n^{t+O(1)} = N^{1+O(1/t)}$.

▶ **Theorem 1** (Deterministic near-linear time list recovery of high-rate tensor codes)**.** *The following holds for any $\delta, \alpha > 0$, and $s = \mathrm{poly}(1/\delta, 1/\alpha)$. Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-globally list recoverable deterministically in time T. Then $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $(\alpha \cdot s^{-t^2}, \ell, L^{s^{t^3} \cdot L^t})$-globally list recoverable deterministically in time $n^t \cdot T \cdot L^{s^{t^3} \cdot L^t}$.*

Applying the expander-based distance amplification method of [2, 3, 18] on the codes given by the above theorem, we obtain the first capacity-achieving list recoverable (and in particular, list decodable) codes with *deterministic* near-linear time global list recovery algorithms.

▶ **Corollary 2** (Deterministic nearly-linear time capacity-achieving list recoverable codes)**.** *For any constants $\rho \in [0, 1]$, $\gamma > 0$, and $\ell \geq 1$ there exists an infinite family of codes $\{C_N\}_N$, where C_N has block length N, alphabet size $N^{o(1)}$, rate ρ, and is $(1 - \rho - \gamma, \ell, N^{o(1)})$-globally list recoverable deterministically in time $N^{1+o(1)}$.*

Applying the random concatenation method of [33, 19], the above corollary yields in turn constant-rate codes approaching the Gilbert-Varshamov bound with *deterministic* near-linear time global unique decoding algorithms.

▶ **Corollary 3** (Deterministic near-linear time unique decoding up to the GV bound)**.** *For any constants $\rho \in [0, 0.02]$ and $\gamma > 0$ there exists an infinite family of binary linear codes $\{C_N\}_N$, where C_N has block length N and rate ρ, and is globally uniquely decodable deterministically from $\frac{H_2^{-1}(1-\rho)-\gamma}{2}$-fraction of errors in time $N^{1+o(1)}$.*

1.2.2 Local list recovery

Our second main result shows that if the base code is *both* globally list recoverable and locally correctable, then the tensor product is (genuinely) locally list recoverable (in the local correction version).

▶ **Theorem 4** (Local list recovery of high-rate tensor codes)**.** *The following holds for any $\delta, \alpha > 0$, and $s = \mathrm{poly}(1/\delta, 1/\alpha)$. Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-globally list recoverable, and locally correctable from $(\delta/2)$-fraction of errors with query complexity Q, and $t \geq 3$. Then $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $(\alpha \cdot s^{-t^3}, \ell, L^{s^{t^3} \cdot \log^t L})$-locally list recoverable with query complexity $n^{O(1)} \cdot Q^{O(t)} \cdot L^{s^{t^3} \cdot \log^t L}$.*

Once more, applying the expander-based distance amplification method of [2, 3, 18, 17], as well as the random concatenation method of [33, 19, 17], the above theorem yields constant-rate codes approaching the Gilbert-Varshamov bound that are *locally correctable* with query complexity $N^{o(1)}$.

▶ **Corollary 5** (Local correction up to the GV bound). *For any constants $\rho \in [0, 0.02]$ and $\gamma > 0$ there exists an infinite family of binary linear codes $\{C_N\}_N$, where C_N has block length N and rate ρ, and is locally correctable from $\frac{H_2^{-1}(1-\rho)-\gamma}{2}$-fraction of errors with query complexity $N^{o(1)}$.*

1.2.3 Combinatorial lower bound on output list size

Our final main result shows a nearly-tight combinatorial lower bound on output list size for list recovering high-rate tensor codes.

▶ **Theorem 6** (Output list size for list recovering high-rate tensor codes). *Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of rate $1-\gamma$, and that $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $(0, \ell, L)$-list recoverable. Then $L \geq \ell^{1/\gamma^t}$.*

The above bound can be instantiated concretely as follows.

▶ **Corollary 7.** *For any $\delta > 0$ and $\ell > 1$ there exists $L > 1$ such that the following holds for any sufficiently large n. There exists a linear code $C \subseteq \mathbb{F}^n$ of relative distance δ that is $(\Omega(\delta), \ell, L)$-list recoverable, but $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is only $(0, \ell, L')$-list recoverable for $L' \geq \exp((2\delta)^{-(t-3/2)} \cdot \sqrt{\log L})$.*

Finally, we also obtain a nearly-tight lower bound of $N^{\Omega(1/\log\log N)}$ on the product of query complexity and output list size for locally list recovering high-rate tensor codes.

▶ **Corollary 8.** *For any $\delta > 0$ and sufficiently large n there exists a linear code $C \subseteq \mathbb{F}^n$ of relative distance δ such that the following holds. Suppose that $C^{\otimes t} \subseteq \mathbb{F}^N$ is $(\frac{1}{N}, 2, L)$-locally list recoverable with query complexity Q. Then $Q \cdot L \geq N^{\Omega_\delta(1/\log\log N)}$.*

2 Preliminaries

For a prime power q we denote by $\mathbb{F}_q$ the finite field of q elements. For any finite alphabet Σ and for any pair of strings $x, y \in \Sigma^n$, the relative distance between x and y is the fraction of coordinates $i \in [n]$ on which x and y differ, and is denoted by $\mathrm{dist}(x, y) := |\{i \in [n] : x_i \neq y_i\}|/n$. For a subset $Y \subseteq \Sigma^n$, we denote by $\mathrm{dist}(x, Y)$ the minimum relative distance of a string $y \in Y$ from x. For a positive integer ℓ we denote by $\binom{\Sigma}{\ell}$ the collection of all subsets of Σ of size ℓ and by $\binom{\Sigma}{\leq \ell}$ the collection of all nonempty subsets of Σ of size at most ℓ. For any string $x \in \Sigma^n$ and tuple $S \in \binom{\Sigma}{\leq \ell}^n$ we denote by $\mathrm{dist}(x, S)$ the fraction of coordinates $i \in [n]$ for which $x_i \notin S_i$, that is, $\mathrm{dist}(x, S) := |\{i \in [n] : x_i \notin S_i\}|/n$. For a string $x \in \Sigma^n$ and a subset $T \subseteq [n]$, we use $x|_T \in \Sigma^{|T|}$ to denote the restriction of x to the coordinates in T. Throughout the paper, we use $\exp(n)$ to denote $2^{\Theta(n)}$, and whenever we use log, it is base 2, unless noted otherwise.

2.1 Error-correcting codes

An error-correcting code is simply a subset $C \subseteq \Sigma^n$. We call Σ the alphabet of the code, and n its block length. The elements of C are called codewords. If $\mathbb{F}$ is a finite field and Σ is a vector space over $\mathbb{F}$, we say that a code $C \subseteq \Sigma^n$ is $\mathbb{F}$-linear if it is an $\mathbb{F}$-linear subspace of the $\mathbb{F}$-vector space Σ^n. If $\Sigma = \mathbb{F}$, we simply say that C is linear.

The rate of a code is the ratio $\rho := \frac{\log|C|}{\log(|\Sigma|^n)}$, which for $\mathbb{F}$-linear codes equals $\frac{\dim_{\mathbb{F}}(C)}{n \cdot \dim_{\mathbb{F}}(\Sigma)}$. The relative distance $\mathrm{dist}(C)$ of C is the minimum $\delta > 0$ such that for every pair of distinct codewords $c_1, c_2 \in C$ it holds that $\mathrm{dist}(c_1, c_2) \geq \delta$. We denote by $\Delta(C) := \mathrm{dist}(C) \cdot n$ the (absolute) distance of C.

The best known general trade-off between rate and distance of codes is the Gilbert-Varshamov bound, attained by random (linear) codes. For $x \in [0,1]$ let

$$H_q(x) = x \log_q(q-1) + x \log_q(1/x) + (1-x)\log_q(1/(1-x))$$

denote the q-ary entropy function.

▶ **Theorem 9** (Gilbert-Varshamov (GV) bound, [12, 35]). *For any prime power q, $\delta \in (0, 1-\frac{1}{q})$, and $\rho \in (0, 1-H_q(\delta))$, a random linear code $C \subseteq \mathbb{F}_q^n$ of rate ρ has relative distance at least δ with probability $1-\exp(-n)$.*

▶ **Corollary 10.** *For any $\rho \in [0,1]$ and $\gamma > 0$, and prime power $q \geq 2^{H_2(1-\rho-\gamma)/\gamma}$, a random linear code $C \subseteq \mathbb{F}_q^n$ of rate ρ has relative distance at least $1-\rho-\gamma$ with probability $1-\exp(-n)$.*

An encoding map for C is a bijection $E_C : \Sigma^k \to C$, where $|\Sigma|^k = |C|$. We call the elements in the domain of E_C messages, and k the message length. We say that C is encodable in time T if an encoding map for C can be computed in time T. For a code $C \subseteq \Sigma^n$ of relative distance δ and a given parameter $\alpha < \delta/2$, we say that C is decodable from α-fraction of errors in time T if there exists an algorithm, running in time T, that given a received word $w \in \Sigma^n$, computes the unique codeword $c \in C$ (if any) which satisfies $\mathrm{dist}(c,w) \leq \alpha$.

▶ **Proposition 11** (Reed-Solomon codes, [29, 8]). *For any prime power q and integers $k \leq n \leq q$, there exists a linear code $C \subseteq \mathbb{F}_q^n$ of rate $\rho := k/n$ and relative distance at least $1-\rho$ that is encodable and decodable from $\frac{1-\rho}{2}$-fraction of errors in time $\mathrm{poly}(n, \log q)$.*

Let $C \subseteq \mathbb{F}^n$ be a linear code of dimension k. A generating matrix for C is an $n \times k$ matrix G such that $\mathrm{Im}(G) = C$. A parity-check matrix for C is an $(n-k) \times n$ matrix H such that $\ker(H) = C$. The dual code $C^\perp \subseteq \mathbb{F}^n$ is given by

$$C^\perp = \{y \in \mathbb{F}^n \mid \langle y, c\rangle = 0 \ \forall c \in C\}.$$

It is well-known that $C^{\perp\perp} = C$, and that a matrix G is a generating matrix for C if and only if G^T is a parity-check matrix for $C^\perp$.

2.2 List recoverable codes

List recovery is a generalization of the standard error-correction setting where each entry w_i of the received word w is replaced with a list S_i of ℓ possible symbols of Σ. Formally, for $\alpha \in [0,1]$ and integers ℓ, L we say that a code $C \subseteq \Sigma^n$ is (α, ℓ, L)-list recoverable if for any tuple $S \in \binom{\Sigma}{\leq \ell}^n$ there are at most L different codewords $c \in C$ so that $\mathrm{dist}(c, S) \leq \alpha$. We say that C is (α, L)-list decodable if it is $(\alpha, 1, L)$-list recoverable.

▶ **Corollary 12** ([24], Corollary 2.2). *For any $\rho \in [0,1]$, $\gamma > 0$, and $\ell \geq 1$, and for sufficiently large prime power q, a random linear code $C \subseteq \mathbb{F}_q^n$ of rate ρ is $(1-\rho-\gamma, \ell, q^{O(\ell/\gamma)})$-list recoverable with probability $1-\exp(-n)$.*

We say that C is (α, ℓ, L)-list recoverable in time T if there exists an algorithm, running in time T, that given a tuple $S \in \binom{\Sigma}{\ell}^n$, returns all codewords $c \in C$ (if any) which satisfy $\mathrm{dist}(c, S) \leq \alpha$. The following theorem from [22, 20, 23] gives a family of high-rate linear codes which are efficiently list recoverable with constant alphabet size and nearly-constant output list size.

▶ **Theorem 13** ([24], Theorem A.1). *There exists an absolute constant b_0 so that the following holds. For any $\gamma > 0$, $\ell \geq 1$, $q \geq \ell^{b_0/\gamma}$ that is an even power of a prime*[4]*, and integer $n \geq q^{b_0 \ell/\gamma}$, there exists a linear code $C \subseteq \mathbb{F}_q^n$ of rate $1-\gamma$ and relative distance $\Omega(\gamma^2)$ that is $(\Omega(\gamma^2), \ell, L)$-list recoverable for $L = q^{q^{(\ell/\gamma)\cdot \exp(\log^* n)}}$. Moreover, C can be encoded in time $\mathrm{poly}(n, \log q)$ and list recovered in time $\mathrm{poly}(n, L)$.*

2.3 Local codes

2.3.0.1 Locally testable codes

Intuitively, a code is said to be locally testable [11, 30, 16] if, given a string $w \in \Sigma^n$, it is possible to determine whether w is a codeword of C, or rather far from C, by reading only a small part of w. For our purposes, we shall also require an additional *tolerance* property of determining whether w is sufficiently close to the code.

▶ **Definition 14** (Tolerant locally testable code (Tolerant LTC)). *We say that a code $C \subseteq \Sigma^n$ is (Q, α, β)-tolerantly locally testable if there exists a randomized algorithm A that satisfies the following requirements:*

- ***Input:*** *A gets oracle access to a string $w \in \Sigma^n$.*
- ***Query complexity:*** *A makes at most Q queries to the oracle w.*
- ***Completeness:*** *If $\mathrm{dist}(w, C) \leq \alpha$, then A accepts with probability at least $\frac{2}{3}$.*
- ***Soundness:*** *If $\mathrm{dist}(w, C) \geq \beta$, then A rejects with probability at least $\frac{2}{3}$.*

▶ Remark 15. The definition requires $0 \leq \alpha < \beta \leq 1$. The above success probability of $\frac{2}{3}$ can be amplified using sequential repetition, at the cost of increasing the query complexity. Specifically, amplifying the success probability to $1 - \exp(-t)$ requires increasing the query complexity by a multiplicative factor of $O(t)$.

Locally correctable codes

Intuitively, a code is said to be locally correctable [4, 32, 25] if, given a codeword $c \in C$ that has been corrupted by some errors, it is possible to decode any coordinate of c by reading only a small part of the corrupted version of c.

▶ **Definition 16** (Locally correctable code (LCC)). *We say that a code $C \subseteq \Sigma^n$ is (Q, α)-locally correctable if there exists a randomized algorithm A that satisfies the following requirements:*

- ***Input:*** *A takes as input a coordinate $i \in [n]$, and also gets oracle access to a string $w \in \Sigma^n$ that is α-close to a codeword $c \in C$.*
- ***Query complexity:*** *A makes at most Q queries to the oracle w.*
- ***Output:*** *A outputs c_i with probability at least $\frac{2}{3}$.*

▶ Remark 17. The definition requires $\alpha < \mathrm{dist}(C)/2$. The above success probability of $\frac{2}{3}$ can be amplified using sequential repetition, at the cost of increasing the query complexity. Specifically, amplifying the success probability to $1 - \exp(-t)$ requires increasing the query complexity by a multiplicative factor of $O(t)$.

[4] That is, q is of the form p^{2t} for a prime p and for an integer t.

Locally list recoverable codes

The following definition from [14, 32, 17] generalizes the notion of locally correctable codes to the setting of list decoding/recovery. In this setting, the local list recovery algorithm is required to output in an implicit sense all codewords that are consistent with most of the input lists.

▶ **Definition 18** (Locally list recoverable code). *We say that a code $C \subseteq \Sigma^n$ is $(Q, \alpha, \varepsilon, \ell, L)$-locally list recoverable if there exists a randomized algorithm A that satisfies the following requirements:*

- ***Input:*** *A gets oracle access to a string $S \in \binom{\Sigma}{\leq \ell}^n$.*
- ***Query complexity:*** *A makes at most Q queries to the oracle S.*
- ***Output:*** *A outputs L randomized algorithms $A_1, \ldots, A_L$, where each A_j takes as input a coordinate $i \in [n]$, makes at most Q queries to the oracle S, and outputs a symbol in Σ.*
- ***Completeness:*** *For any codeword $c \in C$ which satisfies $\mathrm{dist}(c, S) \leq \alpha$, with probability at least $1 - \varepsilon$ over the randomness of A, the following event happens: there exists some $j \in [L]$ such that for all $i \in [n]$,*

$$\Pr[A_j(i) = c_i] \geq \frac{2}{3}, \tag{1}$$

  where the probability is over the internal randomness of A_j.
- ***Soundness:*** *With probability at least $1 - \varepsilon$ over the randomness of A, the following event happens: for every $j \in [L]$, there exists some $c \in C$ such that for all $i \in [n]$,*

$$\Pr[A_j(i) = c_i] \geq \frac{2}{3},$$

  where the probability is over the internal randomness of A_j.

We say that A has preprocessing time T_{pre} if A outputs the description of the algorithms $A_1, \ldots, A_L$ in time at most T_{pre}, and has running time T if each A_j has running time at most T. As before, we say that the code C is $(Q, \alpha, \varepsilon, L)$-locally list decodable if it is $(Q, \alpha, \varepsilon, 1, L)$-locally list recoverable.

2.4 Tensor codes

In this paper we study the list recovery properties of the high-rate tensor product codes, defined as follows.

▶ **Definition 19** (Tensor product codes). *Let $C_1 \subseteq \mathbb{F}^{n_1}$, $C_2 \subseteq \mathbb{F}^{n_2}$ be linear codes. Their tensor product code $C_1 \otimes C_2 \subseteq \mathbb{F}^{n_1 \times n_2}$ consists of all matrices $M \in \mathbb{F}^{n_1 \times n_2}$ such that all the rows of M are codewords of C_2 and all the columns are codewords of C_1.*

3 Deterministic near-linear time global list recovery

3.1 Deterministic near-linear time list recovery of high-rate tensor codes

In this section we prove Theorem 1, restated bollow, which shows that the tensor product of an efficient (poly-time) high-rate globally list recoverable code is globally list recoverable in *deterministic* near-linear time.

▶ **Theorem 1** (Deterministic near-linear time list recovery of high-rate tensor codes)**.** *The following holds for any $\delta, \alpha > 0$, and $s = \mathrm{poly}(1/\delta, 1/\alpha)$. Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-globally list recoverable deterministically in time T. Then $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $(\alpha \cdot s^{-t^2}, \ell, L^{s^{t^3} \cdot L^t})$-globally list recoverable deterministically in time $n^t \cdot T \cdot L^{s^{t^3} \cdot L^t}$.*

Theorem 1 follows by applying the lemma below iteratively.

▶ **Lemma 20.** *The following holds for any $\delta, \alpha, \delta_{\mathrm{dec}}, \delta'_{\mathrm{dec}} > 0$, and $\bar{s} = \mathrm{poly}(1/\delta, 1/\alpha, 1/\delta_{\mathrm{dec}}, 1/\delta'_{\mathrm{dec}})$.*

Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-globally list recoverable deterministically in time T, and $C' \subseteq \mathbb{F}^{n'}$ is a linear code that is (α', ℓ, L')-globally list recoverable deterministically in time T'. Suppose furthermore that C, C' are uniquely decodable deterministically from $\delta_{\mathrm{dec}}, \delta'_{\mathrm{dec}}$-fraction of errors in times $T_{\mathrm{dec}}, T'_{\mathrm{dec}}$, respectively.

Then $C \otimes C' \subseteq \mathbb{F}^{n \times n'}$ is $(\alpha'/\bar{s}, \ell, (L')^{\bar{s} \cdot L/(\alpha')^2})$-globally list recoverable deterministically in time

$$(L')^{\bar{s} \cdot L/(\alpha')^2} \cdot n \cdot (n' \cdot (T + T_{\mathrm{dec}}) + n \cdot T'_{\mathrm{dec}} + T') \,.$$

We now sketch the proof of Lemma 20. Our plan is to derandomize the approximate local list recovery algorithm for high-rate tensor codes of [23]. Recall that an approximate local list recovery algorithm (local correction version) is a randomized algorithm A that outputs a collection of (without loss of generality, deterministic) local algorithms A_j satisfying the following: for any codeword c that is consistent with most of the input lists, with high probability (over the randomness of A) one of the local algorithms A_j locally corrects most of the coordinates of c.

As observed in [23], an approximate local list recovery algorithm naturally gives a *probabilistic* near-linear time *global* list recovery algorithm as follows. First run the algorithm A to obtain the collection of local algorithms A_j. Then for each A_j, output a codeword that is obtained by applying A_j on each codeword coordinate, and then uniquely decoding the resulting word to the closest codeword. The guarantee now is that any codeword that is consistent with most of the input lists will be output with high probability.

To derandomize the probabilistic global algorithm described above, we note that the preprocessing algorithm A in [23] produces the collection of local algorithms A_j by choosing a random subset of rows in the tensor product,[5] that is chosen uniformly at random amongst all subsets of the appropriate size. We then observe that this subset can be alternatively chosen using a randomness-efficient *sampler* without harming much the performance. Finally, since the sampler uses a small amount of randomness (logarithmic in the blocklength of C), we can afford to iterate over all seeds and return the union of all output lists. This gives a *deterministic* near-linear time global list recovery algorithm that outputs all codewords that are consistent with most of the input lists.

3.1.1 Samplers

We start by defining the appropriate samplers we use.

[5] In [23], the role of columns and rows is swapped.

▶ **Definition 21** ((averaging) sampler). *An (n, η, γ)-sampler with randomness r and sample size m is a randomized algorithm that tosses r random coins and outputs a subset $I \subseteq [n]$ of size m such that the following holds. For any function $f : [n] \to [0, 1]$, with probability at least $1 - \eta$ over the choice of I,*

$$\left|\mathbb{E}_{i \in I}[f(i)] - \mathbb{E}_{i \in [n]}[f(i)]\right| \leq \gamma.$$

We shall use the following construction from Goldreich [13].

▶ **Theorem 22** ([13], Corollary 5.6). *For any $\eta, \gamma > 0$ and integer n, there exists an (n, η, γ)-sampler with randomness $\log(n/\gamma)$, sample size $O\left(1/(\eta\gamma^2)\right)$, and running time $\mathrm{poly}(\log n, 1/\eta, 1/\gamma)$.*

In what follows, let Γ denote the (n, η, γ)-sampler promised by the above theorem, where we set $\eta := \frac{0.1}{L} \cdot \frac{\delta_{\mathrm{dec}} \cdot \delta'_{\mathrm{dec}}}{3}$ and $\gamma := \alpha' \cdot \frac{\delta \cdot \delta_{\mathrm{dec}} \cdot \delta'_{\mathrm{dec}}}{24}$. Let $r := \log(n/\gamma) \leq \log(n \cdot \bar{s}/\alpha')$ and $m := O(1/(\eta\gamma^2)) \leq L \cdot \bar{s}/(\alpha')^2$ denote the randomness and sample size of Γ, respectively (assuming that $\bar{s}$ is a sufficiently large polynomial).

3.1.2 Randomness-efficient algorithm

We first describe a randomness-efficient global list recovery algorithm $\tilde{A}$ for $C \otimes C'$ that is obtained by replacing the choice of a uniform random subset of rows made in [23] with a sample from Γ. We will later observe that the randomness can be eliminated by iterating over all seeds of Γ and returning the union of all output lists.

The algorithm $\tilde{A}$ behaves as follows. First, it uses Γ to sample a subset of m rows $I = \{i_1, \ldots, i_m\} \subseteq [n]$. Then for $k = 1, \ldots, m$, it runs the list recovery algorithm A' for C' on the i_k-th row $S|_{\{i_k\} \times [n']}$; let $\mathcal{L}'_{i_1}, \mathcal{L}'_{i_2}, \ldots, \mathcal{L}'_{i_m} \subseteq C'$ denote the lists output by A' on each of the rows in I. Finally, for any choice of codewords $c'_1 \in \mathcal{L}'_{i_1}, c'_2 \in \mathcal{L}'_{i_2}, \ldots, c'_m \in \mathcal{L}'_{i_m}$, the algorithm $\tilde{A}$ outputs a codeword $\tilde{c} \in C \otimes C'$ that is obtained as follows.

For each column $j \in [n']$, the algorithm $\tilde{A}$ runs the list recovery algorithm A for C on the j-th column $S|_{[n] \times \{j\}}$; let $\mathcal{L}_1, \mathcal{L}_2, \ldots \mathcal{L}_{n'} \subseteq C$ denote the lists output by A on each of the n' columns. Then the algorithm $\tilde{A}$ chooses for each column $j \in [n']$ the codeword $c_j \in \mathcal{L}_j$ whose restriction to I is closest to $((c'_1)_j, (c'_2)_j \ldots, (c'_m)_j)$ (i.e., the restriction of $c'_1, c'_2, \ldots, c'_m$ to the j-th column). Finally, the algorithm $\tilde{A}$ sets the value of each column $j \in [n']$ to c_j, and uniquely decodes the resulting word $\tilde{c}_0$ to the nearest codeword $\tilde{c} \in C \otimes C'$, assuming there is one at distance at most $\delta_{\mathrm{dec}} \cdot \delta'_{\mathrm{dec}}$. If $\mathrm{dist}(\tilde{c}, S) \leq \alpha'/\bar{s}$, then $\tilde{A}$ includes $\tilde{c}$ in the output list $\tilde{\mathcal{L}}$. The formal description is given in Algorithm 1.

3.2 Deterministic nearly-linear time capacity-achieving list recoverable codes

In this section we prove the following lemma which implies Corollary 2 from the introduction.

▶ **Lemma 23.** *For any constants $\rho \in [0, 1]$, $\gamma > 0$, and $\ell \geq 1$ there exists an infinite family of codes $\{C_N\}_N$ that satisfy the following.*

- *C_N is an $\mathbb{F}_2$-linear code of block length N and alphabet size $N^{o(1)}$.*
- *C_N has rate ρ and relative distance at least $1 - \rho - \gamma$.*
- *C_N is $(1 - \rho - \gamma, \ell, N^{o(1)})$-globally list recoverable deterministically in time $N^{1+o(1)}$.*
- *C_N is encodable deterministically in time $N^{1+o(1)}$.*

Algorithm 1 The randomness-efficient global list recovery algorithm $\tilde{A}$ for $C \otimes C'$.

function $\tilde{A}(S \in \binom{\mathbb{F}}{\leq \ell}^{n \times n'})$
  Sample $I = \{i_1, \ldots, i_m\} \subseteq [n]$ of size m using sampler Γ.
  for $k = 1, \ldots, m$ **do**
    Run the list recovery algorithm A' for C' on the i_k-th row $S|_{\{i_k\} \times [n']}$, and let $\mathcal{L}'_{i_k} \subseteq C'$ be the list of codewords output by A'.
  end for
  Initialize $\tilde{c}_0 \in \mathbb{F}^{n \times n'}$, $\tilde{\mathcal{L}} \leftarrow \emptyset$.
  for any choice of codewords $c'_1 \in \mathcal{L}'_{i_1}, c'_2 \in \mathcal{L}'_{i_2}, \ldots, c'_m \in \mathcal{L}'_{i_m}$ **do**
    for $j \in [n']$ **do**
      Run the list recovery algorithm A for C on the j-th column $S|_{[n] \times \{j\}}$, and let $\mathcal{L}_j \subseteq C$ be the list of codewords output by A.
      Choose a codeword $c_j \in \mathcal{L}_j$ for which $c_j|_I$ is closest to $((c'_1)_j, (c'_2)_j, \ldots, (c'_m)_j)$ (breaking ties arbitrarily).
      Set the j-th column of $\tilde{c}_0$ to c_j.
    end for
    Uniquely decode $\tilde{c}_0$ from $(\delta_{\mathrm{dec}} \cdot \delta'_{\mathrm{dec}})$-fraction of errors, and let $\tilde{c} \in C \otimes C'$ be the resulting codeword (if exists). If $\mathrm{dist}(\tilde{c}, S) \leq \alpha'/\bar{s}$, add $\tilde{c}$ to $\tilde{\mathcal{L}}$.
  end for
end function

To prove the above lemma, we first use Theorem 1 to obtain deterministic nearly-linear time *high-rate* list recoverable codes, and then use the Alon-Edmonds-Luby (AEL) distance amplification method [2, 3] to turn these codes into deterministic nearly-linear time *capacity-achieving* list recoverable codes.

3.3 Deterministic near-linear time unique decoding up to the GV bound

In this section we prove the following lemma which implies Corollary 3 from the introduction.

▶ **Lemma 24.** *For any constants $\rho \in [0, 0.02]$ and $\gamma > 0$ there exists an infinite family of binary linear codes $\{C_N\}_N$, where C_N has block length N and rate ρ, and is globally uniquely decodable deterministically from $\frac{H_2^{-1}(1-\rho)-\gamma}{2}$-fraction of errors in time $N^{1+o(1)}$.*

Furthermore, there exists a randomized algorithm which, on input N, runs in time $N^{1+o(1)}$ and outputs with high probability a description of a code C_N with the properties above. Given the description, the code C_N can be encoded deterministically in time $N^{1+o(1)}$.

To prove the above lemma, we rely on a lemma from [33, 23] which says that one can turn a code that approximately satisfies the Singleton bound into one that approximately satisfies the GV bound via random concatenation.

4 Local list recovery

4.1 Local list recovery of high-rate tensor codes

In this section we prove the following lemma which implies Theorem 4 from the introduction.

▶ **Lemma 25.** *The following holds for any $\delta, \alpha, \varepsilon > 0$ and $s = \mathrm{poly}(1/\delta, 1/\alpha)$. Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-globally list recoverable, and $(Q, \delta/2)$-locally correctable, and $t \geq 3$. Then $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $(\tilde{Q}, \alpha \cdot s^{-t^3}, \varepsilon, \ell, L^{s^{t^3} \cdot \log^t L} \cdot \log(1/\varepsilon))$-locally list recoverable for*

$$\tilde{Q} = n^3 \cdot (Q \log Q)^t \cdot L^{s^{t^3} \cdot \log^t L} \cdot \log^2(1/\varepsilon).$$

Moreover, if C is globally list recoverable in time $\mathrm{poly}(n)$, locally correctable in time T, and globally decodable for $(\delta/2)$-fraction of errors in time $\mathrm{poly}(n)$, then the local list recovery algorithm for $C^{\otimes t}$ has preprocessing time $\mathrm{poly}(n) \cdot L^{s^{t^3} \cdot \log^t L} \cdot \log^2(1/\varepsilon)$ and running time $\mathrm{poly}(n) \cdot (T \log T)^t \cdot (s^{t^3} \log^t L)$.

The above lemma relies on the following lemma from [23] which says that the tensor product of a high-rate globally list recoverable code (which is not necessarily locally correctable) is *approximately* locally list recoverable. Approximate local list recovery is a relaxation of local list recovery, where the local algorithms in the output list are not required to recover *all* the codeword coordinates, but only *most* of them. Formally, a β-approximately $(Q, \alpha, \varepsilon, \ell, L)$-locally list recoverable code $C \subseteq \Sigma^n$ satisfies all the requirements of Definition 18, except that the requirement (1) is replaced with the relaxed condition that

$$\Pr_{i \in [n]} \left[A_j(i) = c_i \right] \geq 1 - \beta, \tag{2}$$

where the probability is over the choice of uniform random $i \in [n]$, and the soundness requirement is eliminated.

▶ **Lemma 26** (Approximate local list recovery of high-rate tensor codes, [24], Lemma 4.1)**.** *The following holds for any $\delta, \alpha, \beta, \varepsilon > 0$ and $s = \mathrm{poly}(1/\delta, 1/\alpha, 1/\beta)$. Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-globally list recoverable. Then $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is β-approximately $(n \cdot (s^{t^2} \log^t L), \alpha \cdot s^{-t^2}, \varepsilon, \ell, L^{s^{t^2} \cdot \log^t L} \cdot \log(1/\varepsilon))$-locally list recoverable.*

Moreover, if C is globally list recoverable in time $\mathrm{poly}(n)$, then the approximate local list recovery algorithm for $C^{\otimes t}$ has preprocessing time $\log(n) \cdot L^{s^{t^2} \cdot \log^t L} \cdot \log(1/\varepsilon)$ and running time $\mathrm{poly}(n) \cdot (s^{t^2} \log^t L)$.

To turn the approximate local list recovery algorithm given by the above lemma into a local list recovery algorithm we shall use the fact that the tensor product of a locally correctable code is also locally correctable with slightly worse parameters. A similar observation was made in [37, Proposition 3.15.].

▶ **Lemma 27** (Local correction of tensor codes)**.** *Suppose that $C \subseteq \mathbb{F}^n$ is a linear code that is (Q, α)-locally correctable. Then $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $\left(\left(O(Q \log Q)\right)^t, \alpha^t\right)$-locally correctable.*

Moreover, if C is locally correctable in time T, then the local correction algorithm for $C^{\otimes t}$ runs in time $(O(T \log T))^t$.

To guarantee the soundness property we shall also use the following lemma which says that high-rate tensor codes are tolerantly locally testable.

▶ **Lemma 28** (Tolerant local testing of high-rate tensor codes)**.** *Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of relative distance δ, and $t \geq 3$. Then $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $(n^2 \cdot \delta^{-O(t)}, \delta^{O(t)}, (\delta/2)^t)$-tolerantly locally testable.*

Moreover, if C is globally decodable from $(\delta/2)$-fraction of errors in time T, then the tolerant local testing algorithm for $C^{\otimes t}$ runs in time $T \cdot n \cdot \delta^{-O(t)}$.

Finally, we show a general transformation that turns an approximately locally list recoverable code that is also locally correctable and tolerantly locally testable into a (genuinely) locally list recoverable code.

▶ **Lemma 29.** *Suppose that $C \subseteq \Sigma^n$ is a β-approximately $(Q, \alpha, \varepsilon, \ell, L)$-locally list recoverable code that is also $(Q_{\text{corr}}, \gamma)$-locally correctable and $(Q_{\text{test}}, \beta, \gamma)$-tolerantly locally testable. Then C is $(\tilde{Q}, \alpha, 2\varepsilon, \ell, L)$-locally list recoverable for*

$$\tilde{Q} = \max\{Q \cdot Q_{\text{test}} \cdot O(|L| \log(|L|/\varepsilon)), Q \cdot Q_{\text{corr}}\}.$$

Moreover, if the approximate local list recovery algorithm has preprocessing time T_{pre} and running time T, and the local correction and tolerant local testing algorithms run in times $T_{\text{test}}, T_{\text{corr}}$, respectively, then the local list recovery algorithm has preprocessing time $T_{\text{pre}} + T \cdot T_{\text{test}} \cdot O(|L| \log(|L|/\varepsilon))$ and running time $T \cdot T_{\text{corr}}$.

4.2 Capacity-achieving locally list recoverable codes

In this section we prove the following lemma which shows the existence of capacity-achieving locally list recoverable codes. An analogous lemma was proven in [24, Lemma 5.3], however only for local decoding *message* coordinates, and without the soundness property. The fact that we are able to locally correct *codeword* coordinates, as well as guarantee the soundness property, will be crucial for our GV bound local correction application.

▶ **Lemma 30.** *For any constants $\rho \in [0, 1]$, $\gamma > 0$, $\varepsilon > 0$, and $\ell \geq 1$ there exists an infinite family of codes $\{C_N\}_N$ that satisfy the following.*

- *C_N is an $\mathbb{F}_2$-linear code of block length N and alphabet size $N^{o(1)}$.*
- *C_N has rate ρ and relative distance at least $1 - \rho - \gamma$.*
- *C_N is $(N^{o(1)}, 1 - \rho - \gamma, \varepsilon, \ell, N^{o(1)})$-locally list recoverable with preprocessing and running time $N^{o(1)}$.*
- *C_N is encodable in time $N^{1+o(1)}$.*

As in the proof of Lemma 23, we first use Lemma 25 to obtain *high-rate* locally list recoverable codes, and then use the Alon-Edmonds-Luby (AEL) distance amplification method [2, 3] to turn these codes into *capacity-achieving* locally list recoverable codes. However, this time we use a version of the AEL method for *local* list recovery from [17].

4.3 Local correction up to the GV bound

In this section we prove the following lemma which implies Corollary 5 from the introduction.

▶ **Lemma 31.** *For any constants $\rho \in [0, 0.02]$ and $\gamma > 0$ there exists an infinite family of binary linear codes $\{C_N\}_N$, where C_N has block length N and rate ρ, and is locally correctable from $\frac{H_2^{-1}(1-\rho)-\gamma}{2}$-fraction of errors with query complexity $N^{o(1)}$.*

Furthermore,

- *The local correction algorithm for C_N runs in time $N^{o(1)}$.*
- *There exists a randomized algorithm which, on input N, runs in time $N^{1+o(1)}$ and outputs with high probability a description of a code C_N with the properties above. Given the description, the code C_N can be encoded deterministically in time $N^{1+o(1)}$.*

The proof is analogous to that of Lemma 24 and relies on concatenation.

▶ **Lemma 32** (Concatenation for local list recovery). *Suppose that $C \subseteq (\Sigma^{\rho' \cdot t})^n$ is $(Q, \alpha, \varepsilon, \ell, L)$-locally list recoverable, and $C_{\mathrm{con}} \subseteq \Sigma^{tn}$ is a code obtained from C by applying a code $C^{(i)} \subseteq \Sigma^t$ of rate ρ' on each coordinate $i \in [n]$ of C. Suppose furthermore that at least $(1-\gamma)$-fraction of the codes $C^{(i)}$ are (α', ℓ', ℓ)-globally list recoverable. Then C_{con} is $(Q \cdot t, (\alpha - \gamma) \cdot \alpha', \varepsilon, \ell', L)$-locally list recoverable.*

Moreover, if the local list recovery algorithm for C has preprocessing time T_{pre} and running time T, and each $C^{(i)}$ can be globally list recovered in time T', then the local list recovery algorithm for C_{con} has preprocessing time $T_{\mathrm{pre}} + Q \cdot T'$ and running time $T + Q \cdot T'$.

References

1 Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Journal of algorithms*, 7(4):567–583, 1986.

2 Noga Alon, Jeff Edmonds, and Michael Luby. Linear Time Erasure Codes with Nearly Optimal Recovery. In *proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 512–519. IEEE Computer Society, 1995.

3 Noga Alon and Michael Luby. A linear time erasure-resilient code with nearly optimal recovery. *IEEE Transactions on Information Theory*, 42(6):1732–1736, 1996.

4 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking Computations in Polylogarithmic Time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–31. ACM Press, 1991. doi:10.1145/103418.103428.

5 Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Structures and Algorithms*, 28(4):387–402, 2006. doi:10.1002/rsa.20120.

6 Eli Ben-Sasson and Michael Viderman. Tensor Products of Weakly Smooth Codes are Robust. *Theory of Computing*, 5(1):239–255, 2009.

7 Eli Ben-Sasson and Michael Viderman. Composition of semi-LTCs by two-wise tensor products. *Computational Complexity*, 24(3):601–643, 2015. doi:10.1007/s00037-013-0074-8.

8 E. R. Berlekamp and L. Welch. Error correction of algebraic block codes. US Patent Number 4,633,470.

9 Don Coppersmith and Atri Rudra. On the robust testability of tensor products of codes. ECCC TR05-104, 2005. URL: https://eccc.weizmann.ac.il/eccc-reports/2005/TR05-104/index.html.

10 Irit Dinur, Madhu Sudan, and Avi Wigderson. Robust local testability of tensor products of LDPC codes. In *proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*, pages 304–315. Springer, 2006.

11 Katalin Friedl and Madhu Sudan. Some Improvements to Total Degree Tests. In *proceedings of the 3rd Israel Symposium on the Theory of Computing and Systems (ISTCS)*, pages 190–198. IEEE Computer Society, 1995.

12 Edgar N. Gilbert. A comparision of signalling alphabets. *Bell System Technical Journal*, 31:504–522, 1952.

13 Oded Goldreich. A sample of samplers: A computational perspective on sampling. *def*, 1:2n, 1997.

14 Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st annual ACM symposium on Theory of computing (STOC)*, pages 25–32. ACM Press, 1989.

15 Oded Goldreich and Or Meir. The tensor product of two good codes is not necessarily locally testable. *Information Processsing Letters*, 112(8-9):351–355, 2012.

16 Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost linear length. *Journal of ACM*, 53(4):558–655, 2006.

17 Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally Testable and Locally Correctable Codes approaching the Gilbert-Varshamov Bound. *IEEE Transactions on Information Theory*, 64(8):5813–5831, 2018.

18 Venkatesan Guruswami and Piotr Indyk. Near-optimal linear-time codes for unique decoding and new list-decodable codes over smaller alphabets. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 812–821. ACM Press, 2002.

19 Venkatesan Guruswami and Piotr Indyk. Efficiently decodable codes meeting Gilbert-Varshamov bound for low rates. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA)*, pages 756–757. SIAM, 2004. URL: `http://dl.acm.org/citation.cfm?id=982792`.

20 Venkatesan Guruswami and Swastik Kopparty. Explicit subspace designs. *Combinatorica*, 36(2):161–185, 2016. `doi:10.1007/s00493-014-3169-1`.

21 Venkatesan Guruswami and Atri Rudra. Explicit Codes Achieving List Decoding Capacity: Error-Correction With Optimal Redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

22 Venkatesan Guruswami and Chaoping Xing. List decoding Reed-Solomon, Algebraic-Geometric, and Gabidulin subcodes up to the Singleton bound. In *Proceedings of the 45th annual ACM symposium on Theory of Computing (STOC)*, pages 843–852. ACM, 2013.

23 Brett Hemenway, Noga Ron-Zewi, and Mary Wootters. Local List Recovery of High-Rate Tensor Codes & Applications. In *Proceedings of the 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 2017.

24 Brett Hemenway, Noga Ron-Zewi, and Mary Wootters. Local List Recovery of High-rate Tensor Codes & Applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:104 (revision 1), 2017.

25 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *STOC '00: Proceedings of the 32nd Annual Symposium on the Theory of Computing*, pages 80–86, 2000. `doi:10.1145/335305.335315`.

26 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-Rate Locally Correctable and Locally Testable Codes with Sub-Polynomial Query Complexity. *Journal of ACM*, 64(2):11:1–11:42, 2017. `doi:10.1145/3051093`.

27 Swastik Kopparty, Noga Ron-Zewi, Shubhangi Saraf, and Mary Wootters. Improved List Decoding of Folded Reed-Solomon and Multiplicity Codes. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 2018.

28 Or Meir. Combinatorial Construction of Locally Testable Codes. *SIAM Journal on Computing*, 39(2):491–544, 2009.

29 Irving S. Reed and Gustave Solomon. Polynomial Codes over Certain Finite Fields. *SIAM Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

30 Ronitt Rubinfeld and Madhu Sudan. Robust Characterization of Polynomials with Applications to Program Testing. *SIAM Journal of Computing*, 25(2):252–271, 1996.

31 Atri Rudra and Mary Wootters. Average-radius list-recoverability of random linear codes. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 644–662. SIAM, 2018.

32 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom Generators without the XOR Lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001. `doi:10.1006/jcss.2000.1730`.

33 Christian Thommesen. The existence of binary linear concatenated codes with Reed - Solomon outer codes which asymptotically meet the Gilbert- Varshamov bound. *IEEE Trans. Information Theory*, 29(6):850–853, 1983. `doi:10.1109/TIT.1983.1056765`.

34 Paul Valiant. The tensor product of two codes is not necessarily robustly testable. In *proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*, pages 472–481. Springer, 2005.

35 R. R. Varshamov. Estimate of the number of signals in error correcting codes. *Doklady Akadamii Nauk*, pages 739–741, 1957.

36 Michael Viderman. Strong LTCs with inverse poly-log rate and constant soundness. In *proceedings of the 54th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 330–339. IEEE Computer Society, 2013.

37 Michael Viderman. A combination of testability and decodability by tensor products. *Random Structures and Algorithms*, 46(3):572–598, 2015.

A Combinatorial lower bound on output list size

In this appendix, we first provide a *combinatorial* lower bound on the output list size for list recovering a high-rate tensor product $C^{\otimes t}$, even in the noiseless setting. In particular, we show that the output list size must be doubly-exponential in t. From this, we are able to deduce certain corollaries demonstrating that our algorithms nearly achieve optimal parameters.

Recall that given vectors $v_1 \in \mathbb{F}^{n_1}, v_2 \in \mathbb{F}^{n_2}, \ldots, v_t \in \mathbb{F}^{n_t}$, their tensor product $v_1 \otimes v_2 \otimes \cdots \otimes v_t$ is the t-dimensional box whose value in the $(i_1, i_2, \ldots, i_t) \in n_1 \times n_2 \cdots \times n_t$ coordinate is given by the product

$$(v_1 \otimes v_2 \otimes \cdots \otimes v_t)_{i_1,i_2,\ldots,i_t} = (v_1)_{i_1} \cdot (v_2)_{i_2} \cdots (v_t)_{i_t} \ .$$

For the special case of $t = 2$, the tensor product $v \otimes u$ can be thought of as the outer product vu^T.

We also record the following standard fact regarding tensor products.

▶ **Proposition 33.** *Let $v_1, \ldots, v_{t_1} \in \mathbb{F}^{n_1}$ and $u_1, \ldots, u_{t_2} \in \mathbb{F}^{n_2}$ be sets of linearly independent vectors. Then the collection $\{v_i \otimes u_j \mid i \in [t_1], j \in [t_2]\}$ is linearly independent in $\mathbb{F}^{n_1 \times n_2}$.*

A.1 Output list size for list recovering high-rate tensor codes

In this section we prove Theorem 6 from the introduction, which we restate here for convenience.

▶ **Theorem 6** (Output list size for list recovering high-rate tensor codes)**.** *Suppose that $C \subseteq \mathbb{F}^n$ is a linear code of rate $1-\gamma$, and that $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is $(0, \ell, L)$-list recoverable. Then $L \geq \ell^{1/\gamma^t}$.*

To prove this theorem, we first prove the following proposition. Informally speaking, we iteratively apply the Singleton bound to conclude that linear codes of rate $1-\gamma$ contain about $1/\gamma$ codewords with pairwise disjoint supports. Recall that, for a vector $v \in \mathbb{F}^n$, the support of v is $\mathrm{Supp}(v) = \{i \in [n] \mid v_i \neq 0\}$.

▶ **Proposition 34.** *Let $C \subseteq \mathbb{F}^n$ be a subspace of dimension k, and let r be a positive integer. Suppose that*

$$\left(1 - \tfrac{1}{r}\right) \cdot n + 1 \leq k \ . \tag{3}$$

Then there exist non-zero vectors $c_1, \ldots, c_r \in C$ such that for all $i \neq j$, $\mathrm{Supp}(c_i) \cap \mathrm{Supp}(c_j) = \emptyset$.

Proof. Let $m := n - k + 1$, and note that Condition (3) is equivalent to

$$(r-1)m \leq k - 1 \ .$$

Take a basis for C of the form $(e_1, u_1), \ldots, (e_k, u_k)$, where $e_i \in \mathbb{F}^k$ is the ith standard basis vector, and $u_1, \ldots, u_k \in \mathbb{F}^{n-k}$ are vectors. For $j = 1, \ldots, r-1$, we can find a nontrivial linear combination of the vectors $u_{(j-1)\cdot m+1}, \ldots, u_{j \cdot m}$ summing to zero, as they

are a (multi-)set of $m = n - k + 1$ vectors lying in $\mathbb{F}^{n-k}$. Taking this linear combination of $(e_{(j-1)\cdot m+1}, u_{(j-1)\cdot m+1}), \ldots, (e_{j\cdot m}, u_{j\cdot m})$, we obtain a nonzero vector whose support is contained in the interval $\{(j-1)\cdot m+1, \ldots, j\cdot m\}$; denote this vector by c_j. In this manner, we obtain $r-1$ nonzero vectors $c_1, \ldots, c_{r-1} \in C$ with pairwise disjoint support. Finally, we may add the vector $c_r := (e_k, u_k)$ to this collection, yielding r vectors, as desired. ◀

Next we prove Theorem 6, based on the above proposition.

Proof of Theorem 6. Let $r := 1/\gamma$, and recall wish to come up with ℓ^{r^t} codewords in $C^{\otimes t}$ that are contained in the output list for appropriately chosen input lists.

In order to accomplish this, we first use Proposition 34 to obtain a subset $C' \subseteq C$ of r nonzero codewords with pairwise disjoint support. We then consider the subset $C'' \subseteq C^{\otimes t}$ containing all tensor products $c_1 \otimes c_2 \otimes \cdots \otimes c_t$ of t (not necessarily distinct) codewords $c_1, \ldots, c_t \in C'$, and our main observation is that all these r^t tensor products are also nonzero with pairwise disjoint support. Finally, we let $B \subseteq \mathbb{F}$ be an arbitrary subset of size ℓ, and consider the subset $\bar{C} \subseteq C^{\otimes t}$ containing all linear combinations of codewords in C'' with coefficients in B. Since all codewords in C'' are nonzero with pairwise disjoint support, they are in particular linearly independent, so the set $\bar{C}$ contains ℓ^{r^t} distinct codewords in $C^{\otimes t}$.

Moreover, since codewords in C'' have pairwise disjoint support, for each coordinate $(i_1, \ldots, i_t) \in [n]^t$, there is at most one codeword $c \in C''$ for which $c_{i_1,\ldots,i_t}$ is nonzero. Therefore this is the only term which can contribute nontrivially to the value in the $(i_1, \ldots, i_t)$ coordinate of a codeword in $\bar{C}$. So we can let the corresponding input list $S_{i_1,\ldots,i_t}$ contain all ℓ multiples of $c_{i_1,\ldots,i_t}$ by elements in B. Details follow.

Since C has rate $1-\gamma$, it has dimension $k = (1-\gamma)n$, and so Proposition 34 guarantees the existence of a subset $C' \subseteq C$ of $r = 1/\gamma$ nonzero codewords with pairwise disjoint support.

Next we let

$$C'' := \left\{c_1 \otimes c_2 \otimes \cdots \otimes c_t \mid c_1, c_2, \ldots, c_t \in C'\right\}$$

be the subset of $C^{\otimes t}$ containing all tensor products of t (not necessarily distinct) codewords in C'. Since all codewords in C' are nonzero, their t-wise tensor products are nonzero as well.

To see that all codewords in C'' have pairwise disjoint support, suppose that $c = c_1 \otimes c_2 \otimes \ldots \otimes c_t \in C''$, and $(i_1, i_2, \ldots, i_t) \in \mathrm{Supp}(c)$. Then

$$0 \neq c_{i_1,i_2,\ldots,i_t} = (c_1)_{i_1} \cdot (c_2)_{i_2} \cdots (c_t)_{i_t},$$

so we must have that $(c_1)_{i_1}, (c_2)_{i_2}, \ldots, (c_t)_{i_t}$ are all nonzero. We conclude that

$$\mathrm{Supp}(c) \subseteq \mathrm{Supp}(c_1) \times \mathrm{Supp}(c_2) \times \cdots \times \mathrm{Supp}(c_t).$$

Now, suppose that $c = c_1 \otimes \ldots \otimes c_t$, $c' = c'_1 \otimes \ldots \otimes c'_t$ are a pair of codewords in C'' with $c_j \neq c'_j$ for some $j \in [t]$. Since all codewords in C' have pairwise disjoint support it must hold that $\mathrm{Supp}(c_j) \cap \mathrm{Supp}(c'_j) = \emptyset$, and we conclude that $\mathrm{Supp}(c) \cap \mathrm{Supp}(c') = \emptyset$.

Now, let $B \subseteq \mathbb{F}$ be an arbitrary subset of size ℓ, and let

$$\bar{C} := \left\{ \sum_{c \in C''} \beta_c \cdot c \;\middle|\; \beta_c \in B \text{ for all } c \in C'' \right\}$$

be the subset of $C^{\otimes t}$ containing all linear combinations of codewords in C'' with coefficients in B. Since all codewords in C'' are nonzero with pairwise disjoint support, they are in particular linearly independent in $\mathbb{F}^{n^t}$,[6] so the set $\bar{C}$ contains ℓ^{r^t} distinct codewords in $C^{\otimes t}$.

[6] This also follows from the fact that all codewords in C'' are linearly independent together with Proposition 33.

Finally, we wish to define input lists $S_{i_1,\dots,i_t}$ for any coordinate $(i_1,\dots,i_t) \in [n]^t$ so that for any codeword $c \in \bar{C}$, and for any coordinate $(i_1,\dots,i_t) \in [n]^t$, it holds that $c_{i_1,\dots,i_t} \in S_{i_1,\dots,i_t}$.

To this end, we observe that since codewords in C'' have pairwise disjoint support, for each coordinate $(i_1,\dots,i_t) \in [n]^t$, there is at most one codeword $c \in C''$ for which $c_{i_1,\dots,i_t}$ is nonzero. Therefore this is the only term which can contribute nontrivially to the value in the $(i_t,\dots,i_t)$ coordinate of a codeword in $\bar{C}$. So we can define the corresponding input list $S_{i_1,\dots,i_t}$ as

$$S_{i_1,\dots,i_t} := \{\beta \cdot c_{i_1,\dots,it} \mid \beta \in B\}$$

if such a codeword c exists, and as $S_{i_1,\dots,i_t} = \{0\}$ otherwise. Note that each set $S_{i_1,\dots,i_t}$ has size at most ℓ, and that they satisfy the required property.

This yields a set of ℓ^{r^t} codewords from $C^{\otimes t}$ that are contained in the output list for the input list tuple S defined above, proving the theorem. ◀

A.2 Concrete lower bound on output list size

In this section, we demonstrate a setting of parameters that yields Corollary 7 from the introduction, restated below.

▶ **Corollary 7.** *For any $\delta > 0$ and $\ell > 1$ there exists $L > 1$ such that the following holds for any sufficiently large n. There exists a linear code $C \subseteq \mathbb{F}^n$ of relative distance δ that is $(\Omega(\delta), \ell, L)$-list recoverable, but $C^{\otimes t} \subseteq \mathbb{F}^{n^t}$ is only $(0, \ell, L')$-list recoverable for $L' \geq \exp((2\delta)^{-(t-3/2)} \cdot \sqrt{\log L})$.*

We use the following result on the list-recoverability of random linear codes from [31].

▶ **Theorem 35** ([31], Corollary 3.3)**.** *There exists an absolute constant b_0 so that the following holds. For any $\gamma > 0$, $\ell \geq 1$, and a prime power $q \geq \ell^{b_0/\gamma}$, a random linear code $C \subseteq \mathbb{F}_q^n$ of rate $1-\gamma$ is $(\Omega(\gamma), \ell, L)$-list recoverable for*

$$L \leq \left(\frac{q\ell}{\gamma}\right)^{(\log \ell)/\gamma} \cdot \exp\left(\frac{\log^2 \ell}{\gamma^3}\right)$$

with probability $1 - \exp(-n)$.

Proof of Corollary 7. Let $C \subseteq \mathbb{F}_q^n$ be the linear code given by Theorem 35 of rate $1-2\delta$ and $q = \ell^{O(1/\delta)}$ that is $(\Omega(\delta), \ell, L)$-list recoverable for $L = \exp((\log^2 \ell)/\delta^3)$, or equivalently, $\ell = \exp(\delta^{3/2} \cdot \sqrt{\log L})$. By Corollary 10, we may further assume that the code C has relative distance at least δ. Now, by Theorem 6 we have that $L' \geq \ell^{(2\delta)^{-t}} = \exp((2\delta)^{-(t-3/2)} \cdot \sqrt{\log L})$. ◀

A.3 Lower bound for local list recovering

We now prove Corollary 8 from the introduction, restated below.

▶ **Corollary 8.** *For any $\delta > 0$ and sufficiently large n there exists a linear code $C \subseteq \mathbb{F}^n$ of relative distance δ such that the following holds. Suppose that $C^{\otimes t} \subseteq \mathbb{F}^N$ is $(\frac{1}{N}, 2, L)$-locally list recoverable with query complexity Q. Then $Q \cdot L \geq N^{\Omega_\delta(1/\log\log N)}$.*

We first show the following lemma which says that a locally list decodable (and in particular locally list recoverable) code with output list size L and query complexity Q is also locally correctable with query complexity roughly $Q \cdot L$.

▶ **Lemma 36.** *Suppose that $C \subseteq \Sigma^n$ is a code of relative distance δ that is $(Q, \alpha, 0.1, L)$-locally list decodable for $\alpha < \delta/2$. Then C is $\left(O\left(Q \cdot L \cdot \frac{\log^2 n}{(\delta/2-\alpha)^2}\right), \alpha\right)$-locally correctable.*

So to prove Corollary 8, it is enough to show a lower bound on the query complexity for local correcting $C^{\otimes t}$, assuming that the output list for list recovering $C^{\otimes t}$ is small. To show such a lower bound, we first observe that for any linear code C, the (absolute) distance of $C^\perp$ is a lower bound on the query complexity for local correcting C.

▶ **Lemma 37.** *Suppose that $C \subseteq \mathbb{F}^n$ is a linear code that is $(Q, \frac{1}{n})$-locally correctable. Then $Q \geq \Delta(C^\perp) - 2$.*

We prove the above lemma in Section A.3.1. To apply this lemma to $C^{\otimes t}$ we further observe that the tensor product preserves the dual distance of the base code.

▶ **Lemma 38.** *Suppose that $C_1 \subseteq \mathbb{F}^{n_1}$, $C_2 \subseteq \mathbb{F}^{n_2}$ are linear codes, and that $C_1^\perp, C_2^\perp$ have distances Δ_1, Δ_2, respectively. Then $(C_1 \otimes C_2)^\perp$ has distance $\min\{\Delta_1, \Delta_2\}$. In particular, if $C \subseteq \mathbb{F}^n$ is a linear code, and $C^\perp$ has distance Δ, then $(C^{\otimes t})^\perp$ has distance Δ for any $t \geq 1$.*

We prove the above lemma in Section A.3.2. We now proceed to the proof of Corollary 8.

Proof of Corollary 8. Let $C \subseteq \mathbb{F}^n$ be a random linear code of rate $1 - 2\delta$. By Corollary 10, for sufficiently large field size, the code C will have relative distance at least δ with high probability. Moreover, since $C^\perp$ has rate 2δ, by the same corollary we also have that $C^\perp$ has relative distance at least $1 - 3\delta$ with high probability. We conclude for any sufficiently large n the existence of a linear code $C \subseteq \mathbb{F}^n$ of rate $1 - 2\delta$ and relative distance at least δ such that $C^\perp$ has relative distance at least $1 - 3\delta$.

Next observe that for the code $C^{\otimes t}$ to be $(Q, \frac{1}{N}, 0.1, 2, L)$-locally list recoverable, it in particular must be $(0, 2, L)$-list recoverable, so the lower bound from Theorem 6 implies that $L \geq 2^{1/(2\delta)^t}$. Now, if $2^{1/(2\delta)^t} \geq N$ then we have that $Q \cdot L \geq 2^{1/(2\delta)^t} \geq N$, and we are done. So we may assume that $2^{1/(2\delta)^t} < N$ which implies in turn that $t = O_\delta(\log \log N)$ and $n = N^{1/t} = N^{\Omega_\delta(1/\log \log N)}$.

Moreover, as we have assumed we have a $(Q, \frac{1}{N}, 0.1, 2, L)$-local list recovery algorithm for $C^{\otimes t}$, we also have a $(Q, \frac{1}{N}, 0.1, L)$-local list decoding algorithm for $C^{\otimes t}$. Lemma 36 then promises that we have a $(O(Q \cdot L \cdot \frac{\log^2 N}{(\delta^t/2 - 1/N)^2}), \frac{1}{N})$-local correction algorithm for $C^{\otimes t}$. Now, by Lemma 38 we have that $(C^{\otimes t})^\perp$ has (absolute) distance at least $(1 - 3\delta)n$, and consequently Lemma 37 implies that

$$O\left(Q \cdot L \cdot \frac{\log^2 N}{(\delta^t/2 - \frac{1}{N})^2}\right) \geq (1 - 3\delta)n - 2 = N^{\Omega_\delta(1/\log \log N)} .$$

This implies $Q \cdot L \geq N^{\Omega_\delta(1/\log \log N)}$, as desired. ◀

A.3.1 Dual distance is a lower bound on query complexity – proof of Lemma 37

First, we recall the standard fact that (absolute) dual distance Δ implies that the uniform distribution over the code is $(\Delta - 1)$-wise independent.

▶ **Proposition 39** ([1])**.** *Suppose that $C \subseteq \mathbb{F}_q^n$ is a linear code, and that $C^\perp$ has (absolute) distance Δ. Then for all $1 \leq i_1 < \cdots < i_s \leq n$ with $s < \Delta$, and all $a_1, \ldots, a_s \in \mathbb{F}_q$,*

$$\Pr_{c \in C}\left[c_{i_1} = a_1 \wedge \cdots \wedge c_{i_s} = a_s\right] = \frac{1}{q^s}.$$

In what follows let $\Delta := \Delta(C^\perp)$, and let q denote the alphabet size of C. Now, making use of Yao's principle, it suffices to show a distribution $\mathcal{D}$ over vectors w at absolute distance at most 1 from C such that the following holds. For any *deterministic* algorithm making at most $\Delta - 2$ queries to its input w sampled according to $\mathcal{D}$, the probability that it correctly computes c_1 is at most $1/3$, where c is the unique codeword in C at absolute distance at most 1 from w. We will in fact show that no deterministic query algorithm can correctly compute c_1 with probability greater than $1/q$.

Let $\mathcal{D}$ denote the distribution that samples $c \in C$ uniformly at random and then sets $c_1 = 0$. Let A be a deterministic algorithm making at most $\Delta - 2$ queries, and let $j_1, \ldots, j_s \in [n]$ denote the queries made by A, where we assume $s \leq \Delta - 2$. Note that querying 1 does not help A, as it will always read 0. Hence, without loss of generality, $1 \notin \{j_1, \ldots, j_s\}$.

Now, by Proposition 39 and Bayes' rule, for any $b_1, \ldots, b_s, a \in \mathbb{F}_q$,

$$\Pr_{c \in C}[c_1 = a | c_{j_1} = b_1, \ldots, c_{j_s} = b_s] = \frac{\Pr[c_1 = a, c_{j_1} = b_1, \ldots, c_{j_s} = b_s]}{\Pr[c_{j_1} = b_1, \ldots, c_{j_s} = b_s]} = \frac{q^{-(s+1)}}{q^{-s}} = \frac{1}{q} .$$

Additionally, observe that the distribution of the tuple $(c_{j_1}, \ldots, c_{j_s})$ is the same if c is a uniformly random codeword from C or if it is sampled according to $\mathcal{D}$.

Hence, if we think of the query algorithm as implementing a (deterministic) function $g : \mathbb{F}_q^s \to \mathbb{F}_q$ from the responses to its queries to its guess for c_1, regardless of the responses $b_1, \ldots, b_s$ to the queries, we have

$$\Pr_{w \in \mathcal{D}}[c_1 = g(b_1, \ldots, b_s) | w_{j_1} = b_1, \ldots, w_{j_s} = b_s] = \frac{1}{q} ,$$

where c is the unique codeword in C for which $\mathrm{dist}(c, w) \leq \frac{1}{n}$. That is, the query algorithm will not be able to guess c_1 with probability greater than $1/q$, as claimed.

A.3.2 Tensor product preserves dual distance – proof of Lemma 38

First note that we clearly have that $\Delta((C_1 \otimes C_2)^\perp) \leq \min\{\Delta_1, \Delta_2\}$: for example, the matrix whose first column is a vector from $C_1^\perp$ of weight Δ_1 and all other columns are 0 gives a matrix in $(C_1 \otimes C_2)^\perp$ of weight Δ_1, and similarly a matrix in $(C_1 \otimes C_2)^\perp$ of weight Δ_2 can be constructed. We now establish the opposite inequality of $\Delta((C_1 \otimes C_2)^\perp) \geq \min\{\Delta_1, \Delta_2\}$.

It is well-known (and not hard to show) that the (absolute) distance of a code C is the minimum number of linearly dependent columns in a parity-check matrix for C. Furthermore, by duality we have that if G is a generating matrix for C then G^T is a parity-check matrix for $C^\perp$. We conclude that the distance of $C^\perp$ is the minimum number of linearly dependent rows in a generating matrix for C.

Let G_1, G_2 be generating matrices for C_1, C_2, respectively, and note that by the above, any collection of $t_1 < \Delta_1, t_2 < \Delta_2$ rows of G_1, G_2, respectively, are linearly independent. Next recall that $G_1 \otimes G_2$ is a generating matrix for $C_1 \otimes C_2$, and so it suffices to show that for any $t < \min\{\Delta_1, \Delta_2\}$, any collection of t rows of $G_1 \otimes G_2$ are linearly independent.

Let $u_1, u_2, \ldots, u_{n_1}$ and $v_1, v_2, \ldots, v_{n_2}$ denote the rows of G_1, G_2, respectively, and note that each row in $G_1 \otimes G_2$ is of the form $u_i \otimes v_j$ for some $i \in [n_1], j \in [n_2]$. Fix $t < \min\{\Delta_1, \Delta_2\}$, and suppose that $u_{i_1} \otimes v_{j_1}, u_{i_2} \otimes v_{j_2}, \ldots, u_{i_t} \otimes v_{j_t}$ is a collection of t rows of $G_1 \otimes G_2$. Then by the above we have that both collections $u_{i_1}, u_{i_2}, \ldots, u_{i_t}$ and $v_{j_1}, v_{j_2}, \ldots, v_{j_t}$ are linearly independent (ignoring duplications). Proposition 33 implies in turn that the collection $u_{i_1} \otimes v_{j_1}, u_{i_2} \otimes v_{j_2}, \ldots, u_{i_t} \otimes v_{j_t}$ are also linearly independent which concludes the proof of the lemma.

Approximate $\mathbb{F}_2$-Sketching of Valuation Functions

Grigory Yaroslavtsev
Indiana University, Bloomington, IN, USA
The Alan Turing Institute, London, UK
gyarosla@iu.edu

Samson Zhou
Indiana University, Bloomington, IN, USA
samsonzhou@gmail.com

Abstract

We study the problem of constructing a linear sketch of minimum dimension that allows approximation of a given real-valued function $f\colon \mathbb{F}_2^n \to \mathbb{R}$ with small expected squared error. We develop a general theory of linear sketching for such functions through which we analyze their dimension for most commonly studied types of valuation functions: additive, budget-additive, coverage, α-Lipschitz submodular and matroid rank functions. This gives a characterization of how many bits of information have to be stored about the input x so that one can compute f under additive updates to its coordinates.

Our results are tight in most cases and we also give extensions to the distributional version of the problem where the input $x \in \mathbb{F}_2^n$ is generated uniformly at random. Using known connections with dynamic streaming algorithms, both upper and lower bounds on dimension obtained in our work extend to the space complexity of algorithms evaluating $f(x)$ under long sequences of additive updates to the input x presented as a stream. Similar results hold for simultaneous communication in a distributed setting.

2012 ACM Subject Classification Theory of computation → Sketching and sampling

Keywords and phrases Sublinear algorithms, linear sketches, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.69

Category RANDOM

Related Version A full version of the paper is available at `https://arxiv.org/pdf/1907.00524.pdf`.

Acknowledgements We would like to thank Swagato Sanyal for multiple discussions leading to this paper, including the proof of Theorem 11 and Nikolai Karpov for his contributions to Section 3.1. We would also like to thank Amit Chakrabarti, Qin Zhang and anonymous reviewers for their comments.

1 Introduction

Linear sketching is a fundamental tool in efficient algorithm design that has enabled many of the recent breakthroughs in fast graph algorithms and computational linear algebra. It has a wide range of applications, including randomized numerical linear algebra (see survey [56]), graph sparsification (see survey [44]), frequency estimation [3], dimensionality reduction [34], various forms of sampling, signal processing, and communication complexity. In fact, linear sketching has been shown to be the optimal algorithmic technique [41, 2] for dynamic data streams, where elements can be both inserted and deleted. Linear sketching is also a frequently used tool in distributed computing – summaries communicated between the processors in massively parallel computational models are often linear sketches.

In this paper we introduce a study of approximate linear sketching over $\mathbb{F}_2$ (approximate $\mathbb{F}_2$-sketching). This is a previously unstudied but natural generalization of the work of [35], which studies exact $\mathbb{F}_2$-sketching. For a set $S \subseteq [n]$ let $\chi_S\colon \mathbb{F}_2^n \to \mathbb{F}_2$ be a parity function defined as $\chi_S(x) = \sum_{i \in S} x_i$. Given a function $f\colon \mathbb{F}_2^n \to \mathbb{R}$, we are looking for a distribution


© Grigory Yaroslavtsev and Samson Zhou;
licensed under Creative Commons License CC-BY



over k subsets $\mathbf{S}_1, \ldots, \mathbf{S}_k \subseteq [n]$ such that for any input x, it should be possible to compute $f(x)$ with expected squared error at most ϵ from the parities $\chi_{\mathbf{S}_1}(x), \chi_{\mathbf{S}_2}(x), \ldots, \chi_{\mathbf{S}_k}(x)$ computed over these sets. While looking only at linear functions over $\mathbb{F}_2$ as candidate sketches for evaluating f might seem restrictive, this view turns out to be optimal in a number of settings. In the light of recent results of [35, 32], the complexity of $\mathbb{F}_2$-sketching also characterizes the space complexity of streaming algorithms in the XOR-update model as well as communication complexity of one-way multiplayer broadcasting protocols for XOR-functions.

In matrix form, $\mathbb{F}_2$-sketching corresponds to multiplication over $\mathbb{F}_2$ of the row vector $x \in \mathbb{F}_2^n$ by a random $n \times k$ matrix whose i-th column is the characteristic vector of $\chi_{\mathbf{S}_i}$:

$$\begin{pmatrix} x_1 & x_2 & \ldots & x_n \end{pmatrix} \begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ \chi_{\mathbf{S}_1} & \chi_{\mathbf{S}_2} & \cdots & \chi_{\mathbf{S}_k} \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} = \begin{pmatrix} \chi_{\mathbf{S}_1}(x) & \chi_{\mathbf{S}_2}(x) & \ldots & \chi_{\mathbf{S}_k}(x) \end{pmatrix}$$

The goal is to minimize k, ensuring that the sketch alone is sufficient for computing f with expected squared error at most ϵ for any fixed input x. For a fixed distribution D of x, the definition of error is modified to include an expectation over D in the error guarantee. We give formal definitions below.

▶ **Definition 1** (Exact $\mathbb{F}_2$-sketching, [35]). *The* exact randomized $\mathbb{F}_2$-sketch complexity *with error δ of a function $f\colon \mathbb{F}_2^n \to \mathbb{R}$ (denoted as $R_\delta^{lin}(f)$) is the smallest integer k such that there exists a distribution $\chi_{\mathbf{S}_1}, \chi_{\mathbf{S}_2}, \ldots, \chi_{\mathbf{S}_k}$ over k linear functions over $\mathbb{F}_2^n$ and a post-processing function $g : \mathbb{F}_2^k \to \mathbb{R}$ that satisfies:*

$$\forall x \in \mathbb{F}_2^n\colon \Pr_{\mathbf{S}_1,\ldots,\mathbf{S}_k} [g(\chi_{\mathbf{S}_1}(x), \chi_{\mathbf{S}_2}(x), \ldots, \chi_{\mathbf{S}_k}(x)) = f(x)] \geq 1 - \delta.$$

The number of parities k in the definition above is referred to as the *dimension* of the $\mathbb{F}_2$-sketch.

▶ **Definition 2** (Approximate $\mathbb{F}_2$-sketching). *The* ϵ-approximate randomized $\mathbb{F}_2$-sketch complexity *of a function $f\colon \mathbb{F}_2^n \to \mathbb{R}$ (denoted as $\bar{R}_\epsilon^{lin}(f)$) is the smallest integer k such that there exists a distribution $\chi_{\mathbf{S}_1}, \chi_{\mathbf{S}_2}, \ldots, \chi_{\mathbf{S}_k}$ over k linear functions over $\mathbb{F}_2^n$ and a post-processing function $g : \mathbb{F}_2^k \to \mathbb{R}$ that satisfies:*

$$\forall x \in \mathbb{F}_2^n\colon \mathop{\mathbb{E}}_{\mathbf{S}_1,\ldots,\mathbf{S}_k} \left[(g(\chi_{\mathbf{S}_1}(x), \chi_{\mathbf{S}_2}(x), \ldots, \chi_{\mathbf{S}_k}(x)) - f(x))^2\right] \leq \epsilon$$

If g is an unbiased estimator of f, then this corresponds to an upper bound on the variance of the estimator. For example, functions with small spectral norm (e.g. coverage functions, [57]) admit such approximate $\mathbb{F}_2$-sketches. Moreover, observe that Definition 2 is not quite comparable with an epsilon-delta guarantee, which only promises that $|g(\chi_{\mathbf{S}_1}(x), \chi_{\mathbf{S}_2}(x), \ldots, \chi_{\mathbf{S}_k}(x)) - f(x)| \leq \epsilon$ with probability $1 - \delta$, but guarantees nothing for δ fraction of the inputs.

In addition to this worst-case guarantee, we also consider the same problem for x from a certain distribution. In this case, a weaker guarantee is required, i.e. the bound on expected squared error should hold only over some fixed known distribution D. An important case is $\mathsf{D} = U(\mathbb{F}_2^n)$, the uniform distribution over all inputs.

▶ **Definition 3** (Approximate distributional $\mathbb{F}_2$-sketching). *For a function $f\colon \mathbb{F}_2^n \to \mathbb{R}$, we define its* ϵ-approximate randomized distributional $\mathbb{F}_2$-sketch complexity *with respect to a distribution D over $\mathbb{F}_2^n$ (denoted as $\bar{\mathcal{D}}_\epsilon^{lin,D}(f)$) as the smallest integer k such that there exists a distribution $\chi_{\mathbf{S}_1}, \chi_{\mathbf{S}_2}, \ldots, \chi_{\mathbf{S}_k}$ over k linear functions over $\mathbb{F}_2$ and a post-processing function $g : \mathbb{F}_2^k \to \mathbb{F}_2$ that satisfies:*

$$\mathop{\mathbb{E}}_{x\sim D}\mathop{\mathbb{E}}_{\mathbf{S}_1,\ldots,\mathbf{S}_k}\left[(g(\chi_{\mathbf{S}_1}(x),\chi_{\mathbf{S}_2}(x),\ldots,\chi_{\mathbf{S}_k}(x))-f(x))^2\right]\leq\epsilon.$$

1.1 Applications to Streaming and Distributed Computing

One of the key applications of our results is to the dynamic streaming model. In this model, the input x is generated via a sequence of additive updates to its coordinates, starting with $x = 0^n$. If $x \in \mathbb{R}^n$, then updates are of the form (i, Δ_i) (turnstile model), where $i \in [n]$, and $\Delta_i \in \mathbb{R}$, which adds Δ_i to the i-th coordinate of x. For $x \in \mathbb{F}_2^n$, only the coordinate i is specified and the corresponding bit is flipped, which is known as the XOR-update model [52][1]. Dynamic streaming algorithms aim to minimize space complexity of computing a given function f for an input generated through a sequence of such updates while also ensuring fast update and function evaluation times.

Note that linear sketching over the reals and $\mathbb{F}_2$-sketching can be used directly in the respective streaming update models. Most interestingly, these techniques turn out to achieve almost optimal space complexity. It is known that linear sketching over the reals gives (almost) optimal space complexity for processing dynamic data streams in the turnstile model for *any* function f [41, 2]. However, the results of [41, 2] require adversarial streams of length triply exponential in n. In the XOR-update model, space optimality of $\mathbb{F}_2$-sketching has been shown recently in [32]. This optimality result holds even for adversarial streams of much shorter length $\Omega(n^2)$. Hence, lower bounds on $\mathbb{F}_2$-sketch complexity obtained in our work extend to space complexity of dynamic streaming algorithms for streams of quadratic length.

A major open question in this area is the conjecture of [35] that the same holds even for streams of length only $2n$. We thus complement our lower bounds on dimension of $\mathbb{F}_2$-sketches with one-way two-player communication complexity lower bounds for the corresponding XOR functions $f^+(x, y) = f(x + y)$. Such lower bounds translate to dynamic streaming lower bounds for streams of length $2n$. Furthermore, whenever our communication lower bounds hold for the uniform distribution, the corresponding streaming lower bound applies to streaming algorithms under uniformly random input updates.

Finally, our upper bounds can be used for distributed algorithms computing $f(x_1 + \cdots + x_M)$ over a collection of distributed inputs $x_1, \ldots, x_M \in \mathbb{F}_2^n$ as $\mathbb{F}_2$-sketches can be used for distributed inputs. On the other hand, our communication lower bounds also apply to the simultaneous message passing (SMP) communication model, since it is strictly harder than one-way communication.

[1] By slightly changing the function to $f'(x_1, \ldots, x_n, y_1, \ldots, y_n) = f(x_1 + y_1, x_2 + y_2, \ldots, x_n + y_n)$, it is easy to see that there are functions for which knowledge of the sign of the update (i.e. whether it is +1 or -1) is not a stronger model than the XOR-update model. For some further motivation of the XOR-update model, consider dynamic graph streaming algorithms, i.e the setting when x represents the adjacency matrix of a graph and updates correspond to adding and removing the edges. Almost all known dynamic graph streaming algorithms (except spectral graph sparsification of [36]) are based on the ℓ_0-sampling primitive [26]. As shown recently, ℓ_0-sampling can be implemented optimally using $\mathbb{F}_2$-sketches [37] and hence almost all known dynamic graph streaming algorithms can handle XOR-updates, i.e. knowing whether an edge was inserted or deleted does not help.

1.2 Valuation Functions and Sketching

Submodular valuation functions, originally introduced in the context of algorithmic game theory and optimization, have received a lot of interest recently in the context of learning theory [10, 9, 18, 29, 47, 23, 22, 24, 25][2], approximation [27, 7] and property testing [16, 49, 12]. As we show in this work, valuation functions also represent an interesting study case for linear sketching and streaming algorithms. While a variety of papers exists on streaming algorithms for optimizing various submodular objectives, e.g. [48, 20, 8, 17, 15, 21, 30, 4, 11], to the best of our knowledge no prior work considers the problem of evaluating such functions under a changing input.

A systematic study of $\mathbb{F}_2$-sketching has been initiated for Boolean functions in [35]. This paper can be seen as a next step, as we introduce approximation into the study of $\mathbb{F}_2$-sketching. One of the consequences of our work is that the Fourier ℓ_1-sampling technique, originally introduced by Bruck and Smolensky [14] (see also [28, 45]), turns out to be optimal in its dependence on both spectral norm and the error parameter. For Boolean functions, a corresponding result is not known as Boolean functions with small spectral norm and necessary properties are hard to construct. Another technical consequence of our work is that the study of learning and sketching algorithms turn out to be related on a technical level despite pursuing different objectives (in learning the specific function is unknown, while in sketching it is). In particular, our hardness result for Lipschitz submodular functions uses a construction of a large family of matroids from [10] (even though in a very different parameter regime), who designed such a family to fool learning algorithms.

1.3 Our Results

A function $f\colon 2^{[n]} \to \mathbb{R}$ is α-Lipschitz if for any $S \subseteq [n]$ and $i \in [n]$, it holds that $|f(S \cup \{i\}) - f(S)| \leq \alpha$ for some constant $\alpha > 0$. A function $f\colon 2^{[n]} \to \mathbb{R}$ is submodular if:

$$f(A \cup \{i\}) - f(A) \geq f(B \cup \{i\}) - f(B) \qquad \forall A \subseteq B \subseteq [n] \text{ and } i \notin B.$$

We consider the following classes of valuation functions of the form $f\colon \mathbb{F}_2^n \to \mathbb{R}$ (all of them submodular) sometimes treating them as $f\colon 2^{[n]} \to \mathbb{R}$ and vice versa. These classes mostly cover all of existing literature on submodular functions[3]. See Table 1 for a summary of the results.

- **Additive (linear).** $f(x) = \sum_{i=1}^n w_i x_i$, where $w_i \in \mathbb{R}$.
  Our results: For additive functions, it is easy to show that dimension of $\mathbb{F}_2$-sketches is $O(\min(\|w\|_1^2/\epsilon, n))$ and give a matching communication lower bound for all $\epsilon \geq \|w\|_2^2$ [57].
- **Budget-additive.** $f(x) = \min(b, \sum_{i=1}^n w_i x_i)$ where $b, w_i \in \mathbb{R}$. An example of such functions is the "hockey stick" function $hs_\alpha(x) = \min(\alpha, \frac{2\alpha}{n}\sum_{i=1}^n x_i)$.
  Our results: For budget-additive functions, it is easy to show that dimension of $\mathbb{F}_2$-sketches is $O(\min(\|w\|_1^2/\epsilon, n))$ [57]. We give a matching communication bound for the "hockey stick" function for constant ϵ [57], which holds even under the uniform distribution of the input.

[2] We remark that in this literature the term "sketching" is used to refer to the space complexity of representing the function f itself under the assumption that it is unknown but belongs to a certain class. This question is orthogonal to our work as we assume f is known and fixed while the input x is changing.

[3] We do not discuss some other subclasses of subadditive functions because they are either superclasses of classes for which we already have an $\Omega(n)$ lower bound (e.g. submodular, subadditive, etc.) or because such a lower bound follows trivially (e.g. for OXS/XOS since for XS-functions a lower bound of $\Omega(n)$ is easy to show, see [57]).

Table 1 Linear sketching complexity of classes of valuation functions. We defer the proofs of several results to the full version [57].

Class	Error	Distribution	Complexity	Result
Additive/Budget additive $\min(b, \sum_{i=1}^{n} w_i x_i)$	ϵ	any	$\Theta\left(\frac{\|w\|_1^2}{\epsilon}\right)$	[57]
$\min(c\sqrt{n}, \frac{2c}{\sqrt{n}} \sum_{i=1}^{n} x_i)$	constant	uniform	$\Omega(n)$	[57]
Coverage	ϵ	any	$O\left(\frac{1}{\epsilon}\right)$	[57]
Matroid Rank 2	exact	any	$\Theta(1)$	Theorem 12
Graphic Matroids Rank r	exact	any	$O(r^2 \log r)$	Theorem 16
Matroid Rank r	exact	any	$\Omega(r)$	Corollary 35
Matroid Rank r	exact	uniform	$O((r \log r + c)^{r+1})$	[57]
Matroid Rank	$1/\sqrt{n}$	uniform	$\Theta(1)$	[57]
$\frac{c}{n}$-Lipschitz Submodular	constant	any	$\Theta(n)$	Theorem 28

- **Coverage.** A function f is a *coverage function* on some universe U of size m if there exists a collection $A_1, \ldots, A_n$ of subsets of U and a vector of non-negative weights $(w_1, \ldots, w_m)$ such that:

$$f(S) = \sum_{i \in \cup_{j \in S} A_j} w_i.$$

Our results: We show a simple upper bound of $O(1/\epsilon)$ for such functions [57].

- **Matroid rank.** A pair $M = ([n], \mathcal{I})$ is called a matroid if $\mathcal{I} \subseteq 2^{[n]}$ is a non-empty set family such that the following two properties are satisfied:
  - If $I \in \mathcal{I}$ and $J \subseteq I$, then $J \in \mathcal{I}$
  - If $I, J \in \mathcal{I}$ and $|J| < |I|$, then there exists an $i \in I \setminus J$ such that $J \cup \{i\} \in \mathcal{I}$.

The sets in $\mathcal{I}$ are called *independent*. A maximal independent set is called a *base* of M. All bases have the same size, which is called the *rank* of the matroid and is denoted as $rk(M)$. The *rank function* of the matroid is the function $rank_M \colon 2^{[n]} \to \mathbb{N}_+$ defined as:

$$rank_M(S) := \max\{|I| \colon I \subseteq S, I \in \mathcal{I}\}.$$

It follows from the definition that $rank_M$ is always a submodular 1-Lipschitz function.
Our results: In order to have consistent notation with the rest of the manuscript we always assume that matroid rank functions are scaled so that their values are in $[0, 1]$. Some of our results are exact, i.e. the corresponding matroid rank function is computed exactly (and in this case rescaling does not matter) while others allow approximation of the function value. In the latter case, the approximation guarantees are multiplicative with respect to the rescaled function.
Our main theorem regarding sketching of matroid rank functions is as follows:

▶ **Theorem 4** (Sketching matroid rank functions). *For (scaled) matroid rank functions:*

- *There exists an exact $\mathbb{F}_2$-sketch of size $O(1)$ for matroids of rank 2 (Theorem 12) and graphic matroids (Theorem 16).*
- *There exists $c = \Omega(1)$ and a matroid of rank r such that a c-approximation of its matroid rank function has randomized linear sketch complexity $\Omega(r)$. Furthermore, this lower bound also holds for the corresponding one-way communication problem (Theorem 34, Corollary 35).*

This can be contrasted with the results under the uniform distribution for which matroids of rank r have an exact $\mathbb{F}_2$-sketch of size $O\left(\left(r\log r+\log\frac{1}{\epsilon}\right)^{r+1}\right)$, where ϵ is the probability of failure ([57], follows from the junta approximation of [13]). Furthermore, matroids of high rank $\Omega(n)$ can be trivially approximately sketched under product distributions, due to their concentration around their expectation (see [57] for details).

- **Lipschitz submodular.** A function $f\colon 2^{[n]}\to\mathbb{R}$ is α-Lipschitz submodular if it is both submodular and α-Lipschitz.
  Our results: We show an $\Omega(n)$ communication lower bound (and hence a lower bound on $\mathbb{F}_2$-sketch complexity) for constant error for monotone non-negative $O(1/n)$-Lipschitz submodular functions (Theorem 28). We note that this hardness result crucially uses a non-product distribution over the input variables since Lipschitz submodular functions are tightly concentrated around their expectation under product distributions (see e.g. [55, 10]) and hence can be approximated using their expectation without any sketching at all.

1.4 Overview and Techniques

1.4.1 Basic Tools: XOR Functions, Spectral Norm, Approximate Fourier Dimension

In Section 2, we introduce the basics of approximate $\mathbb{F}_2$-sketching. Most definitions and results in this section can be seen as appropriate generalizations regarding Boolean functions (such as in [35]) to the case of real-valued functions where we replace Hamming distance with expected squared distance. We then define the randomized one-way communication complexity of the two-player XOR-function $f^+(x,y)=f(x+y)$ corresponding to f. This communication problem plays an important role in our arguments as it gives a lower bound on the sketching complexity of f. We then introduce the notion of approximate Fourier dimension developed in [35]. The key structural results of [35], which characterize both the sketching complexity of f and the one-way communication complexity of f^+ under the uniform distribution using the approximate Fourier dimension, can be extended to the real-valued case as shown in Proposition 10 and Theorem 11. This characterization is our main tool for showing lower bounds under the uniform distribution of x.

Another useful basic tool is a bound on the linear sketching complexity based on the spectral norm of f, which we develop in [57]. In particular, as we show in [57], analogously to the Boolean case, we can leverage properties of the Fourier coefficients of a function f to show that the ϵ-approximate randomized sketching complexity of f is at most $O(\|\hat{f}\|_1^2/\epsilon)$. Thus, we can determine the dimension of $\mathbb{F}_2$-sketches for classes of functions whose spectral norms are well-bounded as well as functions which can be computed as Lipschitz compositions of a small number of functions with bounded spectral norm [57]. Examples of such classes include additive (linear), budget-additive and coverage functions. Finally, we argue that the dependence on the parameters in the spectral norm bound cannot be substantially improved in the real-valued case by presenting a subclass of linear functions which require sketches of size $\Omega(\|\hat{f}\|_1^2/\epsilon)$ [57]. This is in contrast with the case of Boolean functions studied in [35] for which such tightness result is not known.

1.4.2 Matroid Rank Functions, LTF, LTF∘OR

In Section 3, we present our results on sketching matroid rank and Lipschitz submodular functions. In Section 3.1 we show that matroid rank functions of matroids of rank 2 and graphic matroids have constant randomized sketching complexity. This is done by first

observing that rank functions of such matroids can be expressed as a threshold function over a number of disjunctions. Therefore, it remains to determine the sketching complexity of the threshold function on a collection of disjunctions. Unfortunately, known upper bounds for the sketching complexity of even the simpler class of linear threshold functions have a dependence on n and hence one cannot get a constant upper bound directly.

Hence we show how to remove this dependence in Section 3.1.1, also resolving an open question of Montanaro and Osborne [45]. Recall that a linear threshold function (LTF) can be represented as $f(x) = \mathsf{sgn}\left(\sum_{i=1}^{n} w_i x_i - \theta\right)$ for some weights w_i and threshold θ, where we slightly alter the traditional definition of the sign function sgn to output 0 if the input is negative and 1 otherwise. An important parameter of an LTF is its *margin* m, which corresponds to the difference between the threshold and the value of the linear combination closest to it. We first observe that the terms with insignificant coefficients, i.e. weights that are small in absolute value, do not contribute to the final output and thus, we can ignore them. Similarly, the remaining weights can be rounded, without altering the output of the function, to a collection of weights whose size is bounded, independent of n. Furthermore, $f(x) = 0$ only if $x_i = 1$ for at most $\frac{\theta}{2m}$ of these "significant" indices i of x. Thus, we hash the significant indices to a large, but independent of n, number of buckets. As a result, either there are a small number of significant indices i with $x_1 = 1$ and there are no collisions, or there is a large number of significant indices i with $x_i = 1$. Since we can differentiate between these two cases, the sketch can output whether $f(x) = 0$ or $f(x) = 1$ with constant probability. With a more careful choice of hash functions this idea can be extended to linear thresholds of disjunctions. We show in Section 3.1.2 that a threshold function over a number of disjunctions (LTF∘OR) also has linear sketch complexity independent of n.

In Section 3.2.1, we show that there exists an $\Omega(n)$-Lipschitz submodular function f that requires a randomized linear sketch of size $\Omega(n)$. We construct such a function probabilistically by using a large family of matroid rank functions constructed by [10] with an appropriately chosen set of parameters. We show any fixed deterministic sketch fails on a matroid chosen uniformly at random from this parametric family with very high probability. In fact, even if we take a union bound over all possible sketches of bounded dimension, the failure of probability is still negligibly close to 1. By Yao's principle, the randomized linear sketch complexity follows. We then extend this result to a communication lower bound for f^+ in Section 3.2.2. In the one-way communication complexity setting, we show that there exists an $\Omega(n)$-Lipschitz submodular function f whose f^+ requires communication $\Omega(n)$.

1.4.3 Uniform Distribution

In [57], we show lower bounds for a budget additive "hockey stick" function under the uniform distribution. The lower bounds follow from a characterization of communication complexity using approximate Fourier dimension, and to complete the analysis, we lower bound the Fourier spectrum of the hockey stick function in [57]. Although our approach for matroids of rank 2 does not seem to immediately generalize to matroids of higher rank under arbitrary distributions, we show in [57] that under the uniform distribution, we can use ϵ-approximations of disjunctive normal forms (DNFs) by juntas to obtain a randomized linear sketch whose size is independent of n. Furthermore, rank functions of matroids of very high rank admit trivial *approximate* sketches under the uniform distribution as follows from standard concentration results [55] (see [57]).

2 Basics of Approximate $\mathbb{F}_2$-Sketching

2.1 Communication Complexity of XOR functions

In order to analyze the optimal dimension of $\mathbb{F}_2$-sketches, we need to introduce a closely related communication complexity problem. For $f\colon \mathbb{F}_2^n \to \mathbb{R}$ define the XOR-function $f^+\colon \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{R}$ as $f^+(x,y) = f(x+y)$ where $x, y \in \mathbb{F}_2^n$. Consider a communication game between two players Alice and Bob holding inputs x and y respectively. Given access to a shared source of random bits Alice has to send a single message to Bob so that he can compute $f^+(x,y)$. This is known as the one-way communication complexity problem for XOR-functions (see [50, 58, 45, 39, 40, 51, 42, 53, 43, 31, 35] for related communication complexity results).

▶ **Definition 5** (Randomized one-way communication complexity of XOR function)**.** *For a function $f\colon \mathbb{F}_2^n \to \mathbb{R}$, the* randomized one-way communication complexity *with error δ (denoted as $R^{\rightarrow}_{\delta}(f^+)$) of its XOR-function is defined as the smallest size*[4] *(in bits) of the (randomized using public randomness) message $M(x)$ from Alice to Bob, which allows Bob to evaluate $f^+(x,y)$ for any $x, y \in \mathbb{F}_2^n$ with error probability at most δ.*

It is easy to see that $R^{\rightarrow}_{\delta}(f^+) \leq R^{lin}_{\delta}(f)$ as using shared randomness Alice can just send k bits $\chi_{\mathbf{s}_1}(x), \chi_{\mathbf{s}_2}(x), \ldots, \chi_{\mathbf{s}_k}(x)$ to Bob, who can for each $i \in [k]$ compute $\chi_{\mathbf{s}_i}(x+y) = \chi_{\mathbf{s}_i}(x) + \chi_{\mathbf{s}_i}(y)$, which is an $\mathbb{F}_2$-sketch of f on $x+y$ and hence suffices for computing $f^+(x,y)$ with probability $1-\delta$.

Replacing the guarantee of exactness of the output in the above definition with an upper bound on expected squared error, we obtain the following definition.

▶ **Definition 6** (Randomized one-way communication complexity of approximating an XOR function)**.** *For a function $f\colon \mathbb{F}_2^n \to \mathbb{R}$, the* randomized one-way communication complexity *(denoted as $\bar{R}^{\rightarrow}_{\epsilon}(f^+)$) of approximating its XOR-function with error ϵ is defined as the smallest size(in bits) of the (randomized using public randomness) message $M(x)$ from Alice to Bob, which allows Bob to evaluate $f^+(x,y)$ for any $x, y \in \mathbb{F}_2^n$ with expected squared error at most ϵ.*

Distributional communication complexity is defined analogously for the corresponding XOR function and is denoted as $\mathcal{D}_{\epsilon}$.

Finally, in the simultaneous model of computation [6, 5], also called simultaneous message passing (SMP) model, there exist two players and a coordinator, who are all aware of a function f. The two players maintain x and y respectively, and must send messages of minimal size to the coordinator so that the coordinator can compute $f(x \oplus y)$.

▶ **Definition 7** (Simultaneous communication complexity of XOR function)**.** *For a function $f\colon \mathbb{F}_2^n \to \mathbb{R}$, the* simultaneous one-way communication complexity *with error δ (denoted as $R^{sim}_{\delta}(f^+)$) of its XOR-function is defined as the smallest sum of the sizes (in bits) of the (randomized using public randomness) messages $M(x)$ and $M(y)$ from Alice and Bob, respectively, to a coordinator, which allows the coordinator to evaluate $f^+(x,y)$ for any $x, y \in \mathbb{F}_2^n$ with error probability at most δ.*

Observe that a protocol for randomized one-way communication complexity of XOR function translates to a protocol for the simultaneous model of computation.

[4] Formally the minimum here is taken over all possible protocols where for each protocol the size of the message $M(x)$ refers to the largest size (in bits) of such message taken over all inputs $x \in \mathbb{F}_2^n$. See [38] for a formal definition.

2.2 Distributional Approximate $\mathbb{F}_2$-Sketch Complexity

Fourier analysis plays an important role in the analysis of distributional $\mathbb{F}_2$-sketch complexity over the uniform distribution. In our discussion below, we make use of some standard facts from Fourier analysis of functions over $\mathbb{F}_2^n$. For definitions and basics of Fourier analysis of functions of such functions we refer the reader to the standard text [46] and [57]. In particular, Fourier concentration on a low-dimensional subspace implies existence of a small sketch which satisfies this guarantee:

▶ **Definition 8** (Fourier concentration). *A function $f\colon \mathbb{F}_2^n \to \mathbb{R}$ is γ-concentrated on a linear subspace A_d of dimension d if for this subspace it satisfies:*

$$\sum_{S \in A_d} \hat{f}(S)^2 \geq \gamma.$$

We also use the following definition of approximate Fourier dimension from [35], adapted for the case of real-valued functions.

▶ **Definition 9** (Approximate Fourier dimension). *Let $\mathcal{A}_k$ be the set of all linear subspaces of $\mathbb{F}_2^n$ of dimension k. For $f\colon \mathbb{F}_2^n \to \mathbb{R}$ and $\epsilon \in (0, \|f\|_2^2]$ the ϵ-approximate Fourier dimension $\dim_\epsilon(f)$ is defined as:*

$$\dim_\epsilon(f) = \min_k \left\{ \exists A \in \mathcal{A}_k \colon \sum_{\alpha \in A} \hat{f}^2(\alpha) \geq \epsilon \right\}.$$

▶ **Proposition 10.** *For any $f\colon \mathbb{F}_2^n \to \mathbb{R}$, it holds that:*

$$\bar{\mathcal{D}}_\epsilon^{lin,U}(f) \leq \dim_{\|f\|_2^2 - \epsilon}(f).$$

Furthermore, approximate Fourier dimension can be used as a lower bound on the one-way communication complexity of the corresponding XOR-function. We defer the proof of the following result to [57] as it is follows closely an analogous result for Boolean functions from [35].

▶ **Theorem 11.** *For any $f\colon \mathbb{F}_2^n \to \mathbb{R}$, $\delta \in [0, 1/2]$ and $\xi = \|f\|_2^2 - \epsilon(1 + 2\delta)$ it holds that:*

$$\bar{\mathcal{D}}_\epsilon^{\to,U}(f^+) \geq \frac{\delta}{2} \cdot \dim_\xi(f).$$

3 Sketching Matroid Rank Functions

In this section we analyze sketching complexity of matroid rank functions. We start by considering the most fundamental possible matroids (of rank 2) in Section 3.1 and showing that exactly sketching the matroid rank function requires $O(1)$ complexity. Similarly, we show that exactly sketching the rank of graphic matroids only uses $O(1)$ complexity. On the other hand, we show a lower bound in Section 3.2.1 that even approximating the rank r of general matroids up to certain constant factors requires $\Omega(r)$ complexity.

To sketch matroids of rank 2, we leverage a result by Acketa [1] which characterizes the collection of independent sets of such matroids. This allows us to represent matroid rank functions for matroids of rank 2 as a linear threshold of disjunctions. Thus, we first show the randomized linear sketch complexity of (θ, m)-linear threshold functions, resolving an open question by Montanaro and Osborne [45].

3.1 Matroids of Rank 2 and Graphic Matroids

In this section, we show that there exists a constant-size sketch that can be used to compute exact values of matroid rank functions for matroids of rank 2.

▶ **Theorem 12.** *For every matroid M of rank 2 it holds that $R^{lin}_{\frac{1}{3}}(rank_M) = O(1)$.*

It is well-known that matroids of rank 2 admit the following characterization (see e.g. [1]).

▶ **Fact 13.** *The collection of size 2 independent sets of a rank 2 matroid can be represented as the edges in a complete graph that has edges of some number of disjoint cliques removed.*

We define the following function as a threshold on the Hamming weight of a binary vector x

$$\mathsf{HAM}_{\leq d}(x) = \begin{cases} 0, & \text{if } \sum_{i=1}^{n} x_i \leq d + \frac{1}{2} \\ 1, & \text{otherwise.} \end{cases}$$

We use a series of technical lemmas in the following section to prove the following result, which says that linear threshold functions can be succinctly summarized:

▶ **Theorem 14.** *The function $\mathsf{HAM}_{\leq d}\left(\bigvee_{i\in S_1} x_i, \bigvee_{i\in S_2} x_i, \ldots\right)$ has a randomized linear sketch of size $O(d^2 \log d)$.*

The following fact that upper bounds the sketch complexity for functions with small support:

▶ **Fact 15** (Folklore, see e.g. [45, 35])**.** *For any function $f : \{0,1\}^n \to \{0,1\}$ with $\min_{z\in\{0,1\}} \mathbf{Pr}_{x\in\{0,1\}^n}(f(x) = z) \leq \epsilon$ it holds that $R^{lin}_{\delta}(f) \leq \log \frac{2^{n+1}\epsilon}{\delta}$.*

Using Fact 13, Theorem 14, and Fact 15, we prove Theorem 12 by writing the matroid rank function for M as a linear threshold function of disjunctions.

Proof of Theorem 12. We first claim $\mathbb{F}_2$-sketching complexity of the rank function of any rank 2 matroid M is essentially the same as the complexity of the corresponding Boolean function that takes value 1 if $rank_M(x) = 2$ and takes value 0 otherwise. Indeed, let the function above be denoted as f_M. Without loss of generality, we can assume that all singletons are independent sets in M as otherwise the rank function of M does not depend on the corresponding input. Hence $rank_M(x) = 0$ if and only if $x = 0^n$. Thus $R^{lin}_{\delta}(rank_M) = R^{lin}_{\delta}(f_M) + O(\log 1/\delta)$ as by Fact 15 we can use $O(\log 1/\delta)$-bit sketch to check whether $x = 0^n$ first and then evaluate $rank_M$ using f_M. Recall from Fact 13 that matroids of rank 2 can be represented as edges in a complete graph with edges corresponding to some disjoint union of cliques removed.

Let $S_1, \ldots, S_t$ be the collection of vertex sets of disjoint cliques defining a rank 2 matroid M in Fact 13. Without loss of generality, we can assume that $|\cup_{i=1}^{t} S_i| = n$ by adding singletons. Then:

$$f_M(x) = \mathsf{HAM}_{\geq 2}\left(\bigvee_{j\in S_1} x_j, \bigvee_{j\in S_2} x_j, \ldots, \bigvee_{j\in S_t} x_j\right),$$

where $\mathsf{HAM}_{\geq 2}(z_1, \ldots, z_t) = 1$ if and only if $\sum_{i=1}^{t} z_i \geq 2$ is the threshold Hamming weight function. By Theorem 14, the sketch complexity of $f_M(x)$ is $O(1)$, since the Hamming weight threshold is $d = 2$. ◀

Since the independent bases of a graphic matroid $M(G)$ are the spanning forests of G, the matroid rank function of a graphic matroid of rank r can be expressed as

$$f_M(x) = \mathsf{HAM}_{\geq r}\left(\bigvee_{j\in S_1} x_j, \bigvee_{j\in S_2} x_j, \ldots, \bigvee_{j\in S_t} x_j\right),$$

where each S_i is a separate spanning forest. Therefore, Theorem 14 yields a $O(r^2 \log r)$ space linear sketch for graphic matroids of rank r.

▶ **Theorem 16.** *For every graphic matroid M of rank r, it holds that $R^{lin}_{\frac{1}{3}}(rank_M) = O(r^2 \log r)$.*

We use the remainder of the Section 3.1 to prove Theorem 14, while resolving an open question by Montanaro and Osborne [45].

3.1.1 Linear Threshold Functions

We first define linear threshold functions (LTFs) and (θ, m)-LTFs.

▶ **Definition 17.** *A function $f : \{0,1\}^n \to \{0,1\}$ is a* linear threshold function *(LTF) if there exist constants $\theta, w_1, w_2, \ldots, w_n$ such that $f(x) = \mathsf{sgn}\left(-\theta + \sum_{i=1}^n w_i x_i\right)$, where $\mathsf{sgn}(y) = 0$ for $y < 0$ and $\mathsf{sgn}(y) = 1$ for $y \geq 0$ is the Heaviside step function.*

▶ **Definition 18.** *A monotone linear threshold function $f : \{0,1\}^n \to \{0,1\}$ is a (θ, m)-*LTF *if $m \leq \min_{x\in\{0,1\}^n} |-\theta + \sum_{i=1}^n w_i x_i|$, where θ is referred to as the* threshold *and m as the margin of the LTF.*

Although (θ, m)-LTFs have previously been shown to have randomized linear sketch complexity $O\left(\frac{\theta}{m} \log n\right)$ [42], Montanaro and Osborne asked whether any (θ, m)-LTF can be represented in the simultaneous model with $O\left(\frac{\theta}{m} \log \frac{\theta}{m}\right)$ communication.

▶ **Question 19** ([45]). *Let $g(x, y) = f(x \oplus y)$, where f is a (θ, m)-LTF. Does there exist a protocol for g in the simultaneous model with communication complexity $O\left(\frac{\theta}{m} \log \frac{\theta}{m}\right)$?*

Note that the difference between $\log n$ and $\log \frac{\theta}{m}$ is crucial for obtaining constant randomized linear sketch complexity for functions for matroid rank 2. We answer Question 19 in the affirmitive and show the stronger result that (θ, m)-LTFs admit a randomized linear sketch of size $O\left(\frac{\theta}{m} \log \frac{\theta}{m}\right)$. We first show that we can completely ignore all variables whose weights are significantly smaller than $2m$ in evaluating a (θ, m)-LTF.

▶ **Lemma 20.** *Let $f(x) = \mathsf{sgn}\left(-\theta + \sum_{i=1}^n w_i x_i\right)$ be a (θ, m)-LTF. For $1 \leq i \leq n$, let $w'_i = w_i$ if $w_i \geq 2m$ and $w'_i = 0$ otherwise. Then $f(x) = \mathsf{sgn}\left(-\theta + \sum_{i=1}^n w'_i x_i\right)$.*

As noted, Lemma 20 implies that we can ignore not only variables with zero weights, but all variables whose weights are less than $2m$. We now bound the support of the set $\{x \mid f(x) = 0\}$, where f is a (θ, m)-LTF, and apply Fact 15.

▶ **Lemma 21.** *For any (θ, m)-LTF, there exists a randomized linear sketch of size $O\left(\frac{\theta}{m} \log n\right)$.*

Proof. Let $f(x) = \mathsf{sgn}\left(-\theta + \sum_{i=1}^n w_i x_i\right)$ be a (θ, m)-LTF. By Lemma 20, the output of f remains the same even if we only consider the variables S with weight at least $2m$. On the other hand, if $f(x) = 0$, then at most $\frac{\theta}{2m}$ variables in S can have value 1. Equivalently, at most $\frac{\theta}{2m}$ indices i can have $x_i = 1$ if $f(x) = 0$. Thus, the number of $x \in \{0,1\}^n$ with $f(x) = 0$ is at most $\sum_{0\leq i\leq \theta/2m} \binom{n}{i} \leq (n+1)^{\lceil \theta/2m \rceil}$. Applying Fact 15, there exists a randomized linear sketch for f, of size $O\left(\frac{\theta}{m} \log n\right)$. ◀

In order to fully prove Question 19 and obtain a dependence on $\log \frac{\theta}{m}$ rather than $\log n$, we use the following two observations. First, we show in Lemma 22 that the weights of a (θ, m)-LTF can be rounded to a set that contains $O\left(\frac{\theta}{m}\right)$ elements. Second, we show in Theorem 25 that we can then use hashing to reduce the number of variables down to $\mathsf{poly}\left(\frac{\theta}{m}\right)$ before applying Lemma 21.

▶ **Lemma 22.** *Let $f(x) = \mathsf{sgn}\left(-\theta + \sum_{i=1}^{n} w_i x_i\right)$ be a (θ, m)-LTF. Then there exists a set W with $|W| = O\left(\frac{\theta}{m} \log \frac{\theta}{m}\right)$, and a margin $m' = \Theta(m)$ such that $f(x) = \mathsf{sgn}\left(-\theta + \sum_{i=1}^{n} w'_i x_i\right)$, where each $w'_i \in W$ and f is a (θ, m')-LTF.*

The following result is also useful for our construction of a sketch for a (θ, m)-LTF.

▶ **Lemma 23** ([33])**.** *There is a randomized linear sketch with size $O(1)$ for the function*

$$\mathsf{HAM}_{n,d|2d}(x) = \begin{cases} 1, & \text{if } ||x||_0 \leq d \\ 0, & \text{if } ||x||_0 \geq 2d \end{cases}$$

on instances $\{x | x \in \{0,1\}^n$ and $||x||_0 \leq d$ or $||x||_0 \geq 2d\}$.

▶ **Fact 24.** *If $h : [n] \to [M]$ is a random hash function and $S \subseteq [n]$, then the probability that there exist $x, y \in S$ with $h(x) = h(y)$ is at most $\frac{|S|^2}{M}$.*

▶ **Theorem 25.** *Any (θ, m)-LTF admits a randomized linear sketch of size $O\left(\frac{\theta}{m} \log \frac{\theta}{m}\right)$.*

Proof. Let $f(x) = \mathsf{sgn}\left(-\theta + \sum_{i=1}^{n} w_i x_i\right)$ be a (θ, m)-LTF. By Lemma 22, we can assume that $w_i \in W = \{2m(1+\epsilon)^i\}_{i=0}^{t}$ so that the new margin $m' = \frac{4}{5}m$ and $t = \left\lceil \log_{1+\epsilon} \frac{\theta}{m} \right\rceil$ for $\epsilon = \frac{\theta}{10m}$. Recall from Lemma 21, $f(x) = 0$ only if $x_i = 1$ for at most $\frac{\theta}{2m}$ indices i of x. From Lemma 23, we can detect the instances where at least $\frac{\theta}{2m}$ indices i of x satisfy $x_i = 1$.

On the other hand, if less than $\frac{\theta}{2m}$ indices i of x satisfy $x_i = 1$, we can identify these indices and corresponding weights via hashing. Let $h : [n] \to [M]$, where $M = 5\left(\frac{\theta}{m}\right)^2$, and S be a set of indices of x, of size at most $\frac{\theta}{m}$. Then by Fact 24, the probability of a collision in h under elements of S is at most $\frac{1}{5}$. We partition $[n]$ into sets $S_{w,j}$ where $w \in W$ and $j \in [M]$ so that $S_{w,j} = \{i | h(i) = j \wedge w_i = w\}$. Therefore with probability at least $\frac{4}{5}$, there are no collisions in h under elements of S and $|S_{w,j}| \leq 1$ for all $w \in W$ and $j \in [M]$.

Let $y_{w,j} = \sum_{i \in S_{w,j}} x_i$ and note that if there are no collisions in h under elements of S, then

$$\sum_{i=1}^{n} w_i x_i = \sum_{(j,w) \in [M] \times W} w \left(\sum_{i \in S_{w,j}} x_i \right) = \sum_{(j,w) \in [M] \times W} w \cdot y_{w,j}.$$

Thus, $f(x)$ is equivalent to the function $g(y) = \mathsf{sgn}\left(-\theta + \sum_{w,j} w \cdot y_{w,j}\right)$. Since $|W| = O\left(\frac{\theta}{m} \log \frac{\theta}{m}\right)$, $M = 5\left(\frac{\theta}{m}\right)^2$ and $m' = \frac{4}{5}m$ is the margin for $g(y)$, then $g(y)$ depends on $O\left(\left(\frac{\theta}{m}\right)^3 \log \frac{\theta}{m}\right)$ variables $y_{w,j}$. By Lemma 21, there exists a randomized sketch for $g(y)$ of size $O\left(\frac{\theta}{m} \log \frac{\theta}{m}\right)$. ◀

We can also show that Theorem 25 is tight by recalling the function

$$\mathsf{HAM}_{\leq d}(x) = \begin{cases} 0, & \text{if } \sum_{i=1}^{n} x_i \leq d + \frac{1}{2} \\ 1, & \text{otherwise.} \end{cases}$$

Since this function is a $\left(d+\frac{1}{2},\frac{1}{2}\right)$-LTF, it can be represented by a randomized linear sketch of size $O(d\log d)$. On the other hand, Dasgupta, Kumar and Sivakumar [19] notes that the one-way complexity of small set disjointness for two vectors x and y of weight d, which reduces to the function $\mathsf{HAM}_{\leq d}(x\oplus y)$, is $\Omega(d\log d)$. Thus, $\mathsf{HAM}_{\leq d}(x\oplus y)$ also requires a sketch of size $\Omega(d\log d)$.

3.1.2 Linear Threshold of Disjunctions

In this section, we describe a randomized linear sketch for functions that can be represented as 2-depth circuits where the top gate is a monotone linear threshold function with threshold θ and margin m, and the bottom gates are OR functions. Formally, if $g_S(x)=\bigvee_{i\in S}x_i$, q is a linear threshold function, and $w_S\geq 0$, then $f(x)=q(\ldots,g_S(x),\ldots)=\mathsf{sgn}\left(-\theta+\sum_{S\in 2^{[n]}}w_S\cdot g_S(x)\right)$.

▶ **Lemma 26.** *Let $f(x)=\mathsf{sgn}\left(-\theta+\sum_{i=1}^{n}w_ix_i\right)$ be a (θ,m)-LTF where $w_i\in W$ for some set W. Let $h:[n]\to[M]$ be a random hash function where $M=\frac{50\theta^2}{m^2}$ and*

$$f_h(x)=\mathsf{sgn}\left(-\theta+\sum_{(j,w)\in[M]\times W}w\left(\bigvee_{\substack{i:h(i)=j\\ w_i=w}}x_i\right)\right).$$

Then for all x, $\mathbf{Pr}\left[f_h(x)\neq f(x)\right]\leq\frac{1}{50}$.

▶ **Theorem 27.** *Let $g_S(x)=\bigvee_{i\in S}x_i$ with $w_S\geq 0$, q be a (θ,m)-LTF, and*

$$f(x)=q(\ldots,g_S(x),\ldots)=\mathsf{sgn}\left(-\theta+\sum_{S\in 2^{[n]}}w_S\cdot g_S(x)\right).$$

Then there is a randomized linear sketch for f of size $O\left(\left(\frac{\theta}{m}\right)^4\log^2\frac{\theta}{m}\right)$, where m is the margin of q.

Proof. We first apply Lemma 20 and Lemma 22 to q so that weights w_i can be rounded to elements of a set W with $|W|=O\left(\frac{\theta}{m}\log\frac{\theta}{m}\right)$. For each $w_i\in W$, it again suffices to detect whether $\Theta(\frac{\theta}{m})$ disjunctions are nonzero. Hence to hash $O\left(\left(\frac{\theta}{m}\right)^2\log\frac{\theta}{m}\right)$ disjunctions, it suffices to use a hash function with $M=O\left(\left(\frac{\theta}{m}\right)^4\log^2\frac{\theta}{m}\right)$ buckets. By Lemma 26, our resulting randomized linear sketch has size $O\left(\left(\frac{\theta}{m}\right)^4\log^2\frac{\theta}{m}\right)$. ◀

Proof of Theorem 14. Recall that $\mathsf{HAM}_{\leq d}(x)$ is a $\left(d+\frac{1}{2},\frac{1}{2}\right)$-LTF. Furthermore, the set of weights W for $\mathsf{HAM}_{\leq d}(x)$ consists of a single element $\{1\}$, since the coefficient of each disjunction is one. Since $M=O(d^2\log d)$, we can construct a randomized linear sketch with size $O(d^2\log d)$ by Lemma 26. ◀

We note that our approach can be easily generalized to the case where the disjunction include the negations of some variables as well.

3.2 Communication Complexity of Lipschitz Submodular Functions

We discuss the communication complexity of Lipschitz submodular functions in this section. We first show in Section 3.2.1 that there exists an $\Omega(n)$-Lipschitz submodular function f that requires a randomized linear sketch of size $\Omega(n)$. We then show in Section 3.2.2 that in the one-way communication complexity model for XOR functions, there exists an $\Omega(n)$-Lipschitz submodular function f that has communication complexity $\Omega(n)$.

3.2.1 Approximate $\mathbb{F}_2$-Sketching of Lipschitz Submodular Functions

▶ **Theorem 28.** *There exist constants $c_1, c_2, \epsilon \geq 0$ and a monotone non-negative $(\frac{c_1}{n})$-Lipschitz submodular function f (a scaling of a matroid rank function) such that $\bar{R}^{lin}_{\epsilon}(f) \geq c_2 n$.*

Proof. Our proof uses a construction of a large family of matroid rank functions given in [10], Theorem 8. The construction uses the following notion of lossless bipartite expanders:

▶ **Definition 29** (Lossless bipartite expander). *Let $G = (U \cup V, E)$ be a bipartite graph. For $J \subseteq U$ let $\Gamma(J) = \{v | \exists u \in U \colon \{u, v\} \in E\}$. Graph G is a (D, L, ϵ)-lossless expander if:*

$$|\Gamma(\{u\})| = D \quad \forall u \in U$$
$$|\Gamma(J)| \geq (1-\epsilon)D|J| \quad \forall J \subseteq U, |J| \leq L.$$

Here we need different parameters than in [10] so we restate their theorem as follows:

▶ **Theorem 30** ([10]). *Let $(U \cup V, E)$ be a (D, L, ϵ)-lossless expander with $|U| = k$ and $|V| = n$ and let $b = 8 \log k$. If $D \geq b$, $L = 4D/b - 2$ and $\epsilon = \frac{b}{4D}$ then there exists a family of sets $\mathcal{A} \subseteq 2^{[n]}$ and a family of matroids $\{M_{\mathcal{B}} \colon \mathcal{B} \subseteq \mathcal{A}\}$ with the following properties:*

- *$|\mathcal{A}| = k$ and for every $A \in \mathcal{A}$ it holds that $|A| = D$.*
- *For every $\mathcal{B} \subseteq \mathcal{A}$ and every $A \in \mathcal{A}$, we have:*

$$rank_{M_{\mathcal{B}}}(A) = \begin{cases} b & \text{if } A \in \mathcal{B} \\ D & \text{if } A \in \mathcal{A} \setminus \mathcal{B} \end{cases}$$

We use the following construction of lossless expanders from [54], see also [10].

▶ **Theorem 31** ([54]). *Let $k \geq 2$ and $\epsilon \geq 0$. For any $L \leq k$, let $D \geq 2 \log k/\epsilon$ and $n \geq 6DL/\epsilon$. Then a (D, L, ϵ)-lossless expander exists.*

In the above theorem we can set parameters as follows:

$$D = \frac{n}{3 \cdot 2^7}, \quad L = 2^3, \quad \epsilon = 2^{-3}, \quad k = 2^{n/3 \cdot 2^{11}}, \quad b = \frac{n}{3 \cdot 2^8}.$$

Note that under this choice of parameters we have $6DL/\epsilon = n$ and $\frac{2 \log k}{\epsilon} = D$ and hence a (D, L, ϵ)-lossless expander with parameters set above exists.

Now consider the family of matroids $\mathcal{M}$ given by Theorem 30 using the expander construction above. The rest of the proof uses the probabilistic method. We will show non-constructively that there exists a matroid in this family whose rank function does not admit a sketch of dimension $d = o(n)$. Let $\mathcal{D} = U(\mathcal{A})$ be the uniform distribution over $\mathcal{A}$. By Yao's principle it suffices to show that there exists a matroid rank function for which any deterministic sketch fails with a constant probability over this distribution. In the proof below we first show that any fixed deterministic sketch succeeds on a randomly chosen matroid from $\mathcal{M}$ with only a very tiny probability, probability $2^{2^{-\Omega(n)}}$, and then take a union bound over all 2^{dn} sketches of dimension at most d.

Indeed, fix any deterministic sketch $\mathcal{S}$ of dimension $d = n/2^{11}$. Let $\{b_1, \ldots, b_{2^d}\}$ be the set of all possible binary vectors of length d corresponding to the possible values of the sketch, so that each $b_i \in \{0,1\}^d$.

Let $S_{b_i} = \{A \in \mathcal{A} : \mathcal{S}(A) = b_i\}$. Let $t = \frac{1}{4}2^{n/2^{11}}$ and $G = \{b_i \in \{0,1\}^d || S_{b_i}| \geq t\}$. The following proposition follows by a simple calculation.

▶ **Proposition 32.** *If $t = \frac{1}{4}2^{n/2^{11}}$ then $\frac{1}{k}\sum_{b_i \in G} |S_{b_i}| \geq \frac{3}{4}$.*

Proof. We have:

$$\frac{1}{k}\sum_{b_i \in G} |S_{b_i}| \geq 1 - \frac{1}{k} \sum_{b_i :\, |S_{b_i}| < \frac{k}{4 \cdot 2^d}} |S_{b_i}| \geq 1 - \frac{1}{k} \cdot \frac{k}{4 \cdot 2^d} \cdot 2^d \geq \frac{3}{4}.$$ ◀

Let $S^1_{b_i} = \{A \in S_{b_i} : rank_{M_\mathcal{B}}(A) = b\}$ and $S^2_{b_i} = \{A \in S_{b_i} : rank_{M_\mathcal{B}}(A) = D\}$. We require the following lemma.

▶ **Lemma 33.** *Let $t = \frac{1}{4}2^{n/2^{11}}$ and $d = n/2^{11}$. There exists a matroid $M_\mathcal{B} \in \mathcal{M}$ such that for all deterministic sketches $\mathcal{S}$ of dimension d and all $b_i \in G$, $\min(|S^1_{b_i}|, |S^2_{b_i}|) \geq \frac{1}{4}|S_{b_i}|$.*

Fix the set $\mathcal{B}$ constructed in Lemma 33 and consider the function $rank_{M_\mathcal{B}}$. Consider distribution $\mathcal{D}$ over the inputs. The probability that any deterministic sketch over this distribution makes error at least $D - b$ is at least:

$$\frac{1}{k}\sum_{b_i \in \{0,1\}^n} \min(|S^1_{b_i}|, |S^2_{b_i}|) \geq \frac{1}{k}\sum_{b_i \in G} \min(|S^1_{b_i}|, |S^2_{b_i}|) \geq \frac{1}{k}\sum_{b_i \in G} \frac{1}{4}|S_{b_i}|,$$

where the last inequality holds by Lemma 33. Thus by Proposition 32, the probability is at least $\frac{3}{4} \times \frac{1}{4} \geq \frac{1}{6}$.

Finally, the construction of [10] ensures that the function $rank_{M_\mathcal{B}}$ takes integer values between 0 and D. Using this and the fact that matroid rank functions are 1-Lipschitz, we can normalize it by dividing all values by D and ensure that the resulting function is $O(1/n)$-Lipschitz and takes values in $[0,1]$, while the sketch makes error at least $(D-b)/D = \frac{1}{2}$. ◀

3.2.2 One-Way Communication of Lipschitz Submodular Functions

In this section, we strengthen the lower bound shown above, extending it to the corresponding one-way communication problem. We use the same notation as in the previous section.

▶ **Theorem 34.** *There exists a constant $c_1 > 0$ and a $\frac{c_1}{n}$-Lipschitz submodular function such that $R^{\rightarrow}_{1/3} = \Omega(n)$.*

By restricting the n-dimensional elements to r coordinates and observing that the construction outputs matroids of rank b or D that are separated by a constant gap, we obtain the following result using the same proof:

▶ **Corollary 35.** *There exists $c = \Omega(1)$ such that a c-approximation of matroid rank functions has randomized one-way communication complexity $R^{\rightarrow}_{1/3} = \Omega(r)$ where r is the rank of the underlying matroid.*

References

1 Dragan M Acketa. On the enumeration of matroids of rank-2. *Zbornik radova Prirodnomatematickog fakulteta–Univerzitet u Novom Sadu*, 8:83–90, 1978.

2 Yuqing Ai, Wei Hu, Yi Li, and David P. Woodruff. New Characterizations in Turnstile Streams with Applications. In *31st Conference on Computational Complexity, CCC*, pages 20:1–20:22, 2016.

3 Noga Alon, Yossi Matias, and Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

4 Sepehr Assadi, Sanjeev Khanna, and Yang Li. Tight bounds for single-pass streaming complexity of the set cover problem. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 698–711, 2016.

5 László Babai, Anna Gál, Peter G. Kimmel, and Satyanarayana V. Lokam. Communication Complexity of Simultaneous Messages. *SIAM J. Comput.*, 33(1):137–166, 2003.

6 László Babai and Peter G. Kimmel. Randomized Simultaneous Messages: Solution of a Problem of Yao in Communication Complexity. In *Proceedings of the Twelfth Annual IEEE Conference on Computational Complexity*, pages 239–246, 1997.

7 Ashwinkumar Badanidiyuru, Shahar Dobzinski, Hu Fu, Robert Kleinberg, Noam Nisan, and Tim Roughgarden. Sketching valuation functions. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1025–1035, 2012.

8 Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming submodular maximization: massive data summarization on the fly. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 671–680, 2014.

9 Maria-Florina Balcan, Florin Constantin, Satoru Iwata, and Lei Wang. Learning Valuation Functions. In *COLT 2012 - The 25th Annual Conference on Learning Theory, June 25-27, 2012, Edinburgh, Scotland*, pages 4.1–4.24, 2012.

10 Maria-Florina Balcan and Nicholas J. A. Harvey. Learning submodular functions. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC*, pages 793–802, 2011.

11 MohammadHossein Bateni, Hossein Esfandiari, and Vahab S. Mirrokni. Almost Optimal Streaming Algorithms for Coverage Problems. In *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA*, pages 13–23, 2017.

12 Eric Blais and Abhinav Bommireddi. Testing Submodularity and Other Properties of Valuation Functions. In *8th Innovations in Theoretical Computer Science Conference, ITCS*, pages 33:1–33:17, 2017.

13 Eric Blais, Krzysztof Onak, Rocco Servedio, and Grigory Yaroslavtsev. Concise representations of discrete submodular functions, 2013.

14 Jehoshua Bruck and Roman Smolensky. Polynomial Threshold Functions, AC^0 Functions, and Spectral Norms. *SIAM J. Comput.*, 21(1):33–42, 1992.

15 Amit Chakrabarti and Anthony Wirth. Incidence Geometries and the Pass Complexity of Semi-Streaming Set Cover. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016*, pages 1365–1373, 2016.

16 Deeparnab Chakrabarty and Zhiyi Huang. Testing Coverage Functions. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP, Proceedings, Part I*, pages 170–181, 2012.

17 Chandra Chekuri, Shalmoli Gupta, and Kent Quanrud. Streaming Algorithms for Submodular Function Maximization. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP, Proceedings, Part I*, pages 318–330, 2015.

18 Mahdi Cheraghchi, Adam R. Klivans, Pravesh Kothari, and Homin K. Lee. Submodular functions are noise stable. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1586–1592, 2012.

19 Anirban Dasgupta, Ravi Kumar, and D. Sivakumar. Sparse and Lopsided Set Disjointness via Information Theory. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 15th International Workshop, APPROX, and 16th International Workshop, RANDOM. Proceedings*, pages 517–528, 2012.

20 Erik D. Demaine, Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian. On Streaming and Communication Complexity of the Set Cover Problem. In *Distributed Computing - 28th International Symposium, DISC. Proceedings*, pages 484–498, 2014.

21 Yuval Emek and Adi Rosén. Semi-Streaming Set Cover. *ACM Trans. Algorithms*, 13(1):6:1–6:22, 2016.

22 Vitaly Feldman and Pravesh Kothari. Learning Coverage Functions and Private Release of Marginals. In *Proceedings of The 27th Conference on Learning Theory, COLT*, pages 679–702, 2014.

23 Vitaly Feldman, Pravesh Kothari, and Jan Vondrák. Representation, Approximation and Learning of Submodular Functions Using Low-rank Decision Trees. In *COLT 2013 - The 26th Annual Conference on Learning Theory*, pages 711–740, 2013.

24 Vitaly Feldman and Jan Vondrák. Tight Bounds on Low-Degree Spectral Concentration of Submodular and XOS Functions. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS*, pages 923–942, 2015.

25 Vitaly Feldman and Jan Vondrák. Optimal Bounds on Approximation of Submodular and XOS Functions by Juntas. *SIAM J. Comput.*, 45(3):1129–1170, 2016.

26 Gereon Frahling, Piotr Indyk, and Christian Sohler. Sampling in Dynamic Data Streams and Applications. *Int. J. Comput. Geometry Appl.*, 18(1/2):3–28, 2008.

27 Michel X. Goemans, Nicholas J. A. Harvey, Satoru Iwata, and Vahab S. Mirrokni. Approximating submodular functions everywhere. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 535–544, 2009.

28 Vince Grolmusz. On the Power of Circuits with Gates of Low L_1 Norms. *Theor. Comput. Sci.*, 188(1-2):117–128, 1997.

29 Anupam Gupta, Moritz Hardt, Aaron Roth, and Jonathan Ullman. Privately Releasing Conjunctions and the Statistical Query Barrier. *SIAM J. Comput.*, 42(4):1494–1520, 2013.

30 Sariel Har-Peled, Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian. Towards Tight Bounds for the Streaming Set Cover Problem. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS*, pages 371–383, 2016.

31 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of Protocols for XOR Functions. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS*, pages 282–288, 2016.

32 Kaave Hosseini, Shachar Lovett, and Grigory Yaroslavtsev. Optimality of Linear Sketching under Modular Updates. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:169, 2018. URL: `https://eccc.weizmann.ac.il/report/2018/169`.

33 Wei Huang, Yaoyun Shi, Shengyu Zhang, and Yufan Zhu. The communication complexity of the Hamming distance problem. *Inf. Process. Lett.*, 99(4):149–153, 2006.

34 William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability*, pages 189–206, 1984.

35 Sampath Kannan, Elchanan Mossel, Swagato Sanyal, and Grigory Yaroslavtsev. Linear Sketching over F_2. In *33rd Computational Complexity Conference, CCC*, pages 8:1–8:37, 2018.

36 Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. Single Pass Spectral Sparsification in Dynamic Streams. *SIAM J. Comput.*, 46(1):456–477, 2017.

37 Michael Kapralov, Jelani Nelson, Jakub Pachocki, Zhengyu Wang, David P. Woodruff, and Mobin Yahyazadeh. Optimal Lower Bounds for Universal Relation, and for Samplers and Finding Duplicates in Streams. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 475–486, 2017.

38 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

39 Troy Lee and Shengyu Zhang. Composition Theorems in Communication Complexity. In *Automata, Languages and Programming, 37th International Colloquium, ICALP, Proceedings, Part I*, pages 475–489, 2010.

40 Ming Lam Leung, Yang Li, and Shengyu Zhang. Tight bounds on the randomized communication complexity of symmetric XOR functions in one-way and SMP models. *CoRR*, abs/1101.4555, 2011. `arXiv:1101.4555`.

41 Yi Li, Huy L. Nguyen, and David P. Woodruff. Turnstile streaming algorithms might as well be linear sketches. In *Symposium on Theory of Computing, STOC*, pages 174–183, 2014.

42 Yang Liu and Shengyu Zhang. Quantum and randomized communication complexity of XOR functions in the SMP model. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:10, 2013.

43 Shachar Lovett. Recent Advances on the Log-Rank Conjecture in Communication Complexity. *Bulletin of the EATCS*, 112, 2014.

44 Andrew McGregor. Graph stream algorithms: a survey. *SIGMOD Record*, 43(1):9–20, 2014.

45 Ashley Montanaro and Tobias Osborne. On the communication complexity of XOR functions. *CoRR*, abs/0909.3392, 2009. `arXiv:0909.3392`.

46 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

47 Sofya Raskhodnikova and Grigory Yaroslavtsev. Learning pseudo-Boolean k-DNF and submodular functions. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1356–1368, 2013.

48 Barna Saha and Lise Getoor. On Maximum Coverage in the Streaming Model & Application to Multi-topic Blog-Watch. In *Proceedings of the SIAM International Conference on Data Mining, SDM*, pages 697–708, 2009.

49 C. Seshadhri and Jan Vondrák. Is Submodularity Testable? *Algorithmica*, 69(1):1–25, 2014.

50 Yaoyun Shi and Zhiqiang Zhang. Communication complexities of symmetric XOR functions. *Quantum Inf. Comput*, pages 0808–1762, 2008.

51 Xiaoming Sun and Chengu Wang. Randomized Communication Complexity for Linear Algebra Problems over Finite Fields. In *29th International Symposium on Theoretical Aspects of Computer Science, STACS*, pages 477–488, 2012.

52 Justin Thaler. Semi-Streaming Algorithms for Annotated Graph Streams. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 59:1–59:14, 2016.

53 Hing Yin Tsang, Chung Hoi Wong, Ning Xie, and Shengyu Zhang. Fourier Sparsity, Spectral Norm, and the Log-Rank Conjecture. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 658–667, 2013.

54 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

55 Jan Vondrák. A note on concentration of submodular functions. *CoRR*, abs/1005.2791, 2010. `arXiv:1005.2791`.

56 David P. Woodruff. Sketching as a Tool for Numerical Linear Algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.

57 Grigory Yaroslavtsev and Samson Zhou. Approximate F_2-Sketching of Valuation Functions. *CoRR*, abs/1907.00524, 2019. `arXiv:1907.00524`.

58 Zhiqiang Zhang and Yaoyun Shi. On the parity complexity measures of Boolean functions. *Theor. Comput. Sci.*, 411(26-28):2612–2618, 2010.

A Missing Proofs

Proof of Proposition 10. Indeed, let A_d be a d-dimensional subspace such that $\sum_{S\in A_d}\hat{f}^2(S) \geq \|f\|_2^2 - \epsilon$ and consider the function $g(x) = \sum_{S\in A_d}\hat{f}(S)\chi_S(x)$. Note that in order to compute all values $\chi_S(x)$ for $S \in A_d$ it suffices to evaluate d parities corresponding to sets $S_1, \ldots, S_d$ forming a basis in A_d. Values of all other parities can be computed as linear combinations. Let $\Delta(x) = f(x) - g(x)$. Then the desired guarantee follows from the following calculation:

$$\mathop{\mathbb{E}}_{x\sim U(\{0,1\}^n)}[\Delta(x)^2] = \mathop{\mathbb{E}}_{S\sim U(\{0,1\}^n)}[\hat{\Delta}(S)^2] = \sum_{S\in\{0,1\}^n}(\hat{f}(S) - \hat{g}(S))^2 = \sum_{S\notin A_d}\hat{f}(S)^2 \leq \epsilon,$$

where the first equality holds from Parseval's identity. ◀

Proof of Lemma 20. We show the stronger result that for any j such that $w_j < 2m$, then $f(x) = f(x \oplus e_j)$, where e_j is the elementary unit vector with one in the jþ position, and zeros elsewhere. This implies the lemma since it shows that any variable whose weight is less than $2m$ does not affect the output of the function or the margin of the function and thus might as well have weight zero.

Suppose, by way of contradiction, that $f(x) \neq f(x \oplus e_j)$ and without loss of generality, $f(x) = 0$ with $x_j = 0$. Since f is a linear threshold function and $f(x) = 0$, then $-\theta + \sum_{i=1}^n w_i x_i < 0$. Moreover, f is a (θ, m)-LTF, so $-\theta + \sum_{i=1}^n w_i x_i < -m$. Because $w_j < 2m$, $-\theta + w_j + \sum_{i=1}^n w_i x_i < -\theta + 2m + \sum_{i=1}^n w_i x_i < m$. But because m is the margin of the function, if $-\theta + w_j + \sum_{i=1}^n w_i x_i < m$, then it must hold that $-\theta + w_j + \sum_{i=1}^n w_i x_i < -m$. Therefore, $f(x \oplus e_j) = 0$, so x_j does not affect the output of the function or the margin of the function. ◀

Proof of Lemma 22. Observe that for any $w_i \geq 2\theta$, if $x_i = 1$, then $f(x) = 1$. Thus, if $f(x) = 1$, it suffices to consider $2m \leq w_i \leq 2\theta$.

Let $W = \{2m(1+\epsilon)^i\}_{i=0}^t$ for $t = \left\lceil \log_{1+\epsilon}\left(\frac{\theta}{m}\right)\right\rceil$, where ϵ is some fixed constant that we set at a later time. For each i, let w_i' be the largest element in W that does not exceed w_i. Thus, $w_i' \leq w_i < (1+\epsilon)w_i'$. Observe that since $w_i' \leq w_i$ and f is a (θ, m)-LTF, then $f(x) = 0$ implies $-m > -\theta + \sum_{i=1}^n w_i x_i \geq -\theta + \sum_{i=1}^n w_i x_i$, so that $\mathsf{sgn}\left(-\theta + \sum_{i=1}^n w_i' x_i\right) = 0 = f(x)$ and a margin of m remains.

On the other hand, if $f(x) = 1$, then $\sum_{i=1}^n w_i x_i > \theta + m$ as f is a (θ, m)-LTF. Since $w_i' \leq w_i < (1+\epsilon)w_i'$, then $\sum_{i=1}^n w_i' x_i > \frac{\theta+m}{1+\epsilon} > (1-\epsilon)(\theta+m)$. Observe that $\theta \geq m$ and hence, $\sum_{i=1}^n w_i' x_i > \theta - \epsilon\theta + m - \epsilon m \geq \theta + m - 2\epsilon\theta$. Setting $\epsilon = \frac{\theta}{10m}$ shows that $\mathsf{sgn}\left(-\theta + \sum_{i=1}^n w_i' x_i\right) = 1 = f(x)$ and a margin of $m' = \frac{4}{5}m$ remains. ◀

Proof of Lemma 26. As by Lemma 22, we can assume without loss of generality that $w_i \geq 2m$ and $w \geq 2m$. Let $S = \{i | x_i = 1\}$ so that if there are no collisions under h in S, then

$$\sum_{(j,w)\in[M]\times W} w\left(\bigvee_{\substack{i:h(i)=j\\ w_i=w}} x_i\right) = \sum_i w_i x_i.$$

If $f(x) = 0$, then $|S| \leq \frac{\theta}{2m}$ so that the probability there are collisions under h in S is at most $\frac{1}{200}$ by Fact 24. Thus if $f(x) = 0$, then $f_h(x) = 0$ with probability at least $1 - \frac{1}{200}$.

If $f(x) = 1$, then either $|S| < \frac{\theta}{m}$ or $|S| \geq \frac{\theta}{m}$. If $|S| < \frac{\theta}{m}$, then the probability there are collisions under h in S is at most $\frac{1}{50}$ by Fact 24, so then $f_h(x) = 1$ with probability at least $1 - \frac{1}{50}$. If $|S| \geq \frac{\theta}{m}$, with probability at least $1 - \frac{1}{50}$, there exist $\frac{\theta}{m}$ values j such that there exists $x_i = 1$ and $h(i) = j$. Therefore, we set $f_h(x) = 1$ whenever at least $\frac{\theta}{m}$ buckets of h are non-empty. In all cases, $f_h(x) = f(x)$ with probability at least $1 - \frac{1}{50}$. ◀

Proof of Lemma 33. The proof uses the probabilistic method to show the existence of $\mathcal{B}$ with desired properties. Consider drawing a random matroid from the family $\mathcal{M}$, i.e. pick $\mathcal{B}$ to be a uniformly random subset of $\mathcal{A}$ and consider $M_{\mathcal{B}}$. Fix any deterministic sketch $\mathcal{S}$ and any $b_i \in G$. Since $|S_{b_i}| \geq t$, by the Chernoff bound, it holds that:

$$\Pr_{\mathcal{B} \subseteq \mathcal{A}} \left[\left|S^1_{b_i}\right| > \left(\frac{1}{2} + \delta\right) |S_{b_i}| \right] \leq e^{-c\delta^2 |S_{b_i}|} \leq e^{-c\delta^2 t}.$$

Setting $\delta = 1/4$, we have that the above probability is at most e^{-Ct} for some constant $C > 0$. Applying the argument above to both $S^1_{b_i}$ and $S^2_{b_i}$, we have that:

$$\Pr_{\mathcal{B} \subseteq \mathcal{A}} \left[\min(\left|S^1_{b_i}\right|, \left|S^2_{b_i}\right|) < \frac{1}{4} |S_{b_i}| \right] \leq 2e^{-Ct}.$$

Let $\mathcal{E}$ denote the event that $\min(\left|S^1_{b_i}\right|, \left|S^2_{b_i}\right|) \geq \frac{1}{4}|S_{b_i}|$.

Note that the total number of deterministic sketches of dimension d is at most 2^{dn}, since each sketch is specified by a collection of d linear functions over $\mathbb{F}_2^n$. Also note that for each sketch $|G| \leq 2^d$. Taking a union bound over all sketches and all sets G by the choice of t and d event $\mathcal{E}$ holds for all $\mathcal{S}$ and $b_i \in G$ with probability at least $1 - 2^{(n+1)d+1} e^{-Ct} \geq 1 - 2^{(n+1)d+1} 2^{-\frac{C}{4} 2^{n/2^{11}}} = 1 - o(1)$. Thus, there exists some set $\mathcal{B}$ for which the statement of the lemma holds. ◀

Proof of Theorem 34. Let $\alpha = \frac{1}{3 \cdot 2^{11}}$ and $|\mathcal{A}| = k = 2^{\alpha n}$. Suppose Alice holds $x \in \mathcal{A} \subseteq \{0,1\}^n$ and Bob holds $y \in \{0,1\}^n$. Recall that in the one-way communication model for XOR functions, Alice must pass a message of minimal length to Bob, who must then compute $f(x \oplus y)$ with some probability, say $\frac{2}{3}$. Here, we let specifically let f be a scaling of a matroid rank function, which is some monotone non-negative $\left(\frac{c_1}{n}\right)$-Lipschitz submodular function. By Yao's principle, it suffices to show that every deterministic one-way communication protocol using at most $\frac{\alpha}{4} n$ bits fails with probability greater than $\frac{1}{3}$ over $\mathcal{A}$. Suppose by way of contradiction, that Alice and Bob succeed through a deterministic one-way communication protocol, using at most $\frac{\alpha}{4} n$ bits. For the purpose of analysis, we furthermore suppose that Bob's input is fixed.

We now claim that if Alice passes a message to Bob using at most $\frac{\alpha}{4} n$ bits, then there are at least $2^{\alpha n} - 4 \cdot 2^{\alpha n/4}$ points in $\mathcal{A}$ that are represented by the same message as at least five other points. Note that Alice can partition the input space $\mathcal{A}$ into at most $2^{\alpha n/4}$ parts, each part with its own distinct representative message. The number of points *not* in parts containing at least five other points is at most $4 \cdot 2^{\alpha n/4}$. The remaining points, at least $2^{\alpha n} - 4 \cdot 2^{\alpha n/4}$ in quantity, are represented by the same message as at least five other points.

Let S be the set of points in $\mathcal{A}$ represented by a given message from Alice. Hence, Alice assigns the same message to each of these points and passes the state of the protocol to Bob. Because Bob cannot distinguish between these points and must perform a deterministic protocol, then Bob must output the same result for each of these points. Recall that we consider Bob's input $y \in \{0,1\}^n$ as fixed. Consider the family of functions

$$\mathcal{F} = \{f : f(x \oplus y) = b \text{ or } f(x \oplus y) = D \text{ for all } x \in \mathcal{A}\}.$$

Thus, if S contains at least five points, there exists $f \in \mathcal{F}$ such that Bob errs on at least $\frac{2}{5}$ fraction of the points in S by setting $f(x \oplus y) = b$ to at least $\left\lfloor \frac{|S|-1}{2} \right\rfloor$ of the points $x \in S$ and similarly for $f(x \oplus y) = D$. Moreover, since Alice partitions the points in $\mathcal{A}$, then there exists an $f \in \mathcal{F}$ such that Bob errs on at least $\frac{2}{5}$ fraction on *all* points that are represented by the same message as at least five other points. Hence, the total number of inputs that Bob errs is at least $\frac{2}{5}\left(2^{\alpha n} - 6 \cdot 2^{\alpha n/4}\right) > \frac{1}{3} \cdot 2^{\alpha n}$ for sufficiently large values of n. This contradicts the assumption that the communication protocol, using at most $\frac{\alpha}{4}n$ bits, succeeds with probability $\frac{2}{3}$. ◀

Streaming Verification of Graph Computations via Graph Structure

Amit Chakrabarti
Dartmouth College, Hanover, NH, USA

Prantar Ghosh
Dartmouth College, Hanover, NH, USA

Abstract

We give new algorithms in the annotated data streaming setting – also known as verifiable data stream computation – for certain graph problems. This setting is meant to model outsourced computation, where a space-bounded verifier limited to sequential data access seeks to overcome its computational limitations by engaging a powerful prover, without needing to trust the prover. As is well established, several problems that admit no sublinear-space algorithms under traditional streaming do allow protocols using a sublinear amount of prover/verifier communication and sublinear-space verification. We give algorithms for many well-studied graph problems including triangle counting, its generalization to subgraph counting, maximum matching, problems about the existence (or not) of short paths, finding the shortest path between two vertices, and testing for an independent set. While some of these problems have been studied before, our results achieve new tradeoffs between space and communication costs that were hitherto unknown. In particular, two of our results disprove explicit conjectures of Thaler (ICALP, 2016) by giving triangle counting and maximum matching algorithms for n-vertex graphs, using $o(n)$ space and $o(n^2)$ communication.

2012 ACM Subject Classification Theory of computation → Streaming models; Theory of computation → Interactive proof systems; Computer systems organization → Cloud computing

Keywords and phrases data streams, interactive proofs, Arthur-Merlin, graph algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.70

Category RANDOM

Related Version https://eccc.weizmann.ac.il/report/2019/101/

Funding This work was supported in part by NSF under award CCF-1907738.

1 Introduction

A major philosophical message of theoretical computer science is that a computationally bounded entity can greatly expand its space of tractable problems with access to a more powerful entity, *without* having to trust the latter. The celebrated IP = PSPACE [28] and PCP Theorems [3, 4] are perhaps the best known such results. In the realm of space-efficient computations on large data streams, there is a growing trend towards results of this flavor [26]. In this case, the powerful entity (henceforth named Prover) is often thought of as a cloud computing service that is free of the space limitations that the computationally bounded data streaming process (henceforth named Verifier) is subject to. This work designs new algorithms for graph computations on data streams in such Verifier/Prover models and proves some related complexity-theoretic results.

Early works on such "prover-enhanced data streaming algorithms" considered the *annotated streams* model [10, 22], where Prover reads the input data stream together with Verifier and, during stream processing and/or at the end, supplies Verifier with a *proof* (streamed to him) that convinces him of the correct answer to what he wants to compute on the stream. Subsequent works [11, 14] considered a more general model of *streaming interactive proofs* (SIPs), where the communication between Verifier and Prover is more general, rather


© Amit Chakrabarti and Prantar Ghosh;
licensed under Creative Commons License CC-BY



than one way. Several recent works in the annotated stream and the SIP models have focused on basic algorithmic problems on graphs [2, 13, 29], often giving sublinear-space algorithms for problems that provably do not admit sublinear solutions in the basic (sans prover) streaming setting.

In this work, we give new algorithms in the annotated streaming setting for certain graph problems, including triangle counting, its generalization to subgraph counting, maximum matching, problems about the existence (or not) of short paths, finding the shortest path between two vertices, and testing for an independent set. Two of our results provide "unexpected" new upper bounds, disproving published conjectures [29] asserting that such bounds would be unattainable.

1.1 Our Results and Techniques

Background and Motivation. Suppose that we wish to compute a function $f(\sigma)$ on an input stream σ consisting of *tokens* from some universe. For instance, for a graph computation, σ could be a stream of vertex pairs (u, v) specifying the input graph's edges, or it could be a stream of edge insertions and/or deletions to an evolving (multi)graph. Following established terminology [10], an *online scheme* is a protocol between Prover and Verifier wherein they observe σ together and, after each token appears, Prover provides zero or more bits of "help" to Verifier (as specified by the protocol). After σ is fully consumed, if Prover has followed the protocol faithfully, Verifier is very likely to output $f(\sigma)$; otherwise, he is very likely to "reject." If Verifier does all his work using at most $O(v)$ bits of working memory and Prover sends at most $O(h)$ bits of help, we call this an (h, v)-*scheme*.[1] A scheme is interesting if we can use $h > 0$ to achieve a value of v asymptotically smaller than what is feasible or known for a basic streaming algorithm, where $h = 0$.

All interesting schemes from previous work in fact use the prover in a more restricted way: Verifier processes all of σ on his own and *then* interacts with Prover. This work continues the tradition. There is practical motivation for building this restriction into the model of computation. Think of a cloud computing service where compute cycles are available only at certain times of day, or need to be booked in advance, whereas the client needs to access and process the input earlier, when it is made available to him. In such a setting, a scheme is most useful if the client can do its own processing first and wait for its time slot with the cloud service to finalize its computation.

Further, we focus only on *schemes*, which feature a single streamed message from Prover to Verifier, rather than the more general setting of SIPs, which allow rounds of interaction. This too is practically motivated: the cloud service need not dedicate a chunk of time to interact with the client, but need only promise that it will perform its portion of the computation *by* an agreed-upon deadline, at which time the client will download the "proof" it has constructed. In view of this latter style of computation, we also consider *multi-pass schemes*, where Verifier may use a "few" passes over its input σ and later receive a single streamed message from Prover, after which he produces his output. Most of our schemes will be single-pass (and we shall call them simply "schemes"), but in a few cases, we will give multi-pass schemes when they can achieve provably better costs than single-pass schemes.

Setup and Terminology. All problems studied in this paper involve an input graph, multigraph, or digraph $G = (V, E)$ on the vertex set $[n] := \{1, 2, \ldots, n\}$. We shall reserve the

[1] We will drop the qualifier "online" and simply call our protocols "schemes" because we will not be considering the more powerful setting of "prescient schemes" [10] in this paper.

basic term "graph" for simple, undirected graphs. The input is described either as a stream of edges (the default case) or as a stream of edge insertions and deletions: the latter type of stream is called a *dynamic* or *turnstile* graph stream. For an (h, v)-scheme to be interesting we at least require $v = o(n^2)$. If we also have $h = o(n^2)$, we call it a *sublinear* scheme. If we have $v = o(n)$ while $h = o(n^2)$, we call it a *frugal* scheme. This is an especially interesting setting of parameters, because most interesting graph problems provably require $\Omega(n)$ space in the basic streaming setting [15]. A frugal scheme shows that one can beat this space bound with the aid of only a sublinear-length proof. Recall that while $h \gg v$ is allowed, the proof must be processed using only $O(v)$ space.

For an (h, v)-scheme we refer to h as its hcost (short for "help cost") and v as its vcost (short for "verification cost"). We use the notation $[h, v]$-scheme as a shorthand for an $(\tilde{O}(h), \tilde{O}(v))$-scheme.[2] An $[n, n]$-scheme is called a *semi-streaming* scheme.

Subgraph Counting. The literature on graph streaming contains many works on the central problem of triangle counting (henceforth, TriangleCount): given a multigraph G as a dynamic stream, compute T, the number of triangles in G [6, 7, 18, 25, 29]. In Section 3, we study this and the more general problem of subgraph-counting (SubgraphCount$_k$) [7, 19, 20, 29], where the goal is to compute T_H, the number of copies of a fixed, k-sized graph H, where k is a constant. In the basic streaming model, computing T or T_H exactly is impossible in sublinear space and it becomes necessary to approximate. In contrast, we design a family of $(o(n^2), o(n))$-schemes for TriangleCount that give exact answers. Such a frugal scheme had been conjectured not to exist [29]. We extend our ideas to give sublinear $(o(n^2), o(n^2))$-schemes for SubgraphCount$_k$.

Maximum Matching. Determining $\alpha'(G)$, the cardinality of a maximum-sized matching in G, is a central problem in graph algorithms and has received a lot of attention in the recent literature on streaming algorithms [5, 12, 15, 16, 21, 24]. In Section 4, we consider this problem (henceforth, MaxMatching) for multigraphs given by dynamic streams. As with TriangleCount, we give a frugal scheme for MaxMatching, which had been conjectured to be impossible [29]. In the process, we present a frugal scheme for the subproblem of verifying that the purported connected components of a graph are indeed disconnected from each other, which might be of independent interest for future work on connectivity-related problems.

Independent Sets and Length-Three Paths. In Section 5, we study the independent set testing problem (IndSetTest), where we are given a multigraph G and a set $U \subseteq V$ (also streamed and interleaved with the edge stream arbitrarily) and we must determine whether or not U is independent. We also study the st-3Path problem, where G (which might be a digraph) has two designated vertices v_s and v_t and we must determine whether G has a path of length at most 3 from v_s to v_t. By results from prior work, any (h, v)-scheme for these problems must have total cost $h + v = \Omega(n)$. We therefore design two-pass schemes for these problems, achieving $h + v = \tilde{O}(n^{2/3})$. In fact, we obtain a more general tradeoff, giving a two-pass $[t^2, s]$-scheme for any parameters t, s with $ts = n$. Our schemes instantiate a protocol for the abstract problem CrossEdgeCount, which asks for a count of the number of edges in G from $U \subseteq V$ to $W \subseteq V$, where these sets U and W are also streamed.

[2] The notation $\tilde{O}(\cdot)$ hides factors polynomial in $\log n$.

In each case, we *can* design ordinary (one-pass) schemes with the same complexity parameters under a natural assumption on the way the stream is ordered, and these schemes still beat the space bound achievable by basic (sans prover) streaming algorithms.

Short Paths and Shortest Path. Finally, in Section 6, we consider shortest path problems, perhaps the most basic problem in classic graph algorithms. We study the ST-kPATH problem, which is to detect whether or not G has a path of length at most k from v_s to v_t, where k, v_s, and v_t are prespecified. We first present a $[kn, n]$-scheme for ST-kPATH. This gives a semi-streaming scheme for detecting short (of length polylogarithmic in n) paths, which is optimal in terms of total cost. It also implies a $[kn, n]$-scheme for ST-SHORTESTPATH problem – where k is the length of the shortest path from v_s to v_t – which is to find the shortest path between vertices v_s and v_t, and output NO if none exists. For directed graphs of small (polylogarithmic in n) diameter, it implies a semi-streaming scheme for checking s,t-connectivity. Note that these problems require $\Omega(n^2)$ space in the basic data streaming model, even for constant k or constant-diameter graphs [15].

Targeting a different cost regime, we generalize our result for ST-3PATH from Section 5 to obtain multi-pass (h, v)-schemes for ST-kPATH with total cost $h + v = o(n)$, for constant k. To be precise, we present a $\lceil k/2 \rceil$-pass $[n^{1-1/k}, n^{1-1/k}]$-scheme for ST-kPATH.

Our Techniques. Similar to past work in the area of streaming verification – indeed, hearkening back to classic interactive proof protocols [23, 28] – our schemes make heavy use of "arithmetization," i.e., they recast the underlying problem in terms of evaluating certain polynomials and exploit the encoding properties of polynomials (as captured in the Schwartz-Zippel Lemma) to protect the verifier from a cheating prover. Also as in past work, we use what we call the *shaping technique*, where we conceptually shape a data vector into an array with two or more dimensions. This seemingly innocuous trick allows us to consider input data as a table of values of a *multivariate* polynomial and we can use the "separation" afforded by these multiple variables to divide up work between Verifier and Prover.

The novelty in our algorithms comes from a twist on the shaping technique that was hitherto unexploited. At a high level, almost all earlier annotation schemes or SIPs for graph problems viewed the edge stream as a flat vector (a characteristic vector in the case of graphs or a frequency vector in the case of multigraphs). We crucially exploit the fact that the index set of this vector has additional structure: it consists of pairs of *vertices* and these vertices are very meaningful entities in the context of graph problems. Simply put, we exploit *graph structure* more fully in our use of the shaping technique.

Another novel feature of our schemes is that they involve Prover sending *multivariate* polynomials; their correctness analysis then involves the full multivariate strength of the Schwartz-Zippel Lemma. In all past work on interactive proofs and schemes, Prover only sent univariate polynomials (and the corresponding analyses used the more basic statement that a nonzero, degree-d, univariate polynomial has at most d roots). Thus, our scheme designs can be seen as exploiting the power of arithmetization more fully.

1.2 Related Work

The annotated data streaming model of computation was motivated in part by the need to develop a theory to capture ideas such as the stream punctuations of Tucker et al. [30] and the stream outsourcing framework of Yi et al. [31]. Chakrabarti et al. [10] formulated the model and provided the first theoretical results, focusing largely on the traditional statistical problems of frequency moments and heavy hitters, but also giving a handful of basic results for graph problems. Other early works in the same model include Klauck and Prakash [22]

and Chakrabarti et al. [9]. Cormode et al. [14] generalized the model to SIPs, which allow a few interactive rounds of communication between Verifier and Prover; this generalized setting was studied further in Chakrabarti et al. [11] and Abdullah et al. [2]. We refer the reader to the expository article of Mitzenmacher and Thaler [26] for a more detailed survey of this area.

We turn to graph computations and the specific problems studied in this work. For simplicity, we state complexities in terms of n alone, rather than using both m and n (m being the number of edges of the input graph). Cormode et al. [13] gave annotated data stream algorithms (schemes, in our terminology) for many canonical graph problems, often exploiting linear programming formulations of the problems. In particular, they gave an $[n^2, 1]$-scheme For MaxMatching. For a weighted version of st-ShortestPath, on simple graphs (not multigraphs) they gave $[h, v]$-schemes with $hv \geqslant dn^2$ and $h \geqslant dn$, where d is the maximum distance to *any* node reachable from v_s. Contrast this with our $[kn, n]$-scheme for unweighted multigraphs, where k is the length of the shortest v_s, v_t-path.

Thaler [29] studied the problems TriangleCount, MaxMatching, and SubgraphCount$_k$. He gave semi-streaming schemes for the first two. In the same paper, he explicitly conjectured that these two problems would not admit frugal schemes: he imagined that achieving vcost $= o(n)$ would bump up the hcost to $\Omega(n^2)$. Our results here disprove these conjectures. For SubgraphCount$_k$, Thaler gave a $[k^3 n, kn]$-SIP with $k - 2$ rounds of interaction. We achieve sublinear cost with just a single Prover-to-Verifier message. Sublinear schemes for SubgraphCount$_k$ were hitherto unknown for any $k > 3$.

For the TriangleCount problem, Chakrabarti et al. [10] gave an $[h, v]$-scheme for any h, v with $hv = n^3$, and also an $[n^2, 1]$-scheme. For the same problem, Abdullah et al. [2] gave a $(\log^2 n, \log^2 n)$-SIP that uses $\log n$ rounds of interaction, and a $(n^{1/\gamma} \log n, \log n)$-SIP with $\gamma = O(1)$ rounds of interaction. The latter paper also studied MaxMatching, giving a $(\rho + n^{1/\gamma'} \log n, \log n)$-SIP with γ rounds of interaction, where γ' is a linear function of γ, and ρ is the weight of an optimal matching (weighted or unweighted).

Guruswami and Onak [17] show a space lower bound of $\Omega(n^{1+\Omega(1/k)}/k^{O(1)})$ for st-kPath (where k is even) in $k/2 - 1$ passes in the basic streaming model. In contrast, our results show that, for any k, with the help of a prover, one can get a total cost of $\tilde{O}(n^{1-1/k})$ in $\lceil k/2 \rceil$ passes.

2 Preliminaries

For a positive integer n, we denote the set $\{1, 2, \ldots, n\}$ by $[n]$ and the set $\{-n, -n+1, \ldots, n-1, n\}$ by $[\![n]\!]$. The ring of polynomials in variables $X_1, \ldots, X_k$ with coefficients in the ring R is denoted by $R[X_1, \ldots, X_k]$. If S is a finite set, we write $r \in_R S$ to say that r is a random element drawn uniformly from S. In an undirected graph $G = (V, E)$, the *ith neighborhood* of a vertex v is the set of vertices u such that there is a walk of length i from v to u. We denote this set by $N_i(v)$. We put $N(v) := N_1(v)$ and $N[v] := N_1(v) \cup \{v\}$.

Following Chakrabarti et al. [10], an annotated data streaming algorithm, a.k.a. *scheme*, is a pair $\mathcal{A} = (\mathfrak{h}, \mathcal{B})$, where $\mathfrak{h}$ is a help function and $\mathcal{B}$ is a data stream algorithm that computes a function f of an input $\mathbf{x} \in \mathcal{U}^m$, where $\mathcal{U}$ is some universe. We see $\mathfrak{h}$ as an m-tuple $(\mathfrak{h}_1, \ldots, \mathfrak{h}_m)$, where $\mathfrak{h}_i : \mathcal{U}^i \to \{0,1\}^*$ is the *annotation* provided to $\mathcal{B}$ after the ith stream update x_i, depending on the elements seen so far, i.e. $x_1, \ldots, x_i$. Thus, $\mathcal{B}$ sees the *annotated* stream $\mathbf{x}^{\mathfrak{h}} := (x_1, \mathfrak{h}_1(x_1), x_2, \mathfrak{h}_2(x_1, x_2), \ldots, x_m, \mathfrak{h}_m(x_1, \ldots, x_m))$. Using a random string R, it processes this annotated stream, giving an output $\text{out}(\mathcal{B}; \mathbf{x}^{\mathfrak{h}}, R)$. We say that $\mathcal{A}$ is a δ-*error* scheme if

- (completeness) for all $\mathbf{x} \in \mathcal{U}^m$: $\Pr_R[\text{out}(\mathcal{B}; \mathbf{x}^{\mathfrak{h}}, R) \neq f(\mathbf{x})] \leqslant \delta$; and

- (soundness) for all $\mathbf{x} \in \mathcal{U}^m$, $\mathfrak{h}' = (\mathfrak{h}'_1, \mathfrak{h}'_2, \ldots, \mathfrak{h}'_m) \in (\{0,1\}^*)^m$:
$$\Pr_R[\text{out}(\mathcal{B}; \mathbf{x}^{\mathfrak{h}'}, R) \notin \{f(\mathbf{x}), \perp\}] \leqslant \delta,$$

where "$\perp$" is a special symbol indicating that $\mathcal{B}$ rejects the annotation (proof) provided, having detected cheating. When δ is left unspecified, we assume a default value of 1/3. The hcost (help cost) of $\mathcal{A}$ is $\max_{\mathbf{x}\in\mathcal{U}^m} \sum_i |\mathfrak{h}_i(\mathbf{x})|$, and the vcost (verification cost) is the space usage of $\mathcal{B}$.

The scheme $\mathcal{A}$ is said to be an (h, v)-scheme (resp. $[h, v]$-scheme) if its hcost is $O(h)$ (resp. $\tilde{O}(h)$) and its vcost is $O(v)$ (resp. $\tilde{O}(v)$). The sum hcost + vcost is called the *total cost* of $\mathcal{A}$. In the context of problems on n-vertex graphs, an $(o(n^2), o(n^2))$-scheme is called a *sublinear* scheme, an $[n, n]$-scheme is called a *semi-streaming* scheme and an $(o(n^2), o(n))$-scheme is called a *frugal* scheme.

A *multi-pass scheme* – more precisely, a p-pass scheme with $p \geqslant 2$ – is a scheme $\mathcal{A} = (\mathfrak{h}, \mathcal{B})$ where $\mathcal{B}$ makes $p-1$ passes over the input $\mathbf{x}$ followed by a final pass over the annotated stream $\mathbf{x}^{\mathfrak{h}}$. As discussed in Section 1.1, all schemes and multi-pass schemes we design in this work have the feature that the entire annotation $\mathfrak{h}(\mathbf{x})$ arrives only after $\mathcal{B}$ is done processing the plain stream $\mathbf{x}$. That said, the negative results in this work do not require the scheme to be restricted in this way.

Let f be a k-dimensional array with dimensions $(s_1, \ldots, s_k)$ each of whose entries is an integer in $[\![M]\!]$. Equivalently, we have a function $f\colon [s_1] \times \cdots \times [s_k] \to [\![M]\!]$. For a finite field $\mathbb{F}$ of sufficiently large characteristic,[3] we define the $\mathbb{F}$-extension of f to be the unique polynomial $\tilde{f}(X_1, \ldots, X_k) \in \mathbb{F}[X_1, \ldots, X_k]$ such that

- for all $(x_1, \ldots, x_k) \in [s_1] \times \cdots [s_k]$, we have $\tilde{f}(x_1, \ldots, x_k) = f(x_1, \ldots, x_k)$, and
- for all $i \in [k]$, we have $\deg_{X_i} \tilde{f} \leqslant s_i - 1$.

Note that $\tilde{f}$ can be described explicitly using Lagrange interpolation:

$$\tilde{f}(X_1, \ldots, X_k) = \sum_{(u_1,\ldots,u_k)\in[s_1]\times\cdots\times[s_k]} f(u_1, \ldots, u_k)\, \delta_{u_1,\ldots,u_k}(X_1, \ldots, X_k)\,, \quad \text{where} \tag{1}$$

$$\delta_{u_1,\ldots,u_k}(X_1, \ldots, X_k) = \prod_{i=1}^{k} \prod_{x_i\in[s_i]\setminus\{u_i\}} (u_i - x_i)^{-1}(X_i - x_i)\,. \tag{2}$$

In particular, if f is built up from a stream of pointwise updates, where the jth update adds Δ_j to entry $(u_1, \ldots, u_k)_j$ of the array, then

$$\tilde{f}(X_1, \ldots, X_k) = \sum_j \Delta_j\, \delta_{(u_1,\ldots,u_k)_j}(X_1, \ldots, X_k)\,. \tag{3}$$

This leads to the following fact that we use in all our protocols. For details and a thorough discussion, including implementation considerations, see Cormode et al. [14].

▶ **Fact 1.** *Given a point $(p_1, \ldots, p_k) \in \mathbb{F}^k$ and a stream of pointwise updates to an array with dimensions $(s_1, \ldots, s_k)$ that is initially all-zero, we can keep track of the value $\tilde{f}(p_1, \ldots, p_k)$ using $O(\log |\mathbb{F}|)$ space, performing $O(k)$ field arithmetic operations after each update.*

We record results proved in Chakrabarti et al. [9, 10] that can be seen as generalizing the Aaronson-Wigderson protocol for Merlin-Arthur communication complexity of set disjointness [1].

[3] We need the characteristic to be at least $\max\{s_1, \ldots, s_k, 2M+1\}$ to avoid "wrap around problems," i.e., to ensure that all integers in each $[s_i]$ as well as all integers in $[\![M]\!]$ have distinct images under the ring homomorphism from $\mathbb{Z}$ to $\mathbb{F}$.

▶ **Fact 2** (SUBSET and INTERSECTION schemes; Prop. 4.1 of [10] and Thm. 5.3 of [9]). *Consider a stream consisting of elements of two sets $S, T \subseteq [N]$ interleaved arbitrarily. Then, for any h, v with $hv \geqslant N$, there are $[h, v]$-schemes to compute $|S \cap T|$ and to determine whether $S \subseteq T$. For the latter problem, there is a $[\ell h, v]$-scheme handling the more general setting where S and T are multisets updated dynamically by the stream and the multiplicity of each element is at most ℓ.*

▶ **Fact 3** (Schwartz-Zippel Lemma). *For a nonzero polynomial $P(X_1, \ldots, X_n) \in \mathbb{F}[X_1, \ldots, X_n]$ of total degree d, where $\mathbb{F}$ is a finite field,* $\Pr_{(r_1,\ldots,r_n)\in_R \mathbb{F}^n}[P(r_1, \ldots, r_n) = 0] \leqslant d/|\mathbb{F}|$.

3 Subgraph Counting

We begin by describing a frugal scheme for TRIANGLECOUNT and then extend our ideas to obtain a sublinear scheme for the more general problem SUBGRAPHCOUNT. Throughout, we assume that the input is an n-vertex multigraph $G = (V, E)$ with adjacency matrix A, built up through a stream of edge insertions and deletions.

3.1 Triangle Counting

Let $T = T(G)$ be the number of triangles in G taking edge multiplicities into account, i.e., two triangles are considered distinct iff their corresponding sets of *edges* are distinct. Then,

$$6T = \sum_{v_1,v_2,v_3 \in V} A_{v_1v_2} A_{v_2v_3} A_{v_3v_1} \,. \tag{4}$$

Let t and s be integer-valued parameters such that $ts = n$. Using a canonical bijection, we represent each vertex $v \in V$ by a pair of integers $(x, y) \in [t] \times [s]$. This transforms the matrix A into a 4-dimensional array a, given by $a(x_1, y_1, x_2, y_2) = A_{v_1v_2}$. Let $\tilde{a}$ be the $\mathbb{F}$-extension of a for a sufficiently large finite field $\mathbb{F}$ to be chosen later. Equation (4) now gives

$$6T = \sum_{x_1,x_2,x_3 \in [t]} p(x_1, x_2, x_3) \,, \quad \text{where} \tag{5}$$

$$p(X_1, X_2, X_3) = \sum_{y_1,y_2,y_3 \in [s]} \tilde{a}(X_1, y_1, X_2, y_2)\, \tilde{a}(X_2, y_2, X_3, y_3)\, \tilde{a}(X_3, y_3, X_1, y_1) \,. \tag{6}$$

Note that, for each $i \in \{1, 2, 3\}$, we have $\deg_{X_i} p \leqslant 2t - 2$. Thus, the number of monomials in p is at most $(2t - 1)^3 \leqslant 8t^3$ and the total degree $\deg p \leqslant 6t - 6 \leqslant 6t$.

Our scheme for triangle counting operates as follows.

Stream processing. Verifier starts by picking $r_1, r_2, r_3 \in_R \mathbb{F}$. As the edge stream arrives, he maintains the three 2-dimensional arrays $\tilde{a}(r_1, w, r_2, z)$, $\tilde{a}(r_2, w, r_3, z)$, and $\tilde{a}(r_3, w, r_1, z)$, for all $(w, z) \in [s] \times [s]$ (using Fact 1). At the end of the stream, he uses these arrays to compute $p(r_1, r_2, r_3)$, using Equation (6).

Help message. Prover sends Verifier a polynomial $\hat{p}(X_1, X_2, X_3)$ that she claims equals $p(X_1, X_2, X_3)$; in particular, for each $i \in \{1, 2, 3\}$, $\deg_{X_i} \hat{p} \leqslant 2t - 2$. She streams the coefficients of $\hat{p}$ one at a time, according to some canonical ordering of the possible monomials.

Verification and output. As $\hat{p}$ is streamed in, Verifier computes the check value $C := \hat{p}(r_1, r_2, r_3)$ and the result value $\hat{T} := \frac{1}{6} \sum_{x_1,x_2,x_3 \in [t]} \hat{p}(x_1, x_2, x_3)$. If he finds that $C \neq p(r_1, r_2, r_3)$, he outputs $\perp$. Otherwise, he believes that $\hat{p} \equiv p$ and accordingly, based on Equation (5), outputs $\hat{T}$ as the answer.

The analysis of this scheme is along now-standard lines.

Error probability. Clearly, if Prover is honest (i.e., $\hat{p} \equiv p$), then the output is always correct. So the scheme errs only when $\hat{p} \not\equiv p$ but Verifier's check passes. This means that the random point $(r_1, r_2, r_3) \in \mathbb{F}^3$ is a root of the nonzero polynomial $\hat{p} - p$, which has total degree at most $6t$. By the Schwartz-Zippel Lemma (Fact 3), the probability of this event is at most $6t/|\mathbb{F}| < 1/n$, by choosing $|\mathbb{F}|$ large enough.

Help and Verification costs. The number of bits used to describe the polynomial $\hat{p}$ is the hcost. As noted, the polynomial $\hat{p}$ has $O(t^3)$ many coefficients, each of which is an element of $\mathbb{F}$, and hence has size $O(\log n)$. So the hcost is $\tilde{O}(t^3)$. The Verifier maintains three $s \times s$ arrays, where each entry is an element of $\mathbb{F}$. Hence, the vcost is $\tilde{O}(s^2)$. Therefore, we get a $[t^3, s^2]$-scheme for triangle counting, for parameters t, s with $ts = n$. Setting $t = n^\alpha$ for $\alpha \in (1/2, 2/3)$, we get a $(o(n^2), o(n))$-scheme, which is frugal.

The result in this section is captured in the theorem below.

▶ **Theorem 4.** *For any parameters t, s with $ts = n$, there is a $[t^3, s^2]$-scheme for* TriangleCount. *In particular, there is an $(o(n^2), o(n))$-scheme for* TriangleCount.

This disproves Thaler's conjecture [29], which stated that TriangleCount has no frugal scheme.

3.2 Generalization to Counting Copies of an Arbitrary Subgraph

Now we consider the SubgraphCount$_k$ problem. Let H be a fixed k-vertex graph. The goal is to determine $T_H = T_H(G)$, the number of copies of H in the n-vertex multigraph G given by an input stream: n is growing whereas $k = O(1)$. As before, we take edge multiplicities into account.

Fix a numbering of the vertices of H as $1, 2, \ldots, k$. Write $i \sim j$ to denote $\{i, j\} \in E(H) \wedge i < j$. To generalize Equation (4), note that the expression $\prod_{i \sim j} A_{v_i v_j}$ counts the number of copies of H occurring amongst vertices $v_1, \ldots, v_k$ in G where $i \in V(H)$ is mapped to $v_i \in V$, *provided that $v_1, \ldots, v_k$ are distinct.* This subtlety of explicitly requiring the v_is to be distinct did not arise for TriangleCount because $A_{v_1 v_2} A_{v_2 v_3} A_{v_3 v_1}$ is zero unless v_1, v_2, v_3 are distinct. To enforce the distinctness condition in our more general setting, define an $n \times n$ Boolean matrix B by $B_{uv} = 1$ iff $u \neq v$. Then, defining α_H to be the number of automorphisms of H,

$$\alpha_H T_H = \sum_{v_1, \ldots v_k \in V} \left(\prod_{i \sim j} A_{v_i v_j} \right) \left(\prod_{i \neq j \in [k]} B_{v_i v_j} \right) . \tag{7}$$

As before, we shape V into $[t] \times [s]$ for parameters t and s with $ts = n$. This turns the 2-dimensional matrices A, B into 4-dimensional arrays a, b, which in turn have $\mathbb{F}$-extensions $\tilde{a}, \tilde{b}$. Equation (7) gives

$$\alpha_H T_H = \sum_{x_1, \ldots, x_k \in [t]} p(x_1, \ldots, x_k) , \quad \text{where} \tag{8}$$

$$p(X_1, \ldots, X_k) = \sum_{y_1, \ldots, y_k \in [s]} \left(\prod_{i \sim j} \tilde{a}(X_i, y_i, X_j, y_j) \right) \left(\prod_{i \neq j \in [k]} \tilde{b}(X_i, y_i, X_j, y_j) \right) . \tag{9}$$

For each $i \in [k]$, $\deg_{X_i} p \leqslant 2(k-1)(t-1) = O(t)$. So the total degree $\deg p = O(t)$ and p has at most $O(t^k)$ monomials. This leads to a scheme for subgraph counting that naturally generalizes our earlier scheme for triangle counting. We sketch the salient features and the analysis.

Stream processing. Verifier picks $r_1, \ldots, r_k \in_R \mathbb{F}$ and maintains (using Fact 1) $O(k^2) = O(1)$ many $s \times s$ arrays: $\tilde{a}(r_i, w, r_j, z)$ for each $i \sim j \in [k]$ and $\tilde{b}(r_i, w, r_j, z)$ for each $i \neq j \in [k]$, where $(w, z) \in [s] \times [s]$. The $\tilde{b}$ arrays do not depend on the input stream and can be computed once and for all. At the end of the stream, he computes $p(r_1, \ldots, r_k)$ with the help of these values, using Equation (9).

Help message. Prover sends a polynomial $\hat{p}(X_1, \ldots, X_k)$ that she claims to be $p(X_1, \ldots, X_k)$. She streams the $O(t^k)$ coefficients of $\hat{p}$, using some canonical ordering of the monomials.

Verification and output. Verifier computes the check value $C := \hat{p}(r_1, \ldots, r_k)$ and the result value $\hat{T}_H := \alpha_H^{-1} \sum_{x_1,\ldots,x_k \in [t]} \hat{p}(x_1, \ldots, x_k)$. He outputs $\perp$ if $C \neq p(r_1, \ldots, r_k)$. Else, believing $\hat{p} \equiv p$, he outputs $\hat{T}_H$ as the answer, in view of Equation (8).

Error probability. By a Schwartz-Zippel Lemma (Fact 3) argument as before, the error probability is at most $\deg p/|\mathbb{F}| = O(t)/|\mathbb{F}| < 1/n$, by choosing $|\mathbb{F}|$ large enough.

Help and Verification costs. The hcost is $\tilde{O}(t^k)$, by the bound on the number of monomials in $\hat{p}$. Verifier stores $O(1)$ many $s \times s$ arrays, leading to a vcost of $\tilde{O}(s^2)$.

In summary, we obtain a $[t^k, s^2]$-scheme for counting copies of a fixed k-vertex subgraph H, for all choices of parameters t, s with $ts = n$. Setting $t = n^{2/(k+2)}$ and $s = n^{k/(k+2)}$ gives a scheme where both these costs are $\tilde{O}(n^{2k/(k+2)})$, which is $o(n^2)$ for constant k. Thus, we get the following theorem.

▶ **Theorem 5.** *For any parameters t, s such that $ts = n$, there is a $[t^k, s^2]$-scheme for* SubgraphCount$_k$*, where k is a constant. In particular, there is a sublinear scheme for* SubgraphCount$_k$ *with total cost $\tilde{O}(n^{2k/(k+2)})$.*

4 Maximum Matching

We now turn to the MaxMatching problem, again giving a frugal scheme. Our input is an edge stream of an n-vertex graph $G = (V, E)$ and we would like to determine $\alpha'(G)$, the cardinality of a maximum matching in G. We follow the broad outline of the semi-streaming scheme for MaxMatching by Thaler [29]. That scheme has two parts. In the first part, Prover convinces Verifier that $\alpha'(G) \geqslant k$, for some integer k. In the second part, she convinces him that $\alpha'(G) \leqslant k$. For the former, Prover simply provides a suitable matching M and convinces Verifier that $M \subseteq E$ using the subset scheme from Fact 2. For the latter, Prover uses the Tutte-Berge formula [8], which states that

$$\alpha'(G) = \frac{1}{2} \min_{U \subseteq V} \Big(|U| + |V| - \mathrm{odd}(G \setminus U) \Big), \tag{10}$$

where $\mathrm{odd}(G \setminus U)$ denotes the number of connected components in $G \setminus U$ with an odd number of vertices. The most challenging part of the scheme is evaluating $\mathrm{odd}(G \setminus U)$, which involves the sub-problem of verifying whether all the connected components of a graph (as claimed by the Prover) are disconnected from each other. Thaler comments that this is the part that acts as a barrier in reducing the vcost to $o(n)$ without increasing the hcost to $\Omega(n^2)$. We present a novel frugal scheme for this sub-problem. The rest of the protocol solves the same sub-problems as the aforementioned paper. Most of their sub-schemes for these sub-problems, however, were trivial for $\tilde{O}(n)$ space. We need schemes for the same problems that use only $o(n)$ space and hence require more work. We describe our protocol below.

To convince the Verifier that the size of a maximum matching in G is k, Prover proves that it is (a) at least k, and (b) at most k. For (a), she simply sends (as a stream) a set M of k edges that constitutes a matching of G. Verifier can easily check using $O(\log n)$ space that

the set has size k. Next, he needs to check that $M \subseteq E$, and that M is indeed a matching. For the former, we can use Fact 2 and get an $[h, v]$-scheme, where v is the $o(n)$ value we are aiming for and $h = n^2/v$. To verify that M is a matching, we check whether every vertex in M appears exactly once in this stream. Treating M as a stream of vertices, we can do this as follows: First, compute F_2, the second frequency moment of the stream, using an $[h, v]$-scheme where v is the $o(n)$ vcost we want, and $h = n/v$ ([10], Theorem 4.1). Next, verify that it equals $2k$ (this happens iff all $2k$ elements are distinct).

For (b), we apply Equation (10). Prover sends $U^* \subseteq V$ and claims that $k = 1/2 \cdot (|U^*| + |V| - \text{odd}(G \setminus U^*))$. To check this, Verifier just needs to compute $\text{odd}(G \setminus U^*)$. We do this in the following way.

Let $[C]$ be the set of C connected components of $G \setminus U^*$. For $c \in [C]$ and $u \in G \setminus U^*$, Prover sends an array L of pairs (c, u) such that $u \in c$. The array L is sorted in non-decreasing order of c, i.e., she first sends the vertices in connected component 1, followed by those in component 2, and so on. If L is indeed as Prover claims, then $\text{odd}(G \setminus U^*)$ is equal to the number of components c that arrive with odd number of vertices in L. Since L is sorted with respect to c, Verifier can count this number easily using $O(\log n)$ space. He can verify that the vertices in the tuples of L constitute $G \setminus U^*$, and that no vertex u is repeated in different tuples of L, using frugal schemes implied by the standard protocols mentioned above.

Thus, it only remains to verify that L is as claimed. For this, we need to check whether the following two properties hold:

- **(i)** For each $c \in [C]$, the vertices in $G \setminus U^*$ that are claimed to be in component c are all connected in $G \setminus U^*$.
- **(ii)** For every pair (u, v) of vertices in $G \setminus U^*$ that are claimed to be in different components, we have $(u, v) \notin E$.

For Property (i), Prover sends a spanning tree for each connected component c and Verifier can check if all of them are valid using an $[n^{1+\alpha}, n^{1-\alpha}]$-scheme, for any $\alpha \in [0, 1]$ ([10], Theorem 7.7) so as to get the desired $o(n)$ vcost.

Checking Property (ii) is the most challenging part. We give a novel protocol for this part that uses $o(n)$ vcost and $o(n^2)$ hcost. Slightly abusing notation, consider the array L in the form of a $C \times |G \setminus U^*|$ matrix, such that $L_{cu} = 1$ if $u \in c$, and $L_{cu} = 0$ otherwise. Denote the ones' complement of this matrix by $\bar{L}$. Let A be the adjacency matrix of $G \setminus U^*$. Finally, let γ denote the total number of cross edges that go between two connected components in $G \setminus U^*$. Then, we have

$$2\gamma = \sum_{\substack{c \in [C] \\ u,v \in G \setminus U^*}} L_{cu} \bar{L}_{cv} A_{uv} \,. \tag{11}$$

Property (ii) is satisfied iff $\gamma = 0$. Recalling that $C = O(n)$ and $|G \setminus U^*| = O(n)$, we note that Equation (11) has a similar form as that of Equation (4). Thus, it can be exploited in essentially the same way as the $[t^3, s^2]$-scheme for TriangleCount, for parameters t, s with $ts = n$. Once again, setting $t = n^\alpha$ for $\alpha \in (1/2, 2/3)$, we get a frugal scheme.

The next theorem summarizes the result in this section.

▶ **Theorem 6.** *For any parameters t, s with $ts = n$, there is a $[t^3, s^2]$-scheme for* MaxMatching. *In particular, there is an $(o(n^2), o(n))$-scheme for* MaxMatching.

This disproves yet another conjecture of Thaler [29], which stated that MaxMatching has no frugal scheme.

5 Counting Cross-edges and its Applications to Other Problems

Consider the problems IndSetTest and st-3Path defined in Section 1.1. The key task underlying these problems is counting the number of edges crossing between two subsets U and W of V that arrive in some adversarial streaming order along with the edges: for IndSetTest, U and W are the same set; for st-3Path, they are (closed) neighborhoods of the designated vertices v_s and v_t. This is precisely the abstract problem of CrossEdgeCount. Clearly, a scheme for this problem can be used as a subroutine to solve IndSetTest and st-3Path.

Any one-pass (h, v)-scheme for CrossEdgeCount, IndSetTest, or st-3Path must have $hv \geqslant n^2$ and hence, total cost $h + v = \Omega(n)$. In Appendix A.1, we outline how these lower bounds are obtained. We therefore consider *two-pass* schemes for these problems. In particular, we design such a scheme for CrossEdgeCount with total cost $\tilde{O}(n^{2/3})$ and apply it to obtain similar bounds for other graph problems. In Appendix A.2, we discuss these applications. We also note that our schemes can be implemented in one pass each, under natural assumptions on the way the stream is ordered; see Appendix A.3.

5.1 Two-pass Scheme for CrossEdgeCount

We now design a two-pass scheme for CrossEdgeCount, aiming for total cost $o(n)$.

Let $\gamma = \gamma(U, W, G)$ denote the number of Cross-edges between U and W in a (directed or undirected) graph G. Formally, it is the number of ordered pairs $(u, w) \in U \times W$ such that $(u, w) \in E$. Note that, in an undirected graph, γ counts an edge (u, w) with multiplicity 2 whenever $u, w \in U \cap W$. For some applications (e.g., counting number of 3-walks in an undirected graph), we *do* need to count them with multiplicity. We discuss later how we can remove this multiplicity if needed.

We describe a scheme that works even on turnstile graph streams, i.e., a stream of the vertices in U and W intermixed with *updates* to edge multiplicities. Let L and F denote the characteristic vectors of the sets U and W respectively and let A be the (weighted) adjacency matrix of G. Then,

$$\gamma = \sum_{u \in U, w \in W} L_u A_{u,w} F_w \,. \tag{12}$$

Let t and s be integer parameters such that $ts = n$. As usual, using a canonical bijection, we represent each vertex $v \in V$ by a pair of integers $(x, y) \in [t] \times [s]$. As a result, the vectors L, F transform into 2-dimensional arrays ℓ, f given by $\ell(x, y) = L_v$ and $f(x, y) = F_v$. As before, the adjacency matrix A turns into a 4-dimensional array a, such that $a(x_1, y_1, x_2, y_2) = A_{v_1 v_2}$. Let $\tilde{\ell}$, $\tilde{f}$ and $\tilde{a}$ be $\mathbb{F}$-extensions of ℓ, f and a respectively, for a sufficiently large finite field $\mathbb{F}$. Now, Equation (12) yields

$$\gamma = \sum_{x_1, x_2 \in [t]} p(x_1, x_2) \,, \quad \text{where} \tag{13}$$

$$p(X_1, X_2) = \sum_{y_1, y_2 \in [s]} \tilde{\ell}(X_1, y_1)\, \tilde{a}(X_1, y_1, X_2, y_2)\, \tilde{f}(X_2, y_2) \,. \tag{14}$$

For $i \in \{1, 2\}$, $\deg_{X_i} p = 2t - 2$. Thus, it follows that the number of monomials in p is at most $O(t^2)$, and the total degree of p is $O(t)$.

We are now ready to design a two-pass scheme for CrossEdgeCount.

Stream processing. Verifier first chooses $r_1, r_2 \in_R \mathbb{F}$. For $y \in [s]$, define

$$g(y) := \sum_{y' \in [s]} \tilde{a}(r_1, y, r_2, y') \tilde{f}(r_2, y') \tag{15}$$

Thus,

$$p(r_1, r_2) = \sum_{y \in [s]} \tilde{\ell}(r_1, y) g(y) \,. \tag{16}$$

Pass 1. Only process the vertices in L and F in the stream. Maintain (using Fact 1) two s-dimensional vectors: $\tilde{\ell}(r_1, y)$ and $\tilde{f}(r_2, y)$, where $y \in [s]$.

Pass 2. Only process the edges in the stream. We want to maintain the s-dimensional vector $g(y)$ so that we can compute $p(r_1, r_2)$ using Equation (16). Suppose that the jth edge update $(x_1, y_1, x_2, y_2)_j$ adds Δ_j to that edge's multiplicity. This results in updates to several entries of $\tilde{a}$, but we want to use only $O(s)$ space, so we cannot afford to maintain $\tilde{a}$ directly. Instead, for each $j \in [m]$, let g_j and $\tilde{a}_j$ denote the values of g and $\tilde{a}$ (respectively) after the jth stream update. Then

$$\begin{aligned} g_j(y) &= \sum_{y' \in [s]} \tilde{f}(r_2, y')\, \tilde{a}_j(r_1, y, r_2, y') \\ &= \sum_{y' \in [s]} \tilde{f}(r_2, y') \left(\tilde{a}_{j-1}(r_1, y, r_2, y') + \Delta_j\, \delta_{(x_1, y_1, x_2, y_2)_j}(r_1, y, r_2, y') \right) \qquad (17) \\ &= g_{j-1}(y) + h_j(y) \,, \end{aligned}$$

where Equation (17) follows from Equation (3) and

$$h_j(y) := \sum_{y' \in [s]} \tilde{f}(r_2, y')\, \Delta_j\, \delta_{(x_1, y_1, x_2, y_2)_j}(r_1, y, r_2, y') \,. \tag{18}$$

Hence, after the jth update, the Verifier can compute $h_j(y)$ and maintain the vector $g(y)$.

Help message. After the second pass, Prover sends a polynomial $\hat{p}(X_1, X_2)$ (as a stream of coefficients) that she claims equals $p(X_1, X_2)$.

Verification and output. At the end of the second pass, Verifier gets $g(y)_m = g(y)$ for each y. Now, he uses Equation (16) to compute the check value $p(r_1, r_2)$ and the result value $\hat{\gamma} := \sum_{x_1, x_2 \in [t]} \hat{p}(x_1, x_2)$. If he finds that $p(r_1, r_2) \neq \hat{p}(r_1, r_2)$, he outputs $\perp$. Otherwise, he believes that $\hat{p} \equiv p$ and exploiting Equation (13), outputs $\hat{\gamma}$ as the answer.

Now, we analyze the correctness and complexity parameters of the scheme.

Error probability. The protocol errs only when $\hat{p} \not\equiv p$, but Verifier's check passes. Then, $(r_1, r_2) \in \mathbb{F}^2$ must be a root of the nonzero polynomial $\hat{p} - p$. We noted that its total degree is $O(t)$. Thus, the Schwartz-Zippel Lemma bounds the error probability by at most $O(t)/|\mathbb{F}| < 1/n$, for large enough choice of $|\mathbb{F}|$.

Help and Verification costs. The polynomial $\hat{p}$ has $O(t^2)$ monomials, and so, the hcost is $\tilde{O}(t^2)$. Verifier stores constant many vectors of size s at a time and incurs a vcost of $\tilde{O}(s)$. Thus, we obtain a two-pass $[t^2, s]$-scheme for CrossEdgeCount, for parameters t, s with $ts = n$. Setting $t = n^{1/3}$ and $s = n^{2/3}$, we get a scheme with total cost $\tilde{O}(n^{2/3})$.

Finally, we discuss how one can count cross-edges between U and W when they are defined as unordered pairs. Define this problem as CrossEdgeCount-Uniq. Let γ' be the number of edges that γ counts with multiplicity 2, i.e., the number of undirected edges $(u, w) \in U \times W$ such that $u, w \in U \cap W$. Then,

$$\gamma' = \sum_{u \in U, w \in W} L_u F_u A_{u,w} L_w F_w \,. \tag{19}$$

Hence, we modify the definitions of $p(X_1, X_2)$ and $g(y)$ as

$$p(X_1, X_2) := \sum_{y_1, y_2 \in [s]} \tilde{\ell}(X_1, y_1)\tilde{f}(X_1, y_1)\, \tilde{a}(X_1, y_1, X_2, y_2)\, \tilde{\ell}(X_2, y_2)\tilde{f}(X_2, y_2)\,. \tag{20}$$

$$g(y) := \sum_{y' \in [s]} \tilde{a}(r_1, y, r_2, y')\tilde{\ell}(r_2, y')\tilde{f}(r_2, y'). \tag{21}$$

Then, proceeding as in CrossEdgeCount, we compute γ'. Thus, we can compute γ and γ' in parallel and finally output $\gamma - \gamma'$ as the answer to CrossEdgeCount-Uniq.

▶ **Theorem 7.** *For parameters t, s with $ts = n$, there are two-pass $[t^2, s]$-schemes for* CrossEdgeCount *and* CrossEdgeCount-Uniq. *In particular, there are two-pass schemes with total cost $\tilde{O}(n^{2/3})$. If the vertices appear first in the stream, we need only one pass.*

6 Path Problems

In this section, we focus on path-related problems. Specifically, we study st-kPath for $k \geqslant 3$ and the fundamental st-ShortestPath problem (defined in Section 1.1). It follows from simple reduction from INDEX_N for $N = n^2$ that a one-pass algorithm for both of these problems requires $\Omega(n^2)$ space in the basic (sans prover) streaming model, and a one-pass scheme requires a total cost of $\Omega(n)$. We present a scheme for st-kPath for general k that can also be used to solve st-ShortestPath. It is a semi-streaming scheme when k is polylogarithmic in n, and hence matches the lower bound (up to polylogarithmic factors). Next, we explore if we can break the $\Omega(n)$ barrier for schemes for st-kPath at the cost of allowing a few more passes over the input. We achieve this for constant k by generalizing the protocol for st-3Path. We present all our schemes for undirected graphs, but they can be very easily modified to work for directed graphs as well.

6.1 A Single-Pass Semi-Streaming Scheme for Detecting Short Paths

For st-3Path, it is easy to obtain a semi-streaming scheme by checking (using Fact 2) whether the set $N[v_s] \times N[v_t]$ and the edge set E are disjoint. For $k > 3$, it's not that direct and requires more work. We describe the protocol below for a multigraph G.

Let A denote the adjacency matrix of the graph G and let $\tilde{A}$ be the $\mathbb{F}$-extension of A, for some large finite field $\mathbb{F}$. For $u \in N_{i+1}(v_s)$, let $d_{u,i}$ be the number of (in-)neighbors of u in $N_i(v_s)$. It follows that

$$d_{u,i} = \sum_{v \in N_i(v_s)} A(v, u)\,. \tag{22}$$

We are now ready to describe the protocol.

Stream processing. Verifier picks $r \in_R \mathbb{F}$ and stores $\tilde{A}(v, r)$ for each $v \in [n]$, maintaining them dynamically as the stream arrives (Fact 1). He also stores the set $N_1(v_s)$.

Help message. At the end of the stream, Prover sends Verifier $k-1$ polynomials $\hat{p}_1, \ldots, \hat{p}_{k-1}$, and she claims $\hat{p}_i \equiv p_i$ for each $i \in [k]$, where

$$p_i(U) = \sum_{v \in N_i(v_s)} \tilde{A}(v, U)\,. \tag{23}$$

Verifier's computation. Verifier iteratively constructs $N_i(v_s)$ for $i \in [k]$. Each time after computing $N_i(v_s)$ for an i, he checks if t is in the set. If so, he stops and outputs YES. Otherwise, he proceeds to compute $N_{i+1}(v_s)$. If he finds that $t \notin N_i(v_s)$ for any $i \in [k]$, then he outputs NO. The inductive neighborhood computation is done as follows.

Assume that Verifier has the set $N_i(v_s)$ for some $i \in [k-1]$; this holds initially, since he has stored $N_1(v_s)$. He computes $p_i(r)$ using Equation (23) and checks whether $\hat{p}_i(r) = p_i(r)$. If the check passes, he believes that $\hat{p}_i \equiv p_i$ and evaluates $\hat{p}_i(u)$ for each $u \in V$. By Equation (22), $p_i(u)$ equals $d_{u,i}$, which is non-zero iff $u \in N_{i+1}(v_s)$. Hence, he sets $N_{i+1}(v_s) = \{u : \hat{p}_i(u) \neq 0\}$.

Error probability. The protocol errs when we have $\hat{p}_i \not\equiv p_i$ for some i, but Verifier's check passes. This implies that r is a root of the non-zero polynomial $\hat{p}_i - p_i$. For a given i, the total degree of this polynomial is at most $2n$. Then, probability that r is a root is at most $2n/|\mathbb{F}| < 1/n^2$, for large enough choice of $|\mathbb{F}|$. Taking a union bound over all $i \in [k]$, we get that the probability that r is a root of $\hat{p}_i - p_i$ for some i is at most $1/n$.

Help and Verification costs. Since degree of each p_i is $2n$, the total hcost is $\tilde{O}(kn)$. Verifier stores $\tilde{A}(v, r)$ for each $v \in [n]$, which requires $\tilde{O}(n)$ space. Additionally, to compute $N_{i+1}(v_s)$ for some $i \in [k]$, he needs only the set $N_i(v_s)$. Thus, we can store the $N_i(v_s)$ sets by reusing space repeatedly, and this requires $O(n)$ space. Hence, the total vcost of this protocol is $\tilde{O}(n)$. Therefore, we get a $[kn, n]$-scheme for checking the existence of a path of length at most k between vertices v_s and v_t.

▶ **Theorem 8.** *Given an n-vertex (directed or undirected) multigraph $G(V, E)$ and specified vertices $v_s, v_t \in V$, for any $k \leqslant n-1$, there is a $[kn, n]$-scheme for* ST-kPATH. *In particular, there is a semi-streaming scheme for* ST-kPATH *when k is polylogarithmic in n.*

Application to Shortest Path. Based on the scheme in Theorem 8, we have the following straightforward corollary.

▶ **Corollary 9.** *Given a (directed or undirected) multigraph $G(V, E)$ (with edge multiplicity polylogarithmic in n) and specified vertices $v_s, v_t \in V$, there is a $[kn, n]$-scheme for* ST-SHORTESTPATH, *where k is length of the shortest v_s, v_t-path.*

Proof. If there is no v_s, v_t-path, Prover sends the connected component C that v_s is in. The Verifier first checks that C is indeed connected ([10], Theorem 7.7). Next, he verifies that there is no edge going out from C by checking whether the set $C \times (V \setminus C)$ and the edge set E are disjoint (Fact 2). Both of these are $[n, n]$-schemes.

If there is a v_s, v_t-path, and the shortest such path H has length k, then Prover sends it to the Verifier, and he can check whether H is indeed a v_s, v_t-path and that $H \subseteq E$ using an $[n, n]$-scheme, as edge multiplicity is polylogarithmic in n (Fact 2). Parallelly, he uses a $[kn, n]$-scheme to verify that there is no v_s, v_t-path of length at most $k-1$ (Theorem 8). ◀

6.2 A Multi-Pass Scheme for Detecting Short Paths

In Section 5, we obtained a scheme for ST-3PATH of total cost $o(n)$ using two passes over the input. We investigate if the same is true for ST-kPATH (for $k > 3$) if we allow "a few" more passes. For constant k, we answer this in the affirmative as we generalize the scheme for ST-3PATH and obtain such a scheme for ST-kPATH with $\lceil k/2 \rceil$ passes.

As usual, A denotes the adjacency matrix of the graph G. Let L and F be the characteristic vectors of $N[v_s]$ and $N(v_t)$ respectively. Let $\kappa = \kappa(G)$ denote the number of walks of length

at most k from v_s to v_t in G. Then,

$$\kappa = \sum_{u_1,\ldots,u_{k-1}\in V} L_{u_1}\left(\prod_{i=1}^{k-2} A_{u_i,u_{i+1}}\right) F_{u_{k-1}}\,. \tag{24}$$

Note that there is a path of length at most k from v_s to v_t iff $\kappa > 0$. Therefore, computing κ suffices.

Let h and v be integer parameters with $hv = n$. Again, using a canonical bijection, we represent each vertex $u \in V$ by a pair of integers $(x, y) \in [h] \times [v]$. The vectors L and F become 2-dimensional arrays ℓ and f, given by $\ell(x, y) = L_u$ and $f(x, y) = F_u$. Again, the adjacency matrix A turns into a 4-dimensional array a, such that $a(x, y, x', y') = A_{uu'}$. Let $\tilde{\ell}$, $\tilde{f}$ and $\tilde{a}$ be $\mathbb{F}$-extensions of ℓ, f and a respectively, for a sufficiently large finite field $\mathbb{F}$. Thus, Equation (24) gives

$$\kappa = \sum_{x_1,\ldots,x_{k-1}\in[h]} p(x_1,\ldots,x_{k-1})\,, \quad \text{where} \tag{25}$$

$$p(X_1,\ldots,X_{k-1}) = \sum_{y_1,y_2\in[v]} \tilde{\ell}(X_1,y_1)\left(\prod_{i=1}^{k-2}\tilde{a}(X_i,y_i,X_{i+1},y_{i+1})\right)\tilde{f}(X_{k-1},y_{k-1})\,. \tag{26}$$

For $i \in [k-1]$, $\deg_{X_i} p = 2h-2$. Therefore, the number of monomials in p is at most $O(h^{k-1})$ and the total degree is $O(kh)$.

We present a $\lceil k/2\rceil$-pass protocol for ST-kPATH.

Stream processing. First, Verifier chooses $r_1,\ldots,r_{k-1} \in_R \mathbb{F}$.

Pass 1. Process only the vertices in $N_1[v_s]$ and $N_1(v_t)$ in the stream. We maintain, for each $y \in [v]$, two vectors of size v: $\tilde{\ell}(r_1, y)$ and $\tilde{f}(r_{k-1}, y)$, where $y \in [s]$.

Pass i, for $2 \leqslant i \leqslant \lceil k/2\rceil$. Define $g_0(y) := \tilde{\ell}(r_1, y)$ and $g_k(y) = \tilde{f}(r_{k-1}, y)$. For each $y \in [v]$, compute $g_{i-1}(y) := \sum_{y'\in[v]} \tilde{a}(r_{i-1}, y, r_i, y')g_{i-2}(y')$ as well as $g_{k-i+1}(y) := \sum_{y'\in[v]} \tilde{a}(r_{k-i}, y, r_{k-i+1}, y')g_{k-i+2}(y')$. The $g_j(y)$ values are updated dynamically with the stream updates in a similar way as in the protocol for CROSSEDGECOUNT in Section 5.1.

Help message. At the end of the final pass, Prover sends a polynomial $\hat{p}(X_1,\ldots,X_{k-1})$ (as a stream of coefficients) that she claims equals $p(X_1,\ldots,X_{k-1})$.

Verification and output. After the final pass, Verifier computes $\sum_{y\in[v]} g_{\lceil k/2\rceil}(y)g_{\lceil k/2\rceil+1}(y)$, which, by Equation (26), equals $p(r_1,\ldots,r_{k-1})$. If he finds that it doesn't equal $\hat{p}(r_1,\ldots,r_{k-1})$, he outputs $\perp$. Otherwise, he believes that $\hat{p} \equiv p$ and, following Equation (25), computes $\hat{\kappa} := \sum_{x_1,\ldots,x_{k-1}\in[h]} \hat{p}(x_1,\ldots,x_{k-1})$. He outputs YES if $\hat{\kappa} > 0$ and NO otherwise.

Error probability. We err only when $\hat{p} \not\equiv p$, but Verifier's check passes. In this case, $(r_1,\ldots,r_{k-1}) \in \mathbb{F}^{k-1}$ is a root of the nonzero polynomial $\hat{p} - p$. We noted that its total degree is at most $O(kh)$. By the Schwartz-Zippel Lemma (Fact 3), the probability of this event is at most $O(kh)/|\mathbb{F}| < 1/n$, when $|\mathbb{F}|$ is large enough.

Help and Verification costs. The number of monomials of $\hat{p}$ is $O(h^{k-1})$, giving an hcost of $\tilde{O}(h^{k-1})$. Verifier reuses space and, during each pass, stores $O(1)$ many v-dimensional vectors, each entry of which is $O(\log n)$ bits long. Thus, the vcost is $\tilde{O}(v)$.

This gives a $\lceil k/2\rceil$-pass $[h^{k-1}, v]$-scheme for ST-kPATH, for parameters h, v with $hv = n$. Setting $h = n^{1/k}$ and $v = n^{1-1/k}$, we get a scheme with total cost $\tilde{O}(n^{1-1/k})$.

▶ **Theorem 10.** *For any constant k, there is a $\lceil k/2\rceil$-pass $[n^{1-1/k}, n^{1-1/k}]$-scheme for* ST-kPATHCOUNT *in a (directed or undirected) graph.*

We note the contrast between this result and that of Guruswami and Onak [17]. They showed a lower bound of $\Omega(n^{1+\Omega(1/k)}/k^{O(1)})$ for ST-kPATH in $k/2-1$ passes in the basic (sans prover) streaming model (for even k). Our results show that using $\lceil k/2 \rceil$ passes, we can obtain a scheme for the same problem with total cost of $\tilde{O}(n^{1-1/k})$.

References

1. Scott Aaronson and Avi Wigderson. Algebrization: A New Barrier in Complexity Theory. In *Proc. 40th Annual ACM Symposium on the Theory of Computing*, pages 731–740, 2008.
2. Amirali Abdullah, Samira Daruki, Chitradeep Dutta Roy, and Suresh Venkatasubramanian. Streaming Verification of Graph Properties. In *Proc. 27th International Symposium on Algorithms and Computation*, pages 3:1–3:14, 2016.
3. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *J. ACM*, 45(3):501–555, 1998. Preliminary version in *Proc. 33rd Annual IEEE Symposium on Foundations of Computer Science*, pages 14–23, 1992.
4. Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *J. ACM*, 45(1):70–122, 1998. Preliminary version in *Proc. 33rd Annual IEEE Symposium on Foundations of Computer Science*, pages 2–13, 1992.
5. Sepehr Assadi, Sanjeev Khanna, and Yang Li. On Estimating Maximum Matching Size in Graph Streams. In *Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1723–1742, 2017.
6. Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. Reductions in Streaming Algorithms, with an Application to Counting Triangles in Graphs. In *Proc. 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 623–632, 2002.
7. Suman K. Bera and Amit Chakrabarti. Towards Tighter Space Bounds for Counting Triangles and Other Substructures in Graph Streams. In *34th Symposium on Theoretical Aspects of Computer Science (STACS 2017)*, pages 11:1–11:14, 2017.
8. J.A. Bondy and U.S.R Murty. *Graph Theory*. Springer Publishing Company, Incorporated, 1st edition, 2008.
9. Amit Chakrabarti, Graham Cormode, Navin Goyal, and Justin Thaler. Annotations for Sparse Data Streams. In *Proc. 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 687–706, 2014.
10. Amit Chakrabarti, Graham Cormode, Andrew McGregor, and Justin Thaler. Annotations in Data Streams. *ACM Trans. Alg.*, 11(1):Article 7, 2014.
11. Amit Chakrabarti, Graham Cormode, Andrew McGregor, Justin Thaler, and Suresh Venkatasubramanian. Verifiable Stream Computation and Arthur-Merlin Communication. In *Proc. 30th Annual IEEE Conference on Computational Complexity*, pages 217–243, 2015.
12. Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: matchings, matroids, and more. *Math. Program.*, 154(1–2):225–247, 2015. Preliminary version in *Proc. 17th Conference on Integer Programming and Combinatorial Optimization*, pages 210–221, 2014.
13. Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Streaming Graph Computations with a Helpful Advisor. *Algorithmica*, 65(2):409–442, 2013.
14. Graham Cormode, Justin Thaler, and Ke Yi. Verifying Computations with Streaming Interactive Proofs. *Proc. VLDB Endowment*, 5(1):25–36, 2011.
15. Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. Graph Distances in the Data-Stream Model. *SIAM J. Comput.*, 38(6):1709–1727, 2008. Preliminary version in *Proc. 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 745–754, 2005.
16. Ashish Goel, Michael Kapralov, and Sanjeev Khanna. On the communication and streaming complexity of maximum bipartite matching. In *Proc. 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 468–485, 2012.
17. Venkatesan Guruswami and Krzysztof Onak. Superlinear Lower Bounds for Multipass Graph Processing. *Algorithmica*, 76(3):654–683, November 2016.
18. Madhav Jha, C. Seshadhri, and Ali Pinar. A space efficient streaming algorithm for triangle counting using the birthday paradox. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013.

19 John Kallaugher, Andrew McGregor, Eric Price, and Sofya Vorotnikova. The Complexity of Counting Cycles in the Adjacency List Streaming Model. In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 119–133, 2019.

20 Daniel M. Kane, Kurt Mehlhorn, Thomas Sauerwald, and He Sun. Counting Arbitrary Subgraphs in Data Streams. In *Automata, Languages, and Programming - 39th International Colloquium, Proceedings, Part II*, pages 598–609, 2012.

21 Michael Kapralov. Better bounds for matchings in the streaming model. In *Proc. 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1679–1697, 2013.

22 Hartmut Klauck and Ved Prakash. Streaming computations with a loquacious prover. In *Proc. 4th Conference on Innovations in Theoretical Computer Science*, pages 305–320, 2013.

23 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic Methods for Interactive Proof Systems. *J. ACM*, 39(4):859–868, 1992.

24 Andrew McGregor. Finding Graph Matchings in Data Streams. In *Proc. 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 170–181, 2005.

25 Andrew McGregor, Sofya Vorotnikova, and Hoa T. Vu. Better Algorithms for Counting Triangles in Data Streams. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '16, pages 401–411, 2016.

26 Michael Mitzenmacher and Justin Thaler. Technical Perspective: Catching lies (and mistakes) in offloaded computation. *Commun. ACM*, 59(2):102, 2016.

27 Alexander Razborov. On the Distributional Complexity of Disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992. Preliminary version in *Proc. 17th International Colloquium on Automata, Languages and Programming*, pages 249–253, 1990.

28 Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992.

29 Justin Thaler. Semi-Streaming Algorithms for Annotated Graph Streams. In *Proc. 43rd International Colloquium on Automata, Languages and Programming*, pages 59:1–59:14, 2016.

30 Peter A. Tucker, David Maier, Lois M. L. Delcambre, Tim Sheard, Jennifer Widom, and Mark P. Jones. Punctuated Data Streams, 2005.

31 Ke Yi, Feifei Li, Marios Hadjieleftheriou, George Kollios, and Divesh Srivastava. Randomized Synopses for Query Assurance on Data Streams. In *International Conference on Data Engineering*, 2008.

A Missing Details from Section 5

Here, we give the missing proofs of lower bounds for CrossEdgeCount, IndSetTest and st-3Path. Then, we discuss applications of CrossEdgeCount to some standard graph problems. Finally, we show how our two-pass schemes can be made single-pass under certain assumptions on the stream order.

A.1 One-Pass Lower Bounds

We quickly review some relevant material from communication complexity. In the INDEX_N problem, there are two players: Alice, who holds a vector $\mathbf{x} \in \{0,1\}^N$, and Bob, who holds an index $k \in [N]$. Their goal is to output the bit $\mathbf{x}_k$. To prove lower bounds for one-pass *schemes*, we consider the *Online Merlin–Arthur* (OMA) communication model.[4] Here, in addition to Alice and Bob, there is a super-player, Merlin, who knows both their inputs, but is not to be blindly trusted. Merlin sends a message to Bob; then Alice sends a randomized

[4] Note that our semantics are slightly different from the usual definition of Merlin–Arthur where Bob is supposed "accept" each 1-input and reject each 0-input with probability at least 2/3.

message to Bob; finally, Bob either outputs either a bit or $\perp$. If Merlin is honest, Bob should output $\mathbf{x}_k$ with probability at least 2/3; if he is dishonest, Bob should output $\perp$ with probability at least 2/3.

The cost of an OMA protocol is the total number of bits communicated to Bob. The OMA complexity of a communication game is the minimum cost of a correct OMA protocol for it. Chakrabarti et al. [10, Theorem 3.1] showed that the OMA Complexity of INDEX_N is $\Omega(\sqrt{N})$. Our lower bounds follow from this result, using simple reductions from INDEX_N to the various graph problems.

Using a canonical bijection from $[n]^2$ to $[N]$, Alice rewrites her input vector $\mathbf{x} \in \{0,1\}^N$ as a matrix $(\mathbf{x}_{ij})_{i,j\in[n]}$, while Bob looks at his input index $k \in [N]$ as $(y,z) \in [n]^2$. Our reduction creates a graph $G = (V,E)$ on $2n$ vertices: the vertex set V is $L \uplus R$ (here, $\uplus$ denotes disjoint union), where $|L| = |R| = n$. We denote the ith vertex of L (resp. R) by ℓ_i (resp. r_i). The edge set E is given by $\{(\ell_i, r_j) : \mathbf{x}_{ij} = 1\}$. Now, by checking if (ℓ_y, r_z) is an independent set in G, or whether there's a cross-edge between the sets $\{\ell_y\}$ and $\{r_z\}$, or solving ST-3PATH in the graph $G' = (V \cup \{v_s, v_t\}, E \cup \{(v_s, \ell_y), (r_z, v_t)\})$, Bob can solve the INDEX_N problem. Thus, a one-pass scheme that solves any of these problems must have a total cost of $\Omega(n)$. We remark that Fact 2 implies matching semi-streaming upper bounds for each of them.

A.2 Applications of CrossEdgeCount

As we noted earlier, a scheme for CROSSEDGECOUNT can be used as a blackbox for solving a number of other problems. These include standard problems like INDSETTEST and ST-3PATH, as well as their generalizations or variations like the following problems.

- INDUCEDEDGECOUNT: Given a graph $G = (V,E)$ and a subset U of V, find the number of edges in G that are induced by U.
- ROOTEDTRIANGLECOUNT: Given a (directed or undirected) graph $G = (V,E)$ and a vertex $v_r \in V$, find the number of triangles in G that are rooted at v_r.

▶ **Corollary 11.** *Let t and s be parameters such that $ts = n$. Then each of the problems* INDUCEDEDGECOUNT, INDSETTEST, ST-3PATH, *and* ROOTEDTRIANGLECOUNT *admits a two-pass $[t^2, s]$-scheme; in particular, a two-pass scheme with total cost $\tilde{O}(n^{2/3})$.*

Proof. For INDUCEDEDGECOUNT, if the input graph is undirected, then considering U and W as the same set, solve CROSSEDGECOUNT-UNIQ. (Alternatively, solve CROSSEDGECOUNT and divide the answer by two.) If the graph is directed, then solve CROSSEDGECOUNT.

For INDSETTEST, solve INDUCEDEDGECOUNT on U and check whether the answer equals zero.

For ST-3PATH, use a scheme for CROSSEDGECOUNT to find the number of cross-edges between the closed neighborhoods $N[v_s]$ and $N[v_t]$ of vertices v_s and v_t. This actually solves the more general problem of counting the number of walks of length at most 3 from v_s to v_t. Checking whether this number is non-zero decides ST-3PATH.

Finally, for ROOTEDTRIANGLECOUNT, if the input graph is undirected, solve INDUCEDEDGECOUNT on $N(v_r)$. Otherwise, solve CROSSEDGECOUNT on the out-neighborhood $N^+(v_r)$ and in-neighborhood $N^-(v_r)$ of v_r. ◀

A.3 One-Pass Schemes for Certain Stream Orderings

Our two-pass solution to the CrossEdgeCount problem, as well as its corollaries, allowed the vertices and edge updates to be arbitrarily intermixed in the input stream. That said, it is interesting to focus on a natural restriction of these problems where the vertices are streamed first, followed by the edge updates. For the st-3Path problem, the corresponding restriction is that the edges incident to v_s and v_t appear before any other edges in the stream; for RootedTriangleCount, it is that the edges incident to v_r appear first.

Under such a restriction on the stream ordering, our two-pass solutions natural become one-pass, as we now note.

▶ **Corollary 12.** *The schemes for* CrossEdgeCount *and* CrossEdgeCount-Uniq *in Theorem 7 and for* InducedEdgeCount, IndSetTest, st-3Path, *and* RootedTriangleCount *in Corollary 11 can each be implemented in one pass under a restricted stream ordering as noted above.*

Proof. Consider the protocol described in Section 5.1. Note that the first pass processes only vertices and the second pass processes only edges. This implies the claimed results for CrossEdgeCount, CrossEdgeCount-Uniq, InducedEdgeCount, and IndSetTest. For st-3Path, note that requiring edges incident to v_s and v_t to arrive first is equivalent to the vertex sets $N(v_s)$ and $N(v_t)$ arriving first. A similar consideration applies to RootedTriangleCount. ◀

It is important to note that despite the restriction on the stream ordering, the schemes in Corollary 12 are nontrivial. Without Prover's help, the problems remain hard, even with multiple passes. We give the simple proof for the basic problem CrossEdgeCount.

▶ **Proposition 13.** *Any p-pass streaming algorithm for* CrossEdgeCount, *with vertices streamed before edges, requires* $\Omega(n/p)$ *space, even for insertion-only streams.*

Proof. We reduce from DISJ_n, the set-disjointness communication problem on the universe $[n]$. Recall that, in DISJ_n, Alice holds a set $x \subseteq [n]$ and Bob holds a set $y \subseteq [n]$. Their goal is to determine whether or not $x \cap y = \varnothing$. This problem has randomized communication complexity $\text{R}(\text{DISJ}_n) = \Omega(n)$ [27].

Consider an $(n+1)$-vertex graph G where $V(G) = \{0, \ldots, n\}$ and $E(G) = \{\{0, i\} : i \in y\}$. Let $U = \{0\}$ and $W = x$. Then the number of cross edges in G from U to W is non-zero iff $x \cap y \neq \varnothing$. The result now follows along standard lines. ◀

Approximate Degree, Secret Sharing, and Concentration Phenomena

Andrej Bogdanov
Department of Computer Science and Engineering, Chinese University of Hong Kong
Institute for Theoretical Computer Science and Communications, Hong Kong
andrejb@cse.cuhk.edu.hk

Nikhil S. Mande
Department of Computer Science, Georgetown University, USA
nikhil.mande@georgetown.edu

Justin Thaler
Department of Computer Science, Georgetown University, USA
justin.thaler@georgetown.edu

Christopher Williamson
Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong
chris@cse.cuhk.edu.hk

Abstract

The ε-approximate degree $\widetilde{\deg}_\varepsilon(f)$ of a Boolean function f is the least degree of a real-valued polynomial that approximates f pointwise to within ε. A sound and complete certificate for approximate degree being at least k is a pair of probability distributions, also known as a *dual polynomial*, that are perfectly k-wise indistinguishable, but are distinguishable by f with advantage $1-\varepsilon$. Our contributions are:

- We give a simple, explicit new construction of a dual polynomial for the AND function on n bits, certifying that its ε-approximate degree is $\Omega\left(\sqrt{n\log 1/\varepsilon}\right)$. This construction is the first to extend to the notion of weighted degree, and yields the first explicit certificate that the 1/3-approximate degree of any (possibly unbalanced) read-once DNF is $\Omega(\sqrt{n})$. It draws a novel connection between the approximate degree of AND and anti-concentration of the Binomial distribution.
- We show that any pair of *symmetric* distributions on n-bit strings that are perfectly k-wise indistinguishable are also statistically K-wise indistinguishable with at most $K^{3/2}\cdot\exp\left(-\Omega\left(k^2/K\right)\right)$ error for all $k<K\le n/64$. This bound is essentially tight, and implies that any symmetric function f is a reconstruction function with constant advantage for a ramp secret sharing scheme that is secure against size-K coalitions with statistical error $K^{3/2}\cdot\exp\left(-\Omega\left(\widetilde{\deg}_{1/3}(f)^2/K\right)\right)$ for all values of K up to $n/64$ simultaneously. Previous secret sharing schemes required that K be determined in advance, and only worked for $f=\mathsf{AND}$. Our analysis draws another new connection between approximate degree and concentration phenomena.

  As a corollary of this result, we show that for any $d\le n/64$, any degree d polynomial approximating a symmetric function f to error 1/3 must have coefficients of ℓ_1-norm at least $K^{-3/2}\cdot\exp\left(\Omega\left(\widetilde{\deg}_{1/3}(f)^2/d\right)\right)$. We also show this bound is essentially tight for any $d>\widetilde{\deg}_{1/3}(f)$. These upper and lower bounds were also previously only known in the case $f=\mathsf{AND}$.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases approximate degree, dual polynomial, pseudorandomness, polynomial approximation, secret sharing

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.71



© Andrej Bogdanov, Nikhil S. Mande, Justin Thaler, and Christopher Williamson;
licensed under Creative Commons License CC-BY

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh; Article No. 71; pp. 71:1–71:21



Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Category RANDOM

Related Version A full version of the paper is available at `https://eccc.weizmann.ac.il/report/2019/082/`.

Funding *Andrej Bogdanov*: Supported by Hong Kong RGC GRF CUHK14207618.
Nikhil S. Mande: Supported by NSF Grant CCF-1845125.
Justin Thaler: Supported by NSF Grant CCF-1845125.
Christopher Williamson: Supported by the Hong Kong PhD Fellowship Scheme.

Acknowledgements We thank Mark Bun for telling us about the work of Sachdeva and Vishnoi [22], and Mert Sağlam, Pritish Kamath, Robin Kothari, and Prashant Nalini Vasudevan for helpful comments on a previous version of the manuscript. We are also grateful to Xuangui Huang and Emanuele Viola for sharing the manuscript [14].

1 Introduction

The ε-approximate degree of a function $f\colon \{-1,1\}^n \to \{0,1\}$, denoted $\widetilde{\deg}_\varepsilon(f)$, is the least degree of a multivariate real-valued polynomial p such that $|p(x) - f(x)| \leq \varepsilon$ for all inputs $x \in \{-1,1\}^n$.[1] Such a p is said to be an approximating polynomial for f. This is a central object of study in computational complexity, owing to its polynomial equivalence to many other complexity measures including sensitivity, exact degree, deterministic and randomized query complexity [20], and quantum query complexity [6].

By linear programming duality, f has ε-approximate degree more than k if and only if there exist a pair of probability distributions μ and ν over the domain of f such that μ and ν are perfectly k-wise indistinguishable (i.e., all k-wise projections of μ and ν are identical), but are $(1-\varepsilon)$-distinguishable by f, namely $\mathbb{E}_{X\sim\mu}[f(X)] - \mathbb{E}_{Y\sim\nu}[f(Y)] \geq 1-\varepsilon$. Said equivalently, a sound and complete certificate for ε-approximate degree being more than k is a *dual polynomial* $q = (\mu - \nu)/2$ that contains no monomials of degree k or less, and such that $\sum_x |q(x)| = 1$ and $\sum_x q(x)f(x) \geq \varepsilon$.

Dual polynomials have immediate applications to cryptographic secret sharing: a dual polynomial $q = (\mu - \nu)/2$ for f is a description of a cryptographic scheme for sharing a 1-bit secret amongst n parties, where the secret can be reconstructed by applying f to the shares, and the scheme is secure against coalitions of size k (see [4] for details).

Motivation for explicit constructions of dual polynomials. Recent years have seen significant progress in proving new approximate degree lower bounds by explicitly constructing dual polynomials exhibiting the lower bound [8, 24, 9, 25, 10, 7, 11, 27]. These new lower bounds have in turn resolved significant open questions in quantum query complexity and communication complexity. At the technical core of these results are techniques for constructing a dual polynomial for composed functions $f \circ g := f(g, \ldots, g)$, given dual polynomials for f and g individually.

Often, an explicitly constructed dual polynomial showing that $\widetilde{\deg}_\varepsilon(g) \geq d$ exhibits additional metric properties, beyond what is required simply to witness $\deg_\varepsilon(g) \geq d$. Much of the major recent progress in proving approximate degree lower bounds has exploited these additional metric properties [10, 7, 11, 27]. Accordingly, even if cases where an approximate degree lower bound for a function g is known, it can often be useful to construct an explicit dual polynomial witnessing the lower bound. Hence, we are optimistic that the new constructions of dual polynomials given in this work will find future applications.

[1] In this work, for convenience we also consider functions mapping $\{0,1\}^n$ to $\{0,1\}$.

Explicit constructions of dual polynomials are also necessary to implement the corresponding secret-sharing scheme, and to analyze the complexity of the algorithm that samples the shares of the secret.

Our results in a nutshell. Our results fall into two categories. In the first category, we reprove several known approximate degree lower bounds by giving the first explicit constructions of dual polynomials witnessing the lower bounds. Specifically, our dual polynomial certifies that the ε-approximate degree of the n-bit AND function is $\Theta(\sqrt{n \log 1/\varepsilon})$. This construction is the first to extend to the notion of weighted degree, and yields the first explicit certificate that the 1/3-approximate degree of any (possibly unbalanced) read-once DNF is $\Omega(\sqrt{n})$. Interestingly, our dual polynomial construction draws a novel and clean connection between the approximate degree of AND and anti-concentration of the Binomial distribution.

In the second category, we prove new and tight results about the size of the coefficients of polynomials that approximate symmetric functions. Specifically, we show that for any $d \leq n/64$, any degree d polynomial approximating f to error 1/3 must have coefficients of weight (ℓ_1-norm) at least $d^{3/2} \cdot \exp\left(\Omega\left(\widetilde{\deg}_{1/3}(f)^2/d\right)\right)$. We show this bound is tight (up to logarithmic factors in the exponent) for any $d > \widetilde{\deg}_{1/3}(f)$. These bounds were previously only known in the case $f = \mathsf{AND}$ [23, 5]. Our analysis actually establishes a considerably more general result, and as a consequence we obtain new cryptographic secret sharing schemes with symmetric reconstruction procedures (see Section 1.2 for details).

1.1 A New Dual Polynomial for AND

To describe our dual polynomial for AND, it will be convenient to consider the AND function to have domain $\{-1,1\}^n$ and range $\{0,1\}$, with $\mathsf{AND}(x) = 1$ if and only if $x = 1^n$. In their seminal work, Nisan and Szegedy [20] proved that the 1/3-approximate degree of the AND function on n inputs is $\Theta(\sqrt{n})$. More generally, it is now well-known that the ε-approximate degree of AND is $\Theta\left(\sqrt{n \log(1/\varepsilon)}\right)$ [15, 6]. These works do not construct explicit dual polynomials witnessing the lower bounds; this was achieved later in works of Špalek [28] and Bun and Thaler [8].

Our first contribution is the construction of a new dual polynomial ϕ for AND, which is simple enough to describe in a single equation:

$$\phi(x) = \frac{(-1)^n}{Z}\left(\prod_{i\in[n]} x_i\right)\left(\mathbb{E}_S \prod_{i\in S} x_i\right)^2. \tag{1}$$

Here, S is a random subset of $\{1,\ldots,n\}$ of size at most $\frac{1}{2}(n-d)$ (where d determines the degree of the polynomials against which the exhibited lower bound holds), and Z is an (explicit) normalization constant.

In the language of secret sharing, to share a secret $s \in \{-1,1\}$, the dealer samples shares $x \in \{-1,1\}^n$ with probability proportional to $(\mathbb{E}_S \prod_{i\in S} x_i)^2$, conditioned on the parity of the shares $\prod x_i$ being equal to s.

In Corollary 8 we show that ϕ certifies that every degree-d polynomial must differ from the AND function by $2^{-n}\sum_{k=0}^{(n-d)/2}\binom{n}{k}$ at some input. In other words, the approximation error of a degree-d polynomial is lower bounded by the probability that a sum of unbiased independent bits deviates from its mean by $d/2$.

Our function ϕ given in (1), unlike previous dual polynomials [15, 28, 9, 26], also certifies that the *weighted* 1/3-approximate degree of AND with weights $w \in \mathbb{R}^n_{\geq 0}$ is $\Omega(\|w\|_2)$ (see Corollary 9).[2] This lower bound is tight for all w, matching an upper bound of Ambainis [1]. The only difference in our dual polynomial construction for the weighted case is in the distribution over sets S, and the lower bound in the weighted case is derived from anti-concentration of *weighted* sums of Bernoulli random variables.

Both statements are corollaries of the following theorem.

▶ **Theorem 1.** *Define* $\mathsf{AND}\colon \{-1,1\}^n \to \{0,1\}$ *as* $\mathsf{AND}(x) = 1$ *if and only if* $x = 1^n$. *The function* ϕ *defined in Equation* (1) *is a dual witness for* $\widetilde{\deg}_{w,\varepsilon}(\mathsf{AND}) \geq d$ *for* $\varepsilon = \Pr_{X \sim \{-1,1\}^n}[\langle w, X \rangle \geq d]$.

By combining, in a black-box manner, the dual polynomial for the weighted-approximate degree of AND with prior work (e.g., [16, Proof of Theorem 7]), one obtains, for any read-once DNF f, an explicit dual polynomial for the fact that $\widetilde{\deg}_{1/3}(f) \geq \Omega(n^{1/2})$. Very recent work of Ben-David et al. [2] established this result for the first time, shaving logarithmic factors off of prior work [9, 16]. In fact, Ben-David et al. [2] prove more generally that any depth-d read-once AND-OR formula has approximate degree $2^{-O(d)}\sqrt{n}$. Their method, however, does not appear to yield an explicit dual polynomial, even in the case $d = 2$.

Discussion. It has been well known that the ε-approximate degree of the AND function on n variables is $\Theta\left(\sqrt{n \log(1/\varepsilon)}\right)$ [20, 6], a fact which has many applications in theoretical computer science. This is superficially reminiscent of Chernoff bounds, which state that the middle $\Theta\left(\sqrt{n \log(1/\varepsilon)}\right)$ layers of the Hamming cube contain a $1 - \varepsilon$ fraction of all inputs (i.e., "most" n-bit strings have Hamming weight close to $n/2$). However, these two phenomena have not previously been connected, and it is not a priori clear why approximate degree should be related to concentration of measure. An approximating polynomial p for f must approximate f at *all* inputs in $\{-1,1\}^n$. Why should it matter that *most* (but very far from all) inputs have Hamming weight close to $n/2$?

The new dual witness for AND constructed in Equation (1) above provides a surprising answer to this question. The connection between (anti-)concentration and approximate degree of AND arises not because of the number of *inputs* to f that have Hamming weight close to $n/2$, but because of the number of *parity functions* on n bits that have *degree* close to $n/2$. This connection appears to be rather deep, as evidenced by our construction's ability to yield a tight lower bound in the case of weighted approximate degree.

1.2 Indistinguishability for Symmetric Distributions

In this section, for convenience we consider functions mapping $\{0,1\}^n$ to $\{0,1\}$. Two distributions μ and ν over $\{0,1\}^n$ are *(statistically)* (k,δ)*-wise indistinguishable* if for all subsets $S \subseteq \{1,\ldots,n\}$ of size k, the induced marginal distributions $\mu|_S$ and $\nu|_S$ are within statistical distance δ. When $\delta = 0$, we say they are *(perfectly)* k*-wise indistinguishable*.

[2] For a polynomial $p(x_1,\ldots,x_n)$, a weight vector $w \in \mathbb{R}^n_{\geq 0}$ assigns weight w_i to variable x_i. The weighted degree of p is the maximum weight over all monomials appearing in p, where the weight of a monomial is the sum of the weights of the variables appearing within it. The weighted ε-approximate degree of f, denoted $\widetilde{\deg}_{w,\varepsilon}(f)$, is the least weighted degree of any polynomial that approximates f pointwise to error ε.

For general pairs of distributions, perfect k-wise indistinguishability does not imply any sort of security against distinguishers of size $k+1$. Any binary linear error-correcting code of distance $k+1$ and block length n induces a pair of distributions (the uniform distribution over codewords and one of its affine shifts) that are perfectly k-wise indistinguishable, yet perfectly $(k+1)$-wise distinguishable.

In contrast, we prove that perfect k-wise indistinguishability for *symmetric* distributions implies strong statistical security against larger adversaries:

▶ **Theorem 2.** *If μ and ν are symmetric over $\{0,1\}^n$ and perfectly k-wise indistinguishable, then they are statistically $(K, O(K^{3/2}) \cdot e^{-k^2/1156K})$-wise indistinguishable for all $1 \leq k < K \leq n/64$.*

Theorem 2 has the following direct consequence for secret sharing schemes over bits with symmetric reconstruction. We say (μ, ν) are α-reconstructible by f if $\mathbb{E}_{X\sim\mu}[f(X)] - \mathbb{E}_{Y\sim\nu}[f(Y)] \geq \alpha$.

▶ **Corollary 3.** *Let f be a symmetric Boolean function. There exists a pair of distributions μ and ν that are $\left(K, K^{3/2} \cdot e^{-\Omega(\widetilde{\deg}_{1/3}(f)^2/K)}\right)$-indistinguishable for all $K \leq n/64$, but are $\Omega(1)$-reconstructible by f.*

Corollary 3 is an immediate consequence of our Theorem 2, and the fact that any symmetric function has an optimal dual polynomial that is itself symmetric. In the special case $f = \mathsf{AND}$ (or equivalently $f = \mathsf{OR}$), Corollary 3 implies the existence of a *visual secret sharing scheme* (see, for example [19]) that is $\left(K, K^{3/2} \cdot e^{-\Omega(n/K)}\right)$-statistically secure against all coalitions of size K, simultaneously for all K up to size $n/64$. This property, where security guarantees are in place for many coalition sizes at the same time, is in contrast to an earlier result of Bogdanov and Williamson [5] where they proved that for any fixed coalition size K, there is a visual secret sharing scheme that is $(K, e^{-\Omega(n/K)})$-statistically secure. In their construction, the distribution of shares μ and ν depend on the value of K.

We remark that the bound of Corollary 3 cannot hold in general for $K = n$, since there exists distributions that are perfectly $\Omega(n)$-wise indistinguishable but are reconstructible by the majority function on all n inputs. We do not however know if a bound of the form $K \leq (1 - \Omega(1))n$ is tight in this context.

Tight weight-degree tradeoffs for polynomials approximating symmetric functions

Let $f\colon \{0,1\}^n \to \{0,1\}$ be any function. For any integer $d \geq 0$, denote by $W_\varepsilon(f,d)$ the minimum *weight* of any degree-d polynomial that approximates f pointwise to error ε. By the weight of a polynomial, we mean the ℓ_1-norm of its coefficients over the parity (Fourier) basis[3]. In Section B, we observe that Corollary 3 implies weight-degree trade-off lower bounds for symmetric functions.

▶ **Corollary 4.** *For any symmetric function $f\colon \{0,1\}^n \to \{0,1\}$, any constant $\varepsilon \in (0,1/2)$, and any integer K where $n/64 \geq K \geq \widetilde{\deg}_\varepsilon(f)$, we have $W_\varepsilon(f,K) \geq K^{-3/2} \cdot 2^{\Omega\left(\widetilde{\deg}_{1/3}(f)^2/K\right)}$.*

The following theorem shows that the lower bound obtained in Corollary 4 is tight (up to polylogarithmic factors in the exponent) for all symmetric functions.

[3] In fact, our main weight lower bound (Corollary 4) holds over any set of functions (not just parities) that each depend on at most d variables.

▶ **Theorem 5.** *For any symmetric function* $f\colon \{0,1\}^n \to \{0,1\}$*, any constant* $\varepsilon \in (0,1/2)$ *and* $K > \widetilde{\deg}_\varepsilon(f) \cdot \sqrt{\log n}$*,* $W_\varepsilon(f,K) \leq 2^{\tilde{O}(\widetilde{\deg}_{1/3}(f)^2/K)}$*.*[4]

Theorem 5 also implies that Corollary 3 is tight (up to polylogarithmic factors in the exponent) for all symmetric f and for all $K \geq \widetilde{\deg}_{1/3}(f)\sqrt{\log n}$. This is because any improvement to Corollary 3 would yield an improvement to Corollary 4, contradicting Theorem 5.

Essentially Optimal Ramp Visual Secret Sharing Schemes. The following result shows that in the case $f = \mathsf{AND}$, Corollary 3 is essentially tight for *all* $K \geq 2$, and Theorem 2 is tight as a reduction from perfect to approximate indistinguishability for symmetric distributions. It does so by constructing essentially optimal ramp visual secret sharing schemes.[5]

▶ **Theorem 6.** *For all* $2 \leq k < K \leq n$ *there exist symmetric* k*-wise indistinguishable distributions* μ *and* ν *over* n*-bit strings that are* $\sqrt{2^{-4K+3} \cdot \sum_{d>k} \binom{2K}{K+d}^2}$*-reconstructible by* AND_K*, where* $\mathsf{AND}_K(x)$ *is the* AND *of the first* K *bits of* x*.*

Discussion of Theorem 6. This theorem gives the existence of a ramp visual secret sharing scheme that is perfectly secure against any k parties, but in which any $K > k$ parties can reconstruct the secret with the above advantage. This generalizes the schemes in [5] where only reconstruction by all n parties was considered.

Let us express the reconstruction advantage appearing in Theorem 6 in a manner more easily comparable to other results in this manuscript. Standard results on anti-concentration of the Binomial distribution state that $2^{-2K} \cdot \sum_{d>k} \binom{2K}{K+d} = e^{-\Theta(k^2/K)}$ (see, e.g., [17]). The Cauchy-Schwarz inequality then implies that the reconstruction advantage appearing in Theorem 6 is at least $K^{-1/2} \cdot e^{-O(k^2/K)}$.[6]

Hence, the visual secret sharing schemes given in Theorem 6 are nearly optimal; if the reconstruction advantage could be improved by more than the leading $\mathrm{poly}(K)$ factor (or the constant factor in the exponent), then this would contradict Theorem 2 which upper bounds the distinguishing advantage of any statistical test over K bits against symmetric, perfectly k-wise indistinguishable distributions. Theorem 6 also shows that the indistinguishability parameter in Theorem 2 cannot be significantly improved, even in the restricted case where the only statistical test is AND_K.

In Section 4 we describe another application of Theorem 2 to security against share consolidation and "downward self-reducibility" of visual secret shares.

---

[4] Here and throughout, the $\tilde{O}$ notation hides polylogarithmic factors in n.

[5] A visual secret sharing scheme is a scheme where the reconstruction function is the AND of some subset of the shares. A ramp scheme is one where there is not necessarily a sharp threshold between the perfect secrecy and reconstruction thresholds; in particular, we allow for $K > k+1$.

[6] Theorem 6 is closely related to Theorem 1, in that Theorem 6 gives *another* anti-concentration-based proof that $\widetilde{\deg}_\varepsilon(\mathsf{AND}_K) \geq k$ for $\varepsilon = K^{-1/2} \cdot e^{-\Theta(k^2/K)}$. However, the two results are incomparable. Theorem 6 does not yield an explicit dual polynomial for AND_K, and the ε-approximate degree lower bound for AND_K implied by Theorem 6 is loose by the $K^{-1/2}$ factor appearing in the expression for ε. On the other hand, Theorem 1 only yields a visual secret sharing scheme with reconstruction by all n parties, while Theorem 6 yields a ramp scheme with non-trivial reconstruction advantage by the AND of the first K (out of n) parties.

1.3 Related Works

Prior Work. Servedio, Tan, and Thaler [23] established Corollary 4 and Theorem 5 in the special case $f = \mathsf{OR}$, showing that degree d polynomials that approximate the OR function require weight $2^{\tilde{\Theta}(n/d)} = 2^{\tilde{\Theta}(\widetilde{\deg}_{1/3}(\mathsf{OR})^2/d)}$.[7] They used this result to establish tight weight-degree tradeoffs for polynomial threshold functions computing decision lists. As previously mentioned, Bogdanov and Willamson [5] generalized the weight-vs-degree lower bound from [23] beyond polynomials, thereby obtaining a visual secret-sharing scheme for any fixed K that is $(K, e^{-\Omega(n/K)})$-statistically secure.

Elkies [13] and Sachdeva and Vishnoi [22] exploit concentration of measure to prove a tight upper bound on the degree of univariate polynomials that approximate the function $t \mapsto t^n$ over the domain $[-1, 1]$. Their techniques inspired our (much more technical) proof of Theorem 2.

Other Related Work. This work subsumes Bogdanov's manuscript [3], which shows a slightly weaker lower bound on the weighted approximate degree of AND, and does not derive an explicit dual polynomial. In independent work, Huang and Viola [14] prove a weaker form of our Corollary 3: their distributions μ, ν depend on the value of K. They also prove (a slightly tighter version of) Theorem 5, thereby establishing that the statistical distance in Corollary 3 is tight.

1.4 Techniques and Organization

The proof of Theorem 1 (Section 2) is an elementary verification that the function ϕ given in (1) is a dual polynomial. The only property that is not immediate is correlation with AND. Verifying this property amounts to upper bounding the normalization constant Z, which follows from orthogonality of the Fourier characters.

In the proof of Theorem 2 (Section 3), a K-bit statistical distinguisher for symmetric distribution is first decomposed into a sum of at most $K + 1$ tests Q_w that evaluate to 1 only when the input has Hamming weight exactly w. Lemma 13 shows that the univariate symmetrizations p_w of these distinguishers can be pointwise approximated by a degree-k polynomial with error at most $O(K^{1/2}) \cdot e^{-\Omega(k^2/K)}$.

To construct the desired approximation, we derive an identity relating the moment generating function of the squared Chebyshev coefficients of p_w (interpreted as relative probabilities) to the average magnitude of a polynomial g related to p_w on the unit complex circle (Claims 16 and 17). We bound these magnitudes analytically (Claim 18) and derive tail inequalities for the Chebyshev coefficients from bounds on the moment generating function as in standard proofs of Chernoff-Hoeffding bounds.

In the special case when the secrecy parameters k and K are fixed and the number of parties n approaches infinity, $p_w(t)$ turns out to equal $C_w(t-1)^w(t+1)^{K-w}$, where C_w is some quantity independent of t. In this case, the Chebyshev coefficients are the regular coefficients of the polynomial $g^\infty(s) = 2^{-w}C_w(s-1)^{2w}(s+1)^{2(K-w)}$.[8] When $w = 0$, $K/2$, or 1, the coefficients of g^∞ are exponentially concentrated around the middle as they follow

[7] These bounds for OR were implicit in [23], but not explicitly highlighted. The upper bound was explicitly stated in [12, Lemma 4.1], which gave applications to differential privacy, and the lower bound in [9, Lemma 32], which used it to establish tight weight-degree tradeoffs for polynomial threshold functions computing read-once DNFs.

[8] The i-th coefficient of g^∞ is the value of the i-th Kravchuk polynomial with parameter $2K$ evaluated at $2w$.

the binomial distribution. We prove that this exponential decay in magnitudes happens for all values of w, which requires understanding complicated cancellations in the algebraic expansion of $g^{\infty}(s)$.

We generalize this analysis to the finitary setting $n \geq 64K$.

We prove Theorem 5 (Section B) by writing any symmetric function f as a sum of at most $\ell := \min\{|f^{-1}(0)|, |f^{-1}(1)|\}$ many conjunctions, and approximating each conjunction to such low error (namely error $\ll \ell$) that the sum of all approximations is an approximation for f itself. Theorem 5 then follows by constructing low-weight, low-degree polynomial approximations for each conjunction in the sum.

Theorem 6 (Section C) is proved by lower bounding the error of degree k polynomial approximations to the symmetrization f of the function $\mathsf{AND}_K\left(x|_{\{1,\ldots,K\}}\right)$. By duality, a lower bound on approximation error translates into a secret sharing scheme with the same reconstruction advantage. To lower bound the error, we estimate the values of the coefficients in the Chebyshev expansion of f with indices larger than k. Owing to orthogonality, the largest of these coefficients lower bounds the approximation error of any degree-k polynomial.

In Section 4 we formulate a security of secret sharing against consolidation and downward self-reducibility of visual schemes, and derive these properties from the main results.

2 Dual Polynomial For the Weighted Approximate Degree of AND

In this section we prove Theorem 1 and derive its two corollaries about the unweighted and weighted approximate degree of AND.

Notation and Definitions. Let $[n] = \{1, \ldots, n\}$. Given a vector $w \in \mathbb{R}^n_{\geq 0}$, define the weight of a monomial $\chi_S(x) = \prod_{i \in S} x_i, x_i \in \{-1, 1\}$ to equal $\sum_{i \in S} w_i$. Define the w-weighted degree of a polynomial to be the maximum weight of a monomial in it. That is, if $p = \sum_{S \subseteq [n]} c_S \chi_S$, then define

$$\deg_w(p) = \max_{S : c_S \neq 0} w(S).$$

Define the w-weighted ε-approximate degree $\widetilde{\deg}_{w,\varepsilon}(f)$ to be the minimum w-weighted degree of a polynomial p that satisfies $|p(x) - f(x)| \leq \varepsilon$ for all x in the domain of f. Given two real-valued functions f, g over domain $\{-1, 1\}^n$, define $\langle f, g \rangle := \frac{1}{2^n} \sum_{x \in \{-1,1\}^n} f(x) \cdot g(x)$.

▶ **Lemma 7.** *For any finite set X and any function $f \colon X \to \mathbb{R}$, $\widetilde{\deg}_{w,\varepsilon}(f) \geq d$ iff there exists a function $\phi : X \to \mathbb{R}$ satisfying the following conditions.*

- Pure high degree*: For any real polynomial p of weighted degree is at most d, $\langle \phi, p \rangle = 0$.*
- Normalization*: $\sum_{x \in X} |\phi(x)| = 1$,*
- Correlation*: $\langle \phi, f \rangle \geq \varepsilon$,*

We call ϕ a dual witness for $\widetilde{\deg}_{w,\varepsilon}(f) \geq d$. The lemma follows by linear programming duality and is a straightforward generalization of previous results (see e.g. [28, 9]). We prove the "if" direction, which is sufficient for our purposes.

Proof. For any p of weighted degree at most d,

$$\|f - p\|_\infty = \|f - p\|_\infty \|\phi\|_1 \geq \langle \phi, f - p \rangle = \langle \phi, f \rangle - \langle \phi, p \rangle \geq \varepsilon.$$ ◀

The dual polynomial of interest is

$$\phi(x) = \frac{(-1)^n}{Z}\chi_{[n]}(x) \cdot \mathbb{E}_{S\sim\mathcal{H}}[\chi_S(x)]^2,$$

where $x \in \{-1,1\}^n$, $\mathcal{H}$ is the uniform distribution over the sets $\{S \subseteq [n] : w(S) \leq (\|w\|_1 - d)/2\}$, and Z is the normalization constant

$$Z = \sum_{x\in\{-1,1\}^n} \mathbb{E}_{S\sim\mathcal{H}}[\chi_S(x)]^2.$$

Proof of Theorem 1. We prove the theorem by showing that ϕ satisfies the three conditions of Lemma 7. The expression $\mathbb{E}_{S\sim\mathcal{H}}[\chi_S(x)]^2$ can be written as a sum of products of pairs of monomials of weight at most $(\|w\|_1 - d)/2$, so its weighted degree is at most $\|w\|_1 - d$. Thus every monomial that occurs in the expansion of $\chi_{[n]}(x)\,\mathbb{E}_{S\sim\mathcal{H}}[\chi_S(x)]^2$ must have weighted degree *at least* d, and so ϕ has pure high weighted degree at least d as desired.

The scaling by Z in the definition of ϕ ensures that ϕ has L_1 norm 1. The correlation of ϕ and AND is given by $\langle \phi, \mathsf{AND}\rangle = \phi(1^n) = \frac{1}{Z}$. Finally, the normalization constant Z evaluates to

$$\begin{aligned} Z &= \sum_{x\in\{-1,1\}^n} \mathbb{E}_{S\sim\mathcal{H}}[\chi_S(x)]^2 = \sum_{x\in\{-1,1\}^n} \mathbb{E}_{S\sim\mathcal{H}}[\chi_S(x)]\,\mathbb{E}_{T\sim\mathcal{H}}[\chi_T(x)] \\ &= \sum_{x\in\{-1,1\}^n} \mathbb{E}_{S,T\sim\mathcal{H}}[\chi_{S\Delta T}(x)] = \mathbb{E}_{S,T\sim\mathcal{H}} \sum_{x\in\{-1,1\}^n} \chi_{S\Delta T}(x) \\ &= 2^n \Pr[S = T] = \frac{2^n}{|\mathcal{H}|}, \end{aligned}$$

since the inner summation over x evaluates to 2^n when $S = T$, and zero otherwise.

It remains to show that $1/Z = |\mathcal{H}|/2^n$ equals the desired expression for ε. For a set $S \subseteq [n]$, let $X(S) \in \{-1,1\}^n$ be the string that assigns values 1 and -1 to elements inside and outside S, respectively. Then $w(S) = \|w\|_1/2 + \langle w, X(S)\rangle/2$, so

$$\frac{|\mathcal{H}|}{2^n} = \Pr_{S\subseteq[n]}[w(S) \geq \|w\|_1/2 + d/2] = \Pr_{X\sim\{-1,1\}^n}[\langle w, X\rangle \geq d]. \qquad \blacktriangleleft$$

▶ **Corollary 8** (Approximate degree of AND). *Recall that* $\mathsf{AND}\colon \{-1,1\}^n \to \{0,1\}$ *denotes the function satisfying* $\mathsf{AND}(x) = 1$ *if and only if* $x = 1^n$. *If* p *has degree at most* d, *then* $|p(x) - \mathsf{AND}(x)| \geq \Pr[X \leq (n-d)/2]$ *for some* x, *where* X *is a* Binomial$(n, 1/2)$ *random variable.*

The expression on the right is lower bounded by the larger of $1/2 - O(d/\sqrt{n})$ and $2^{-O(d^2/n)}$. In the large d regime ($d \geq \sqrt{n}$), this bound is tight [15, 6].

Proof. Apply Theorem 1 to the weight vector $w = (1, 1, \ldots, 1)$. ◀

Earlier constructions of dual polynomials for AND are quite different from our Corollary 8 [15, 28, 9, 26] and are based on real-valued polynomial interpolation. Specifically, for a carefully chosen set $T \subseteq \{0, 1, \ldots, n\}$ of size $|T| = 2d$, the prior constructions consider a *univariate* polynomial $p(t) = \prod_{i\in[n]\setminus T}(t - i)$, and they define $\psi(x) = p(|x|)$, where $|x|$ denotes the Hamming weight of x. Clearly ψ has degree at most $n - |T|$. A fairly complicated calculation is required to show that, for an appropriate choice of T, defining ψ in this way ensures that $|\psi(1^n)|$ captures an ε-fraction of the L_1-mass of ψ.

▶ **Corollary 9** (Weighted approximate degree of AND). $\widetilde{deg}_{w,3/32}(\mathsf{AND}) \geq \|w\|_2/2$.

The proof uses the Paley-Zygmund inequality:

▶ **Lemma 10** (Paley-Zygmund inequality). *Let $Z \geq 0$ be any random variable with finite variance. Then, for any $0 < \theta < 1$,*

$$\Pr[Z \geq \theta\,\mathbb{E}(Z)] \geq (1-\theta)^2 \frac{(\mathbb{E}[Z])^2}{\mathbb{E}[Z^2]}.$$

Proof of Corollary 9. We apply the Paley-Zygmund inequality to $\langle w, X\rangle^2$. First, $\mathbb{E}[\langle w, X\rangle]^2 = \|w\|_2^2$ and $\mathbb{E}[\langle w, X\rangle^4] = \sum w_i^4 + 3\sum w_i^2 w_j^2 \leq 3\|w\|_2^2$. Then

$$\Pr\left[\langle w, X\rangle \geq \frac{\|w\|_2}{2}\right] = \frac{1}{2}\Pr\left[|\langle w, X\rangle| \geq \frac{\|w\|_2}{2}\right] = \frac{1}{2}\Pr\left[\langle w, X\rangle^2 \geq \frac{\|w\|_2^2}{4}\right] \geq \frac{1}{2}\cdot\frac{9}{16}\cdot\frac{1}{3} = \frac{3}{32},$$

where the first equality follows from the sign-symmetry of X. Applying Theorem 1 with $d = \|w\|_2/2$ yields the claim. ◀

3 Approximate Indistinguishability from Perfect Indistinguishability

In this section, we prove Theorem 2, which states that any pair of symmetric and perfectly k-wise indistinguishable distributions over $\{0,1\}^n$ are also approximately indistinguishable against statistical tests that observe $K > k$ of the bits. We may and will assume without loss of generality that the statistical test is a symmetric function,[9] meaning that it depends only on the Hamming weight of the observed bits of its input.

Let X and Y denote an arbitrary pair of symmetric $(k,0)$-wise indistinguishable distributions over $\{0,1\}^n$. We will be interested in obtaining an upper bound on the statistical distance of their projections to any K indices of $[n]$, namely the advantage $\mathbb{E}_X[T(X|_S)] - \mathbb{E}_Y[T(Y|_S)]$ where $T : \{0,1\}^K \to \{0,1\}$ is a symmetric function and $S \subseteq [n]$ is any set of size K. We can decompose T into a sum of tests $Q_w : \{0,1\}^K \to \{0,1\}$, where Q_w outputs 1 if and only if the Hamming weight of its input is exactly w. Specifically, we decompose T as

$$T = \sum_{w=0}^{K} b_w Q_w, \tag{2}$$

where each b_w is either zero or one. We will bound the distinguishing advantage of each Q_w in the sum individually. This advantage is captured by a univariate function p_w that expresses Q_w in terms of the Hamming weight of its input, after shifting and scaling the Hamming weight to reside in the interval $[-1,1]$.

▶ **Fact 11.** *Let $S \subseteq [n]$ be any set of size K. There exists a univariate polynomial p_w of degree at most K such that the following holds. For all $t \in \{-1, -1+2/n, \ldots, 1-2/n, 1\}$, $p_w(t) = \mathbb{E}_Z[Q_w(Z|_S)]$ where Z is a random string of Hamming weight $\phi^{-1}(t) = (1-t)n/2 \in \{0,1,\ldots,n\}$.*

Proof. This statement is a simple extension of Minsky and Papert's classic symmetrization technique [18]. Specifically, Minsky and Papert showed that for any polynomial $p_n\colon \{0,1\}^n \to \mathbb{R}$, there exists a univariate polynomial P of degree at most the total degree of p_n, such that for all $i \in \{0,\ldots,n\}$, $P(i) = \mathbb{E}_{|x|=i}[p_n(x)]$. Apply this result to $p_n(x) = Q_w(x|_S)$ and let $p_w(t) = P(\phi^{-1}(t)) = P\left((1-t)n/2\right)$. The fact then follows from the observation that the total degree of $Q_w(x|_S)$ is at most K, since this function is a K-junta. ◀

[9] In the full version, we include simple proofs that (1) the marginal distributions of a symmetric distribution are symmetric and that (2) the best distinguisher between a pair of symmetric distributions is a symmetric function.

In particular, the value $p_w(t)$ is a probability for every $t \in \{-1, -1+2/n, \ldots, 1-2/n, 1\}$. Moreover, this probability must equal zero when the Hamming weight of Z is less than w or greater than $n-K+w$. Therefore p_w has K distinct zeros at the points $Z_w = Z_- \cup Z_+$, where

$$Z_- = \{-1+2h/n : h = 0, ..., K-w-1\}, \qquad Z_+ = \{1-2h/n : h = 0, ..., w-1\}. \tag{3}$$

and so p_w must have the form

$$p_w(t) = C_w \cdot \prod_{z \in Z_w} (t-z) \tag{4}$$

for some C_w that does not depend on t.[10] As $p_w(t)$ is probability when $t \in \{-1, -1+2/n, \ldots, 1-2/n, 1\}$, the function p_w is 1-bounded at those inputs. In fact, p_w is uniformly bounded on the interval $[-1, 1]$:

▷ Claim 12. Assuming $n \geq 64K$, $|p_w(t)| \leq 2$ for all $t \in [-1, 1]$.

The proof is omitted due to space limitations but follows a similar structure as the proof of Claim 18 which appears in Section A. Formula (4) and Claim 12 will be applied to show that p_w has a good uniform polynomial approximation on the interval $[-1, 1]$.

▶ **Lemma 13.** *Assuming $n \geq 64K$, there exists a degree-k polynomial q_w such that $|p_w(t) - q_w(t)| \leq 4\sqrt{K}\exp(-k^2/1156K)$ for all $t \in [-1, 1]$.*

Lemma 13 is the main technical result of this section. It is proved in Section 3.1.

Proof of Theorem 2. Now let T be a general distinguisher on K inputs, which we may and will assume to be a symmetric Boolean-valued function. We bound the distinguishing advantage as follows. Recalling that X and Y are $(k, 0)$-indistinguishable symmetric distributions over $\{0,1\}^n$, for any set $S \subseteq [n]$ of size K we have:

$$\begin{aligned}
&\mathbb{E}[T(X|_S)] - \mathbb{E}[T(Y|_S)] \\
&= \sum_{w=0}^{K} b_w \big(\mathbb{E}[Q_w(X|_S)] - \mathbb{E}[Q_w(Y|_S)]\big) && \text{(by (2))} \\
&\leq \sum_{w=0}^{K} \big|\mathbb{E}[Q_w(X|_S)] - \mathbb{E}[Q_w(Y|_S)]\big| && \text{(by boundedness of } b_w\text{)} \\
&= \sum_{w=0}^{K} \big|\mathbb{E}[p_w(\phi(|X|)] - \mathbb{E}[p_w(\phi(|Y|))]\big| && \text{(by symmetry of } X, Y\text{, and Fact 11)} \\
&\leq \sum_{w=0}^{K} \big|\mathbb{E}[q_w(\phi(|X|))] - \mathbb{E}[q_w(\phi(|Y|))]\big| + 8\sqrt{K}\exp(-k^2/1156K) && \text{(by Lemma 13)} \\
&= O(K^{3/2}) \cdot e^{-k^2/1156K} && \text{(by } k\text{-wise indistinguishability of } X, Y\text{)}
\end{aligned}$$

Therefore, X and Y are $(K, O(K^{3/2}) \cdot e^{-k^2/1156K})$-wise indistinguishable for $2 \leq K \leq n/64$. ◀

[10] p_w, C_w, and Z_w also depend on K and n but we omit those arguments from the notation as they will be fixed in the proof.

3.1 Proof of Lemma 13

We will prove Lemma 13 by studying the Chebyshev expansion of p_w. To this end we take a brief detour into Chebyshev polynomials and an even briefer one into Fourier analysis.

Chebyshev polynomials

The Chebyshev polynomials are a family of real polynomials $\{T_d\}$, 1-bounded on $[-1, 1]$, with T_d having degree d. We extend the definition to negative indices by setting $T_{-d} = T_d$. The Chebyshev polynomials are orthogonal with respect to the measure $d\sigma(t) = (1 - t^2)^{-1/2}dt$ supported on $[-1, 1]$. Therefore every degree-K polynomial $p\colon \mathbb{R} \to \mathbb{R}$ has a unique (symmetrized) Chebyshev expansion

$$p(t) = \sum_{d=-K}^{K} c_d T_d(t), \qquad c_{-d} = c_d$$

where $c_{-K}, \ldots, c_K$ are the *Chebyshev coefficients* of p.

The Chebyshev polynomials satisfy the following identity, which plays an important role in our analysis:

▶ **Fact 14.** $t \cdot T_d(t) = \frac{1}{2}T_{d-1}(t) + \frac{1}{2}T_{d+1}(t)$.

This formula, together with the "base cases" $T_0(t) = 1$ and $T_1(t) = t$, specifies all Chebyshev polynomials.

We will also need the following form of Parseval's identity for univariate polynomials.

▷ Claim 15 (Parseval's identity). For every complex polynomial h, the sum of the squares of the magnitudes of the coefficients of h equals $\mathbb{E}_z[|h(z)|^2]$, where z is a random complex number of magnitude 1.

Proof outline

We will argue that the Chebyshev expansion $\sum_{d=-K}^{K} c_d T_d(t)$ of $p_w(t)$ has small weight on the coefficients c_d when $|d| > k$. Zeroing out those coefficients then yields a good degree-k approximation of p_w as desired.

The upper bound on the Chebyshev coefficients of p_w is derived in two steps. The first step, which is of an algebraic nature, expresses the Chebyshev coefficients of p_w as regular coefficients of a related polynomial g.[11] We are interested in the coefficients of the derived polynomial $g_\varepsilon(s) = g((1+\varepsilon)s)$, which represent the Chebyshev coefficients c_d of p_w amplified by the exponential scaling factor $(1+\varepsilon)^d$.

The second step, which is analytic, upper bounds the magnitude of the coefficients of $g_\varepsilon(s)$. The main tool is Parseval's identity, which identifies the sum of the squares of these coefficients by the average magnitude of g_ε over the complex unit circle $\mathbb{E}_\theta |g((1+\varepsilon)e^{i\theta})|^2$. We bound the *maximum* magnitude $\max_\theta |g((1+\varepsilon)e^{i\theta})|^2$ by explicitly analyzing the function g. This step comprises the bulk of our proof.

The third step translates the bound on the squared 2-norm $\sum_{d=-K}^{K}(1+\varepsilon)^{2d}c_d^2$ of the amplified coefficients into a tail bound on c_d by optimizing over a suitable value of ε. This is analogous to the standard derivation of Chernoff-Hoeffding bounds by analysis of the moment generating function of the relevant random variable.

[11] We omit the dependence on w as this parameter remains constant throughout the proof.

We now sketch how this outline is executed for the special case where n tends to infinity while k and K remain fixed. Although this setting is technically much easier, it allows us to highlight the main conceptual points of our argument. The analysis for finite n can be viewed as an approximation of this proof strategy.

Sketch of the limiting case $n \to \infty$

By the expansion (4) of p_w, as n tends to infinity p_w converges uniformly to the function

$$p_w^\infty(t) = C_w \cdot (t-1)^w (t+1)^{K-w},$$

as this corresponds to Fact 11 when the bits of the string Z are independent and $(1-t)/2$-biased. As $p_w^\infty(t)$ is a probability for every $t \in [-1,1]$, Claim 12 follows immediately.

Step 1. Our algebraic treatment of the Chebyshev transform yields that the Chebyshev coefficient c_d of p_w^∞ is the $(K+d)$-th regular coefficient of the polynomial

$$g^\infty(s) = C_w \left(\frac{s-1}{\sqrt{2}}\right)^{2w} \left(\frac{s+1}{\sqrt{2}}\right)^{2(K-w)}. \tag{5}$$

Step 2. The evaluation of the polynomial $g_\varepsilon^\infty(s) = g^\infty((1+\varepsilon)s)$ at $s = e^{i\theta}$ satisfies the identity

$$\left|g^\infty\left((1+\varepsilon)e^{i\theta}\right)\right| = (1+\varepsilon)^K \cdot (1+\delta)^K \cdot C_w \cdot \left(1 - \frac{\cos\theta}{1+\delta}\right)^w \left(1 + \frac{\cos\theta}{1+\delta}\right)^{K-w}, \tag{6}$$

where $\delta = \varepsilon^2/2(1+\varepsilon)$. This happens to equal

$$(1+\varepsilon)^K (1+\delta)^K p_w(\cos\theta/(1+\delta)), \tag{7}$$

and is in particular uniformly bounded by $(1+\varepsilon)^K(1+\delta)^K$ for all θ. This similarity between p^∞ and g_ε^∞ is the crux of our analysis.

Step 3. By Parseval's identity, after suitable shifting and cancellation, the amplified sum of Chebyshev coefficients $\sum_{d=-K}^{K} (1+\varepsilon)^{2d} c_d^2$ is upper bounded by $(1+\delta)^{2K}$. Therefore the tail $\sum_{k \geq d} c_d^2$ can have value at most $(1+\delta)^{2K}/(1+\varepsilon)^{2k} \leq \exp(2K\varepsilon^2 - 2(\varepsilon - \varepsilon^2/2)k)$. This upper bound holds for all $\varepsilon \in [0,1]$, and plugging in the approximate minimizer $\varepsilon = k/2K$ yields a bound of the desired form $\exp(-\Omega(k^2/K))$.

Outline of the general case

We now give the outline of our full proof for the general case and relevant technical statements that we use to prove our main upper bound. Identity (5) generalizes to the following statement:

▷ Claim 16. The Chebyshev coefficient c_d of p_w is the $(K+d)$-th regular coefficient of the polynomial

$$g(s) = C_w \prod_{z \in Z_w} \left(\frac{s^2 - 2sz + 1}{2}\right),$$

where C_w is as in Equation (4).

The general form of identity (6) is:

▷ Claim 17. For $\varepsilon > 0$, $\delta = \varepsilon^2/2(1+\varepsilon)$, and $\theta \in [-\pi, \pi]$,

$$\left|g((1+\varepsilon)e^{i\theta})\right|^2 = (1+\varepsilon)^{2K}(1+\delta)^{2K} \cdot C_w^2 \prod_{z \in Z_w} h_{\delta(1+1/(1+\delta))}\left(\frac{\cos\theta}{1+\delta}, z\right)$$

where $h_\delta(s,z) = (s-z)^2 + \delta(1-z^2)$.

Owing to the second term in h_δ, there is no identity analogous to (7) when n is finite and p_w has zeros inside $(-1,1)$. Nevertheless, $\prod_{z\in Z_w} h_\delta(s,z)$ can be uniformly bounded either by a sufficiently small multiple of $p_w(s)^2$, or a fixed quantity that is constant in the parameter range of interest.

▷ Claim 18. Assume $n \geq 64K$ and $w \leq K/2$. Then

$$C_w^2 \cdot \prod_{z\in Z_w} h_\delta(s,z) \leq \begin{cases} e^{65\delta K} \cdot p_w(s)^2 & \text{if } |s| \leq 1 - w/16K \\ e^{65\delta K} & \text{if } 1 - w/16K \leq |s| \leq 1. \end{cases}$$

We now prove Lemma 13. Due to space limitations, we omit the proof of Claim 16, which follows via induction and is an application of Fact 14, and the proof of Claim 17, which consists of a lengthy but relatively straightforward calculation. Claim 18 is proved in Section A.

▶ **Fact 19.** $p_w(t) = p_{K-w}(1-t)$.

Proof. By Fact 11, both sides are degree-K polynomials that agree on $n+1 > K$ points so they are identical. ◀

Proof of Lemma 13. By Fact 19 we may and will assume that $w \leq K/2$. Let $p_w = \sum_{d=-K}^{K} c_d T_d$. The approximating polynomial q_w is $\sum_{|d|<k} c_d T_d$. It remains to prove a tail upper bound on the Chebyshev coefficients. By Claim 16, the $(K+d)$-th coefficient of $g(s)$ is c_d. Therefore the polynomial $g_\varepsilon(s) = g((1+\varepsilon)s)$ has coefficients $(1+\varepsilon)^{K+d}c_d$ as d ranges from $-K$ to K. We apply Parseval's identity (Claim 15) to g_ε.

It follows that

$$\begin{aligned} \sum_{d=-K}^{K} (1+\varepsilon)^{2(K+d)} c_d^2 &= \mathbb{E}_\theta \left|g((1+\varepsilon)e^{i\theta})\right|^2 \\ &\leq \max_{\theta\in[-\pi,\pi]} \left|g((1+\varepsilon)e^{i\theta})\right|^2 \\ &= \max_{s\in[-1,1]} (1+\varepsilon)^{2K}(1+\delta)^{2K} \cdot C_w^2 \prod_{z\in Z_w} h_{\delta(1+1/(1+\delta))}(s/(1+\delta), z), \end{aligned}$$

by Claim 17. Since $0 \leq \delta = \varepsilon^2/2(1+\varepsilon) \leq 1/2$, for simplicity we may replace $h_{\delta(1+1/(1+\delta))}(s/(1+\delta), z)$ by $h_{2\delta}(s,z)$ in the above inequality. This gives the following approximation bound for $\alpha = \max_{t\in[-1,1]} |p_w(t) - q_w(t)|$:

$$\begin{aligned} \alpha &= \max_{t\in[-1,1]} \left| \sum\nolimits_{|d|\geq k} c_d T_d(t) \right| \\ &\leq \sum\nolimits_{|d|\geq k} |c_d| \max_{t\in[-1,1]} |T_d(t)| \\ &\leq 2\sum_{d\geq k} |c_d| \qquad \text{(by symmetry and boundedness of } T_d) \end{aligned}$$

$$\begin{aligned}
\ldots &\le 2\sqrt{K} \cdot \sqrt{\sum\nolimits_{d \ge k} c_d^2} \qquad \text{(by Cauchy-Schwarz)} \\
&\le 2\sqrt{K} \cdot \sqrt{(1+\varepsilon)^{-2(K+k)} \sum\nolimits_{d \ge k} (1+\varepsilon)^{2(K+d)} c_d^2} \\
&\le 2\sqrt{K} \sqrt{(1+\varepsilon)^{-2k} \cdot (1+\delta)^{2K} \cdot \max_{s \in [-1,1]} C_w^2 \prod_{z \in Z_w} h_{2\delta}(s,z)}.
\end{aligned}$$

By the boundedness of p_w (Claim 12), the upper bounds in Claim 18 can be unified by the inequality $C_w^2 \prod_{z \in Z_w} h_{2\delta}(s,z) \le 4e^{130\delta K}$ that is valid for all $s \in [-1,1]$. Since $1+\delta \le e^{\delta}$ and $1+\varepsilon \ge e^{\varepsilon - \varepsilon^2/2}$ for $0 \le \varepsilon \le 1$,

$$\alpha \le 2\sqrt{K} \cdot \sqrt{\frac{(1+\delta)^{2K}}{(1+\varepsilon)^{2k}} \cdot 4e^{130\delta K}} \le 4\sqrt{K} \cdot \sqrt{e^{132\delta K - 2\varepsilon k + \varepsilon^2 k}} \le 4\sqrt{K} \cdot \sqrt{e^{67\varepsilon^2 K - 2\varepsilon k}},$$

where the last inequality follows from the definition $\delta = \varepsilon^2/2(1+\varepsilon)$. Setting $\varepsilon = k/34K$ we obtain that $\alpha \le 4\sqrt{K} \cdot e^{-k^2/1156K}$. ◀

4 Robustness of Symmetric Secret Sharing Against Consolidation

Consider a secret sharing scheme with tn parties, divided in n blocks of size t, that is perfectly secure against size-k coalitions. If all parties in each block come together and consolidate their information even into a single bit, the number of *blocks* against which the scheme remains secure drops to k/t. In general this is the best possible, with linear schemes providing tight examples.

The following corollary, proven in the full version, shows that if the distribution over shares is symmetric then much better security against this type of attack can be obtained.

▶ **Corollary 20.** *Let $f_1, \ldots, f_n \colon \{0,1\}^t \to \{0,1\}$. Assume X, Y are k-wise indistinguishable symmetrically distributed random variables over tn-bit strings. Write $X = X_1 \ldots X_n$, $Y = Y_1 \ldots Y_n$, where all blocks X_i, Y_i have size t. For every K, the n-bit random variables $X' = f_1(X_1) \ldots f_n(X_n)$ and $Y' = f_1(Y_1) \ldots f_n(Y_n)$ are $O((tK)^{3/2} n^K e^{-k^2/1156tK})$-close to being perfectly K-wise indistinguishable, assuming $K \le n/64$.*

References

1 Andris Ambainis. Quantum Search with Variable Times. *Theory Comput. Syst.*, 47(3):786–807, 2010.

2 Shalev Ben-David, Adam Bouland, Ankit Garg, and Robin Kothari. Classical Lower Bounds from Quantum Upper Bounds. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 339–349, 2018.

3 Andrej Bogdanov. Approximate degree of AND via Fourier analysis. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:197, 2018.

4 Andrej Bogdanov, Yuval Ishai, Emanuele Viola, and Christopher Williamson. Bounded Indistinguishability and the Complexity of Recovering Secrets. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 593–618, 2016.

5 Andrej Bogdanov and Christopher Williamson. Approximate Bounded Indistinguishability. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 53:1–53:11, 2017.

6 Harry Buhrman, Richard Cleve, Ronald de Wolf, and Christof Zalka. Bounds for Small-Error and Zero-Error Quantum Algorithms. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 358–368, 1999.

7 Mark Bun, Robin Kothari, and Justin Thaler. The polynomial method strikes back: tight quantum query bounds via dual polynomials. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 297–310, 2018.

8 Mark Bun and Justin Thaler. Dual Lower Bounds for Approximate Degree and Markov-Bernstein Inequalities. In *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, pages 303–314, 2013.

9 Mark Bun and Justin Thaler. Hardness Amplification and the Approximate Degree of Constant-Depth Circuits. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, pages 268–280, 2015.

10 Mark Bun and Justin Thaler. A Nearly Optimal Lower Bound on the Approximate Degree of AC^0. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 1–12, 2017.

11 Mark Bun and Justin Thaler. The Large-Error Approximate Degree of AC^0. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:143, 2018.

12 Karthekeyan Chandrasekaran, Justin Thaler, Jonathan Ullman, and Andrew Wan. Faster private release of marginals on small databases. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 387–402, 2014.

13 Noam D. Elkies (https://mathoverflow.net/users/14830/noam-d elkies). Uniform approximation of x^n by a degree d polynomial: estimating the error. MathOverflow. URL: `https://mathoverflow.net/q/70527`.

14 Xuangui Huang and Emanuele Viola. Almost Bounded Indistinguishability and Degree-Weight Tradeoffs, 2019. Manuscript.

15 Jeff Kahn, Nathan Linial, and Alex Samorodnitsky. Inclusion-exclusion: Exact and approximate. *Combinatorica*, 16(4):465–477, 1996.

16 Pritish Kamath and Prashant Vasudevan. Approximate Degree of AND-OR trees, 2014. Manuscript available at `https://www.scottaaronson.com/showcase3/kamath-pritish-vasudevan-prashant.pdf`.

17 Philip N. Klein and Neal E. Young. On the Number of Iterations for Dantzig-Wolfe Optimization and Packing-Covering Approximation Algorithms. *SIAM J. Comput.*, 44(4):1154–1172, 2015.

18 Marvin Minsky and Seymour Papert. *Perceptrons*. MIT Press, Cambridge, MA, 1969.

19 Moni Naor and Adi Shamir. Visual Cryptography. In *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*, pages 1–12, 1994.

20 Noam Nisan and Mario Szegedy. On the Degree of Boolean Functions as Real Polynomials. *Computational Complexity*, 4:301–313, 1994.

21 Ramamohan Paturi. On the Degree of Polynomials that Approximate Symmetric Boolean Functions (Preliminary Version). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 468–474, 1992.

22 Sushant Sachdeva and Nisheeth K. Vishnoi. Faster Algorithms via Approximation Theory. *Foundations and Trends in Theoretical Computer Science*, 9(2):125–210, 2014.

23 Rocco A. Servedio, Li-Yang Tan, and Justin Thaler. Attribute-Efficient Learning and Weight-Degree Tradeoffs for Polynomial Threshold Functions. In *COLT 2012 - The 25th Annual Conference on Learning Theory, June 25-27, 2012, Edinburgh, Scotland*, pages 14.1–14.19, 2012.

24 Alexander A. Sherstov. Approximating the AND-OR Tree. *Theory of Computing*, 9:653–663, 2013.

25 Alexander A Sherstov. Breaking the Minsky–Papert Barrier for Constant-Depth Circuits. *SIAM Journal on Computing*, 47(5):1809–1857, 2018.

26 Alexander A. Sherstov. The Power of Asymmetry in Constant-Depth Circuits. *SIAM J. Comput.*, 47(6):2362–2434, 2018.

27 Alexander A Sherstov and Pei Wu. Near-Optimal Lower Bounds on the Threshold Degree and Sign-Rank of AC^0. *arXiv preprint arXiv:1901.00988*, 2019. To appear in STOC 2019.

28 Robert Špalek. A Dual Polynomial for OR. *CoRR*, abs/0803.4516, 2008.

A Proof of Claim 18

The objective is to uniformly bound the value of the function

$$h_\delta(s) = C_w^2 \cdot \prod_{z \in Z_w} h_\delta(s,z), \qquad \text{where} \qquad h_\delta(s,z) = (s-z)^2 + \delta(1-z^2)$$

for $s \in [-1,1]$. When k, K are fixed and n becomes large, all zeros in Z_w approach -1 or $+1$, $h_\delta(s,z)$ uniformly approaches $h_0(s,z) = (s-z)^2$, $h_w(s)$ approaches $h_0(s) = p_w^\infty(s)$ and is therefore uniformly bounded.

The main difficulty in extending this argument to finite n is that $h_\delta(s,z)$ can no longer be uniformly bounded by a multiple of $(s-z)^2$ since when s equals z, the latter function vanishes but the former one doesn't. For this reason, we divide the analysis into two parameter regimes. When s is bounded away from the set of zeros Z_w, an approximation of the infinitary term-by-term argument can be carried out. When s is near the zeroes, we argue that $h_\delta(s)$ cannot be much larger than $h_\delta(s_0)$ for an s_0 that is even farther away from Z_w, and then argue that $h_0(s_0) = p_w(s_0)^2$ must be small because it represents the square of a probability of a rare event.

▶ **Fact 21.** *$h_\delta(s,z)h_\delta(s,-z) = h_\delta(-s,z)h_\delta(-s,-z)$.*

▶ **Fact 22.** *$h_\delta(s,z) \le h_\delta(|s|,z)$ when $z \le 0$ and $s \ge 0$.*

▶ **Fact 23.** *$h_\delta(s,z) \le h_\delta(s_0,z)$ when $s_0 \le s \le 1$, $s_0 \le 2z-1$, and $|z| \le 1$.*

Proof. The fact is equivalent to checking that $(s_0-z)^2 - (s-z)^2 \ge 0$ when $s_0 \le s \le 1$ and $s_0 \le 2z-1$. If $s \le z$ then we have that $s_0 \le s \le z$ from which it immediately follows that $(s_0-z)^2 \ge (s-z)^2$. If $s > z$ then $(s-z)^2$ is at most $(1-z)^2$. However, since $|z| \le 1$, we have that $s_0 \le 2z-1 \le z$ and thus $(s_0-z)^2$ is always at least $(z-(2z-1))^2 = (1-z)^2$. Again we have that $(s_0-z)^2 \ge (s-z)^2$. ◀

We begin by reducing to the case of non-negative inputs $s \in [0,1]$.

▷ Claim 24. Assuming $w \le K/2$, $h_\delta(s) \le h_\delta(|s|)$.

Proof. When $w \le K/2$ then elements of Z_w (3) can be split into w pairs of the form $A = \{(-1+2h/n, 1-2h/n) \colon 0 \le h < w\}$, and $K-2w$ remaining elements $B = \{-1+2h/n \colon w \le h < K-w\}$ are all non-positive. By Fact 21, $\prod_{(-z,z)\in A} h_\delta(s,z)h_\delta(s,-z) = \prod_{(-z,z)\in A} h_\delta(|s|,z)h_\delta(|s|,-z)$. By Fact 22, $\prod_{z\in B} h_\delta(s,z) \le \prod_{z\in B} h_\delta(|s|,z)$. Therefore the product $\prod_{z\in Z_w} h_\delta(s,z) \le \prod_{z\in Z_w} h_\delta(|s|,z)$. ◁

The following claim handles values of s in the range $[0, 1-w/16K]$.

▷ Claim 25. Assuming $0 \le s \le 1 - w/16K$,

$$h_\delta(s,z) \le \begin{cases} (1+\delta)(s-z)^2, & \text{if } z \le -1/\sqrt{2}. \\ (1+(64K/w)\delta)(s-z)^2, & \text{if } z \ge 1 - w/32K \end{cases}$$

Proof. The ratio $h_\delta(s,z)/(s-z)^2$ equals $1+((1-z^2)/(s-z)^2)\delta$. The number $(1-z^2)/(s-z)^2$ is at most 1 when $s \ge 0$ and $z \le -1/\sqrt{2}$ and at most the following when $z \ge 1 - w/32K$.

$$\frac{1-(1-w/32K)^2}{((1-w/16K)-(1-w/32K))^2} \le \frac{2w/32K}{(w/32K)^2} = 64K/w. \qquad \triangleleft$$

▶ **Corollary 26.** *Assuming $0 \le s \le 1 - w/16K$ and $n \ge 64K$, $h_\delta(s) \le e^{65\delta K} h_0(s)$.*

Proof. By the choice of parameters, all zeros in Z_- meet the criterion for the first inequality in Claim 25, while all zeros in Z_+ meet the criterion for the second one. Therefore

$$\begin{aligned} h_\delta(s) &= C_w^2 \prod_{z \in Z_-} h_\delta(s,z) \prod_{z \in Z_+} h_\delta(s,z) \\ &\le C_w^2 \prod_{z \in Z_-} (1+\delta)(s-z)^2 \prod_{z \in Z_+} (1+(64K/w)\delta)(s-z)^2 \\ &\le (1+\delta)^{K-w}(1+(64K/w)\delta)^w \cdot C_w^2 \prod_{z \in Z_-} h_0(s,z) \prod_{z \in Z_+} h_0(s,z) \\ &\le e^{\delta K} \cdot e^{64\delta K} \cdot h_0(s). \end{aligned}$$

◀

The following two claims handle values of s in the range $[1 - w/16K, 1]$.

▷ Claim 27. Assuming $w \le K$ and $1 - w/8K \le s_0 \le 1 - w/16K \le s \le 1$,

$$h_\delta(s,z) \le \begin{cases} h_\delta(s_0,z), & \text{if } z \ge 1 - w/32K \\ (1+w/8K)^2 \cdot h_\delta(s_0,z), & \text{if } z \le -w/8K. \end{cases}$$

Proof. By the choice of parameters the first inequality follows from Fact 23. For the second one, we upper bound the ratio

$$\frac{(s-z)^2}{(s_0-z)^2} \le \frac{(1-z)^2}{(1-z-w/8K)^2} = \left(1 + \frac{w/8K}{1-z-w/8K}\right)^2 \le \left(1 + \frac{w}{8K}\right)^2.$$

This is greater than one, so $(s-z)^2 + \delta(1-z^2) \le (1+w/8K)^2((s_0-z)^2 + \delta(1-z^2))$ as desired. $\triangleleft$

▶ **Corollary 28.** *Assuming $1 - w/8K \le s_0 \le 1 - w/16K \le s \le 1$ and $n \ge 2K$, $h_\delta(s) \le e^{w/4} h_\delta(s_0)$.*

Proof. By the choice of parameters, all zeros in Z_- meet the criterion for the first inequality in Claim 27, while all zeros in Z_+ meet the criterion for the second one. Therefore

$$\begin{aligned} h_\delta(s) &= C_w^2 \prod_{z \in Z_-} h_\delta(s,z) \prod_{z \in Z_+} h_\delta(s,z) \\ &\le C_w^2 \prod_{z \in Z_-} (1+w/8K)^2 \cdot h_\delta(s_0,z) \prod_{z \in Z_+} h_\delta(s_0,z) \\ &= (1+w/8K)^{2|Z_-|} \cdot h_\delta(s_0) \\ &\le (1+w/8K)^{2K} \cdot h_\delta(s_0) \le e^{w/4} h_\delta(s_0). \end{aligned}$$

◀

▷ Claim 29. If s_0 is of the form $1 - 2h/n$ for some integer $0 \le h \le wn/e^2K$ then $0 \le p_w(s_0) \le e^{-w}$.

Proof. By Fact 11, $p_w(s_0)$ is the probability that a random string of Hamming weight h and length n has exactly w ones in its first K positions. The probability that it has at least w ones in its first K positions is at most

$$\binom{K}{w} \cdot \frac{h}{n} \cdot \frac{h-1}{n-1} \cdots \frac{h-w+1}{n-w+1} \le \left(\frac{eK}{w}\right)^w \left(\frac{h}{n}\right)^w \le e^{-w}. \qquad \triangleleft$$

Proof of Claim 18. By Claim 24 we may assume $s \in [0,1]$. When $0 \le s \le 1 - w/16K$ the result follows from Corollary 26. When $1 - w/16K \le |s| \le 1$, by the assumption $n \ge 64K$ there must exist a value s_0 between $1 - w/8K$ and $1 - w/16K$ that is of the form $1 - 2h/n$. In particular $h \le wn/e^2K$. Then

$$h_\delta(s) \le e^{w/4} h_\delta(s_0) \le e^{w/4} e^{65\delta K} p_w(s_0)^2 \le e^{65\delta K - 7w/4},$$

where the inequalities follow from Corollary 28, Corollary 26, and Claim 29, respectively. $\triangleleft$

B Proofs of Corollary 4 and Theorem 5

B.1 Proof of Corollary 4

Proof of Corollary 4. Corollary 3 implies the existence of a $\phi \left(= \frac{\mu - \nu}{2}\right)$ satisfying $\|\phi\|_1 = 1$, $\langle f, \phi \rangle = \varepsilon$ for some $\varepsilon = \Omega(1)$ and $\langle \phi, q \rangle \le K^{3/2} \cdot 2^{-\Omega\left(\widetilde{\deg}_{1/3}(f)^2/K\right)}$ for any parity of degree at most K.

For any p of degree K and weight at most w,

$$\|f - p\|_\infty = \|f - p\|_\infty \|\phi\|_1 \ge \langle \phi, f - p \rangle = \langle \phi, f \rangle - \langle \phi, p \rangle \ge \varepsilon - w \cdot K^{3/2} \cdot 2^{-\Omega\left(\widetilde{\deg}_{1/3}(f)^2/K\right)}.$$

Thus, we conclude that $W_{\varepsilon/2}(f, K) = K^{-3/2} \cdot 2^{\Omega\left(\widetilde{\deg}_{1/3}(f)^2/K\right)}$. Corollary 4 now follows using standard error reduction techniques that show that $\widetilde{\deg}_\varepsilon(f) = \Theta(\widetilde{\deg}_{1/3}(f))$ for all constants $0 < \varepsilon < 1/2$. ◀

B.2 Proof of Theorem 5

We first require the following lemma. This lemma builds on ideas in [23, Claim 2], which showed a similar result for $t = \Theta(1)$.

▶ **Lemma 30.** *For any $y \in \{0,1\}^n$, denote by EQ_y the function on $\{0,1\}^n$ that outputs 1 on input y, and 0 otherwise. Then for any $t > 0$ and $d > \sqrt{nt \log n}$, we have $W_{n^{-O(t)}}(\mathsf{EQ}_y, d) \le 2^{O(nt \log^2(n)/d)}$.*

Proof. Note that for any $y \in \{-1,1\}^n$, the function EQ_y is just the AND function on n input bits (with 0-1 valued output), with possibly negated input variables. Thus it suffices to give an approximating polynomial for the AND function on n bits. We now express AND_n as $\mathsf{AND}_\ell \circ \mathsf{AND}_{n/\ell}$, where ℓ is a parameter we will set later. We compute the inner $\mathsf{AND}_{n/\ell}$ exactly and approximate the outer AND_ℓ to error $n^{-\Omega(t)}$. This can be done with a polynomial p of degree $O\left(\sqrt{\ell \log(n^t)}\right)$ [15, 6]. Combining the fact that p is bounded by $1 + n^{-\Omega(t)} \le 2$ at all Boolean inputs with Parseval's identity and the Cauchy-Schwarz inequality, it can

be seen that the weight of p is at most $\ell^{O\left(\sqrt{\ell \log(n^t)}\right)}$.[12] It is well known that the exact multilinear polynomial representation of $\mathsf{AND}_{n/\ell}$ has constant weight. Hence, by composing p with the multilinear polynomial that exactly computes $\mathsf{AND}_{n/\ell}$, we obtain an approximation q for AND_n of degree $O\left(n\sqrt{\frac{t\log n}{\ell}}\right)$, error $n^{-\Omega(t)}$, and weight $2^{O\left(\sqrt{\ell t \log^3 n}\right)}$. We now fix the value of ℓ to $\ell := \frac{n^2 t \log n}{d^2} < n$, thereby ensuring that the degree of q is at most d. With this setting of ℓ, the weight of q is at most $2^{O(nt\log^2(n)/d)}$, proving the lemma. ◀

Proof of Theorem 5. Let $f : \{0,1\}^n \to \{0,1\}$ be any symmetric function, corresponding to the univariate predicate $D_f : \{0\} \cup [n] \to \{0,1\}^n$. For the purpose of this proof, let us denote by k_f the smallest i for which f is constant on inputs of Hamming weight in the interval $[i+1, n-i-1]$. Without loss of generality, $f(x) = 0$ for strings of x Hamming weight between $k_f + 1$ and $n - k_f - 1$. The case where $f = 1$ on input strings of Hamming weight between $k_f + 1$ and $n - k_f - 1$ can be proved using a similar argument. Define $\mathrm{supp}(f) := \{x \in \{0,1\}^n : f(x) = 1\}$. Note that $|\mathrm{supp}(f)| \leq 2 \cdot n^{k_f}$.

Observe that $f(x) = \sum_{y \in \mathrm{supp}(f)} \mathsf{EQ}_y(x)$. Lemma 30 implies, for each $y \in \mathrm{supp}(f)$, the existence of polynomials p_y of degree K and weight $2^{O(nk_f \log^2(n)/K)}$, which approximate EQ_y to error $\frac{1}{6} \cdot n^{-k_f}$. Define a polynomial $p\colon \{0,1\}^n \to \mathbb{R}$ by $p(x) = \sum_{y \in \mathrm{supp}(f)} p_y(x)$. Clearly p has degree K, weight at most $n^{O(k_f)} \cdot 2^{O(nk_f \log^2(n)/K)} = 2^{\tilde{O}(nk_f/K)}$, and error at most $|\mathrm{supp}(f)| \cdot n^{-k_f}/6 \leq 1/3$, where the upper bounds on the weight and error follow from the triangle inequality.

The theorem now follows standard error reduction techniques and Paturi's theorem [21], which states that for symmetric functions, $\widetilde{\deg}(f) = \Theta\left(\sqrt{n \cdot k_f}\right)$. ◀

▶ Remark 31. The upper bound obtained in Theorem 5 is more general than as stated, and the only property of symmetric functions it exploits is that symmetric functions of low approximate degree are highly biased. More specifically, the proof of Theorem 5 shows that any function $f\colon \{0,1\}^n \to \{0,1\}$ with $\min\{|f^{-1}(0)|, |f^{-1}(1)|\} \leq n^t$ satisfies $W_\varepsilon(f,K) \leq 2^{\tilde{O}(nt/K)}$ for any $K \geq \sqrt{nt \log n}$.

C Proof of Theorem 6

Proof outline. As we explain in more detail in the proof itself, it is sufficient to establish the theorem for fixed k and K and infinitely many n because the statement is downward reducible in n.

Using the Chebyshev approximation formulas from Section 3 we derive explicit lower bounds on the large Chebyshev coefficients on the polynomial p_0 representing the distinguishing advantage of the AND function on K inputs. Owing to orthogonality and boundedness of the Chebyshev polynomials, this is a lower bound on the approximate degree of AND_K. By strong duality as given in the following Claim (see [4]) we obtain Theorem 6.

▷ Claim 32. If $\widetilde{\deg}_{\varepsilon/2}(F_n) \geq k$ then there exists a pair of perfectly k-wise indistinguishable distributions μ, ν over $\{0,1\}^n$ such that $\mathbb{E}_{X\sim\mu}[F_n(X)] - \mathbb{E}_{Y\sim\nu}[F_n(Y)] \geq \varepsilon$.

[12] Building on [6], It is possible to derive explicit ε-approximating polynomials for AND where the degree is $O\left(\sqrt{\ell \log(1/\varepsilon)}\right)$ and the weight is $2^{O\left(\sqrt{\ell \log(1/\varepsilon)}\right)}$ rather than $\ell^{O\left(\sqrt{\ell \log(1/\varepsilon)}\right)}$. Using this tighter weight bound would improve our final result by a factor of $\log n$ in the exponent. We omit this tighter result for brevity.

Recall that the Chebyshev polynomials are orthogonal under the measure $d\sigma(t) = (1-t^2)^{-1/2}dt$ supported on $[-1,1]$. We will need the following identity for their average square magnitude under this measure:

$$\mathbb{E}_{t\sim\sigma}[T_d(t)^2] = 1/2 \qquad \text{when } d > 0. \tag{8}$$

Proof of Theorem 6. By symmetry of the distinguishers, μ and ν can be assumed symmetric. Let F_n denote the function on $\{0,1\}^n$ that outputs $\mathsf{AND}_K\left(x|_{\{1,\ldots,K\}}\right)$, i.e., F_n outputs the AND of the first $K < n$ bits of the input. We prove the theorem for $G_n(x_1,\ldots,x_n) = \mathsf{NOR}(x|_{\{1,\ldots,K\}})$. By the symmetry of 0 and 1 inputs the theorem also holds for F_n.

First, we claim that the statement of Theorem 6 is stronger as n becomes larger, so it is sufficient to prove it in the limiting case when n approaches infinity and k, K are fixed. Suppose that μ and ν are distributions over n bit strings that are k-wise indistinguishable yet are ε-reconstructable by G_n. We must show that there are distributions μ' and ν' over $\{0,1\}^{n-1}$ are k-wise indistinguishable yet are ε-reconstructable by G_{n-1}. But this holds for μ' (respectively ν') that generate a random sample from μ (respectively, ν) and then throw away the last bit.

If the statement was false then by Claim 32 there would exist degree-k polynomials $\tilde{G}_n$ that approximate G_n pointwise on $\{0,1\}^n$ to within error $\varepsilon = \sqrt{2^{-4K+1}\sum_{d>K}\binom{2K}{K+d}^2}$ for almost all n. Applying the construction from the proof of Fact 11 to $\tilde{G}_n$, there exist univariate degree-k polynomials $\tilde{p}_0^n$ approximating p_0^n on the set of points $W_n = \{-1+2h/n \colon 0 \le h \le n\}$ to within error ε. We emphasize the dependence on n as it will play a role in the proof.

By Formula (3) the polynomial p_0^n has the form

$$p_0^n(t) = C_0^n \prod_{z\in Z_0^n}(t-z),$$

where $Z_0^n = \{-1+2h/n \colon 0 \le h < K\}$ (the set Z_+ is empty). The value $p_n^0(1)$ is the probability that G_n accepts the all-zero string, so it must equal one. The constant C_0^n must therefore equal $\prod_{z\in Z_0^n}(1-z)^{-1}$. As n tends to infinity, the set Z_0 converges to a single zero at -1 of multiplicity K, so the sequence p_0^n converges uniformly to the polynomial

$$p_0^\infty(t) = 2^{-K}(t+1)^K.$$

By the triangle inequality, for every $\delta > 0$ and all sufficiently large n, $\tilde{p}_0^n$ is within $\varepsilon+\delta$ of p_0^∞ on the set W_n. A degree-k polynomial is determined by its values on W_{k+1} and the set of degree-k polynomials that are within $\varepsilon+\delta$ of p_0^∞ on W_{k+1} is compact. Therefore the sequence of approximating polynomials $\tilde{p}_0^n$ must contain a subsequence (for values of n that are multiples of $k+1$) that converges (uniformly) to a limiting degree-k polynomial $\tilde{p}_0^\infty$. Since $\tilde{p}_0^n$ is within $\varepsilon+\delta$ of p_0^n on W_n for infinitely many n, $\tilde{p}_0^\infty$ must be within $\varepsilon+2\delta$ of p_0^∞ on W_n for infinitely many n. The union of these sets W_n is dense in $[-1,1]$, and by continuity p_0^∞ can be $\varepsilon+\delta$-approximated by the degree-k polynomial $\tilde{p}_0^\infty$ everywhere on $[-1,1]$. As δ was arbitrary it follows that the ε-approximate degree of p_0^∞ can be at most k.

All that remains to prove that this is not true, i.e., to show a lower bound of k on the ε-approximate degree of p_0^∞. This lower bound is known (see, e.g., [13]); we provide the details in the full version. ◀

Improved Extractors for Recognizable and Algebraic Sources

Fu Li
Department of Computer Science, University of Texas at Austin, USA
fuli2015@cs.utexas.edu

David Zuckerman
Department of Computer Science, University of Texas at Austin, USA
diz@cs.utexas.edu

Abstract

We study the task of seedless randomness extraction from recognizable sources, which are uniform distributions over sets of the form $\{x : f(x) = 1\}$ for functions f in some specified class $\mathcal{C}$. We give two simple methods for constructing seedless extractors for $\mathcal{C}$-recognizable sources.

Our first method shows that if $\mathcal{C}$ admits XOR amplification, then we can construct a seedless extractor for $\mathcal{C}$-recognizable sources by using a mildly hard function for $\mathcal{C}$ as a black box. By exploiting this reduction, we give polynomial-time, seedless randomness extractors for three natural recognizable sources: (1) constant-degree algebraic sources over any prime field, where constant-degree algebraic sources are uniform distributions over the set of zeros of a system of constant degree polynomials; (2) sources recognizable by randomized multiparty communication protocols of cn bits, where $c > 0$ is a small enough constant; (3) halfspace sources, or sources recognizable by linear threshold functions. In particular, the new extractor for each of these three sources has linear output length and exponentially small error for min-entropy $k \geq (1 - \alpha)n$, where $\alpha > 0$ is a small enough constant.

Our second method shows that a seed-extending pseudorandom generator with exponentially small error for $\mathcal{C}$ yields an extractor with exponentially small error for $\mathcal{C}$-recognizable sources, improving a reduction by Kinne, Melkebeek, and Shaltiel [16]. Using the hardness of the parity function against AC^0 [13], we significantly improve Shaltiel's extractor [25] for AC^0-recognizable sources. Finally, assuming sufficiently strong one-way permutations, we construct seedless extractors for sources recognizable by BPP algorithms, and these extractors run in quasi-polynomial time.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization; Theory of computation

Keywords and phrases Extractor, Pseudorandomness

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.72

Category RANDOM

Related Version https://eccc.weizmann.ac.il/report/2018/110/

Funding Supported by NSF Grant CCF-1526952, NSF Grant CCF-1705028, and a Simons Investigator Award (#409864, David Zuckerman).

Acknowledgements We wish to thank Salil Vadhan, Ronen Shaltiel, Avishay Tal, and William Hoza for helpful discussions and comments.

1 Introduction

Randomness is needed for many applications, such as statistics, algorithms and cryptography. However, most physical sources are not truly random, in the sense that they can have substantial biases and correlations. Weak random sources can also arise in cryptography when an adversary can learn partial information about a uniformly random string.


© Fu Li and David Zuckerman;
licensed under Creative Commons License CC-BY

A natural approach to dealing with weak random sources is to apply a randomness extractor – a function that transforms a weak random source into an almost-perfect random source. However, it is impossible to give a single function that extracts even one bit of randomness from sufficiently general classes of sources [24]. There are two ways to combat this. One is to extract with the help of another short random string. An object constructed in this manner is called a seeded extractor [21]. The focus of this paper is the second way: to extract from more structured sources (without using additional random bits). Such a function is called a seedless, or deterministic, extractor.

More formally, a random source X is modeled as a probability distribution over n bit strings with some entropy k. In the context of randomness extraction, the standard measure of entropy is the so called min-entropy – the min-entropy k of a source X is defined as $H_\infty(X) = \min_s(\log(1/\Pr[X = s]))$. Then, the definition of a seedless extractor can be presented as follows.

▶ **Definition 1** (Seedless extractors for structured sources)**.** *Let $\mathcal{D}$ be a class of distributions over $\{0,1\}^n$. We say a function $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ is a (k,ϵ)-extractor for $\mathcal{D}$ if for any distribution $D \in \mathcal{D}$ with min-entropy at least k, we have*

$$\mathsf{Ext}(D) \approx_\epsilon U_m,$$

where U_m denotes the uniform distribution over $\{0,1\}^m$ and $\approx_\epsilon$ stands for ϵ-close in statistical distance (Definition 16).

By the probabilistic method, it is known that for any constant $\alpha > 0$ and any distribution family $\mathcal{D}$ of at most $2^{2^{(1-\alpha)k}}$ sources of min-entropy k, there is a seedless extractor outputting $m = (1-\alpha)k$ bits with error $2^{-\alpha k/3}$.

A large body of research has been devoted to constructing explicit seedless extractors for various structured sources. There are mainly two natural perspectives to limit the structure of a distribution: an algebraic perspective and a computational perspective.

The algebraic perspective is to impose some algebraic structure on the distribution, such as an affine source [5]. Later, affine sources were generalized to distributions defined using low-degree polynomials. On one hand, Dvir, Gabizon and Wigderson [10] studied polynomial sources, which are the images of low-degree polynomial maps. On the other hand, viewing an affine source as the kernel, or set of zeros, of an affine mapping, Dvir [9] introduced the class of sources sampled uniformly from kernels or sets of common zeros of one or more polynomials, which he called algebraic sources[1].

The computational perspective is to assume a distribution has "low complexity". This started with Trevisan and Vadhan [27], who considered distributions that can be sampled by efficient algorithms. They showed that constructing a seedless extractor for this class is closely related to proving lower bound for circuits and gave a conditional construction of such an extractor based on lower bound assumptions. Later, in [15], an unconditional extractor was constructed for sources generated by space-bounded algorithms. More recently, Viola [29] constructed a seedless extractor for AC^0-samplable sources.

[1] For clarification, in [9], Dvir mentioned sources which are distributed uniformly on varieties. A variety is also a set of common zeros of one or more polynomials, but it is often defined to require the ground field to be algebraically closed.

1.1 Recognizable sources

We focus on recognizable sources, first suggested by Shaltiel [25]. Recognizable sources are uniform distributions over sets of the form $\{x : f(x) = v\}$ for functions f coming from some specified class. Formally, for any boolean function $f : \{0,1\}^n \to \{0,1\}$, define the source recognizable by f, denoted by U_f, as the uniform distribution over $f^{-1}(1)$. For short, we call this distribution the f-recognizable source. For any boolean function family $\mathcal{C}$, the set of $\mathcal{C}$-recognizable sources is the set of f-recognizable sources, for each $f \in \mathcal{C}$.

This notion naturally interacts with the algebraic and computational perspectives to limit the structure of a distribution, and also captures several distributions that were widely studied. For example, distributions with algebraic structures are those distributions recognizable by algebraic classes – affine sources are distributions recognizable by affine functions and algebraic sources are distributions recognizable by products of low-degree polynomials. Moreover, distributions that have "low complexity" could also be the distributions recognizable by low-complexity classes, such as small circuits.

Shaltiel [25] initially proposed an extractor for recognizable sources. He showed that such extractors produced randomness that was in some sense not correlated with the input and hence could be used for derandomization. In particular, to derandomize any class of randomized algorithms, he needed to explicitly construct an extractor for distributions recognizable by the class. He showed that without further changes, some appropriate known extractors could work for distributions recognizable by decision trees, streaming algorithms, and AC^0. What's more, assuming average-case hardness against polynomial-size circuits, he showed that applying the hard function on disjoint blocks of the input was an extractor for distributions recognizable by general polynomial-time algorithms.

Later, Kinne, Melkebeek and Shaltiel [16] improved the derandomization results in [25] by using "seed-extending pseudorandom generators", which are pseudorandom generators that reveal their seed. They gave reductions between seed-extending PRGs and extractors for recognizable sources. However, both Shaltiel [25] and this later paper [16] focused on derandomization rather than constructing new extractors.

1.2 XOR Amplification

Given a boolean function $f : \{0,1\}^n \to \{0,1\}$, let $f^{\oplus m}(x_1, \ldots, x_m) := \bigoplus_{i \in [m]} f(x_i)$ denote the XOR of m independent copies of f. The XOR Amplification Lemma[2] states that if a function f is hard on average for some computational class $\mathcal{C}$, (i.e., f cannot be computed correctly by any function in $\mathcal{C}$ on at most a $(1/2 + p)$-fraction of of the inputs), then $f^{\oplus m}$ cannot be computed correctly on at most a $(1/2 + p^{\Omega(m)})$-fraction of of the inputs. Loosely speaking, the hardness of f is amplified when the outputs of independent copies of f are XOR together. Indeed, this idea is analogous to the information theoretic setting. If f is a biased coin with $\Pr[f = 1] = 1/2 + p$, then the XOR of m independent biased coins, $f^{\oplus m}$, induces a coin with $\Pr[f^{\oplus m} = 1] = 1/2 - (-2p)^m/2$. However, showing that such an idea holds in the computational setting is significantly more involved.

There are several works dedicated to proving XOR amplification for computational models. Yao [31] first suggested XOR amplification, and proved that XOR (hardness) amplification held for polynomial-size circuits. Unfortunately, the amplification stops when XORing more than logarithmically many copies, which makes it not so useful for us. Later, Viola and

[2] This is usually called simply the XOR lemma, or Yao's XOR lemma, but we want to distinguish it from a different XOR lemma.

Wigderson [30] showed XOR amplification for multi-party communication complexity and polynomials over GF(2). Subsequently, their proof was extended by Bogdanov, Kawachi and Tanaka [4], to prove XOR amplification for polynomials over any prime field.

In this paper, we give a new application of XOR amplification – constructing seedless extractors for recognizable sources.

2 Overview of our results

2.1 From XOR amplification to Extractors for recognizable sources

It is folklore that one can use correlation bounds to extract a single bit. In this paper, we use XOR amplification to extend the output length from one bit to linear in the input length.

Intuitively, XOR amplification states that if a function f is hard on average for some complexity class $\mathcal{C}$ of Boolean functions, then $f^{\oplus m}(x_1, \ldots, x_m) = f(x_1) \oplus \cdots \oplus f(x_m)$ is exponentially harder on average. We actually only need a weaker condition: that there exists some h for which $h^{\oplus k}$ gets exponentially harder.

More precisely, let $\mathcal{C} \subseteq \{\{0,1\}^* \to \{0,1\}\}$ be a class of Boolean functions. For a positive constant α, we say $\mathcal{C}$ has α-XOR amplification if there exists a function $h : \{0,1\}^t \to \{0,1\}$ such that for any positive integer k, the correlation between $h^{\oplus k}$ and g is no more than $2^{-\alpha k}$, for any $g \in \mathcal{C}$.

We show that if $\mathcal{C}$ is closed under restrictions and $\mathcal{C}$ has α-XOR amplification, then there is an efficient extractor for $\mathcal{C}_n$-recognizable sources, where $\mathcal{C}_n$ denotes the set of all n-variate functions in $\mathcal{C}$.

▶ **Theorem 2.** *Let $\mathcal{C} \subseteq \{\{0,1\}^* \to \{0,1\}\}$ be any boolean function class closed under restrictions and α be any positive constant. If $\mathcal{C}$ has α-XOR amplification, then for any positive integer n, there is an explicit seedless $((1-\beta)n, 2^{-\Omega(\alpha n)})$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ for $\mathcal{C}_n$-recognizable sources, where $\beta = \Theta(\alpha) > 0$, $m = \Omega(\alpha n)$, and $\mathcal{C}_n$ denotes the set of all n-variate functions in $\mathcal{C}$.*

Our construction uses $h : \{0,1\}^t \to \{0,1\}$ from the definition of XOR amplification. Since the function h is fixed, its input length t is a constant, and it is computable efficiently (by hardwiring it). We also use the generator matrix M of an asymptotically good $[l, m, r]$-code, where $l = n/t$, so the distance $r = \Omega(l) = \Omega(n/t)$. Then $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ is simply

$$\mathsf{Ext}(x) = h^{(l)}(x)M, \text{ where } h^{(l)}(x = (x_1, \ldots, x_l)) = (h(x_1), \ldots, h(x_l)).$$

Occasionally we will apply a variation of this theorem when t grows with n, in which case we need h to be computable in time polynomial in n. For example, if the input length of h is $t = O(\log n)$, then h should be computable in exponential time.

Li [18] uses a similar construction to extend the output length of two-source extractors from one bit to more. Raz [22] had a related but different way to extend the output length of his specific extractor using small biased spaces. Raz uses the XOR lemma, but his method would not work with any asymptotically good code (as ours and Li's does); in fact, Raz does not even mention codes or distance.

2.1.1 Algebraic sources

An algebraic set is a set of common zeros of one or more multivariate polynomials defined over a finite field $\mathbb{F}$. An *algebraic source* is a random variable distributed uniformly over an algebraic set, which was originally introduced by Dvir [9]. Algebraic sources are a natural

generalization of affine sources that have been widely studied. Furthermore, we say that an algebraic source has degree d if the algebraic source can be defined by polynomials of degree at most d.

▶ **Definition 3** (Algebraic extractor). *We say that* $\mathsf{Ext} : \mathbb{F}^n \to \mathbb{F}^m$ *is a* (k, d, ϵ)*-algebraic extractor over* $\mathbb{F}$ *if for any degree-*d *algebraic source* U_V *with* $|V| \geq |\mathbb{F}|^k$, $\mathsf{Ext}(U_V) \approx_\epsilon U_m$.

Dvir obtained explicit extractors for degree-d algebraic sources with entropy rate greater than $1/2$ over moderately sized fields, where $|\mathbb{F}| = \text{poly}(d)$, and with small entropy rate over large fields, where $|\mathbb{F}| = d^{\Omega(n^2)}$.

Golovnev and Kulikov [12] related the study of Boolean dispersers for quadratic algebraic sets to improving circuit lower bounds. A disperser is a relaxation of an extractor, which is only required to output a non-constant bit from a weak random source. They posed the open question of constructing a disperser for any algebraic set of size $2^{0.03n}$ and defined by using at most $1.78n$ quadratic polynomials. Such a disperser yields a new circuit lower bound.

Nevertheless, to our knowledge, there were only two papers on explicitly constructing dispersers or extractors for algebraic sources over GF(2). Cohen and Tal [8] constructed an extractor for algebraic sources defined by at most $(\log \log n)^{1/(2e)}$ quadratic polynomials. They also constructed dispersers for algebraic sources defined by at most n^α polynomials of degree at most $\log^{0.1}(n)$ for some constant $\alpha < 1$. Our extractor construction subsumes both their extractor and disperser, outputting n^γ random bits for algebraic sources with higher degree $c \log n$ and the same bound n^α for the number of defining polynomials, where γ, c are constants. Remscrim [23] constructed the best extractors before our work, outputting one bit with error $O(1/\sqrt{n})$ for min-entropy $n - n^c$ for any $c < 1/2$. It can handle fairly large degree, up to $n^{1/2-\alpha}$, where $\alpha > 0$ is a constant. Our construction significantly improves the extractor for constant-degree algebraic sources, outputting more bits and handling lower min-entropy.

Using Theorem 2, we construct a seedless extractor for algebraic sources of constant degree for some linear min-entropy. In particular, the new extractor has linear output length and exponentially small error for min-entropy $k \geq (1 - \alpha)n$, where $\alpha > 0$ is a small enough constant.

▶ **Theorem 4.** *For any positive integer* d*, there is an efficient* $\left((1 - 1/c_d)n, d, 2^{-\Omega(n/c_d)}\right)$*-algebraic extractor* $\mathsf{Ext} : \mathbb{F}_2^n \to \mathbb{F}_2^m$*, where* $c_d = \Theta(d^2 4^d)$*,* $m = \Omega(n/c_d)$.

Even for degree $c \log n$ for a small enough constant $c > 0$, our extractor outputs n^γ bits with error $2^{-\Omega(n^\alpha)}$ for $n - n^\alpha$ min-entropy, where $\gamma, \alpha > 0$ are some constants.

We can extend our algebraic extractor to any prime field $\mathbb{F}_q$.

▶ **Theorem 5.** *For any positive integer* d *and any prime field* $\mathbb{F}_q$*, there is an efficient* $\left((1 - 1/c_{d,q})n, d, q^{-\Omega(n/c_{d,q})}\right)$*-algebraic extractor* $\mathsf{Ext} : \mathbb{F}_q^n \to \mathbb{F}_q^m$*, where* $c_{d,q} = \Theta(d^2 2^{2d} q^3 \log q)$*,* $m = \Omega(n/c_{d,q})$.

2.1.2 Sources recognizable by communication protocols

We consider a boolean function class that has low communication complexity. Communication complexity was defined by Yao [32], who introduced a standard 2-party communication model. Later, Chandra, Furst, and Lipton [6] generalized this to the multiparty model. In a t-party communication NOF (number-on-forehead) model, each party holds a separate input and each party knows all but its own input. These parties attempt to compute (or approximate) a given function of these t inputs by exchanging few bits of communication. The

complexity of a communication protocol is the number of bits exchanged on the worst input. Both deterministic and randomized communication protocols are considered. A randomized protocol can be viewed as a distribution on deterministic protocols.

For deterministic 2-party protocols, Shaltiel [25] already constructed an efficient extractor that has linear output for linear min-entropy and exponentially small error. To do this, he proved that 2-source extractors are also extractors for sources recognizable by deterministic 2-party protocols, and hence some known constructions of 2-source extractors could be used. However, this approach is tailored to the 2-party case and does not generalize to the t-party case for some $t > 2$.

We construct an extractor for sources recognizable by randomized t-party protocols. Formally, we prove the following theorem.

▶ **Theorem 6.** *There exists an explicit seedless $((1 - 1/c_t)n,\ 2^{-c_1 n/c_t})$ extractor $\mathsf{Ext} : \left(\{0,1\}^{n/t}\right)^t \to \{0,1\}^{c_2 n/c_t}$ for sources recognizable by randomized t-party communication protocols of at most $c_3 n/4^t$ bits, where $c_t = \Theta(t4^t)$ and c_1, c_2, c_3 are some positive constants.*

This extractor has linear output for linear min-entropy and exponentially small error, and is simply $\mathsf{Ext}(x) = \left(\wedge_t^{(l)}(x)\right) M$, where $l = n/t$, $\wedge_t$ is the AND function over t variables and M is the $l \times (c_2 n/c_t)$ generator matrix of a good linear code.

2.1.3 Halfspace sources

Halfspace sources are sources recognizable by linear threshold functions. A linear threshold function (abbreviated LTF) is a boolean function $f : \{0,1\}^n \to \{0,1\}$ that can be represented as $f(x) = 1_{\sum_{i \in n} a_i x_i > a_0}$ for some constants $a_0, a_1, \ldots, a_n \in \mathbb{R}$. From a geometric perspective, a boolean LTF is a halfspace-indicator to the discrete cube $\{0,1\}^n$.

We construct an efficient extractor that has linear output for linear min-entropy and exponentially small error for halfspace sources.

▶ **Theorem 7.** *There exists an explicit seedless $((1 - c_1)n,\ 2^{-c_2 n})$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{c_3 n}$ for halfspace sources, where c_1, c_2, c_3 are some positive small enough constants.*

The construction of this extractor is simply $\mathsf{Ext}(x) = \left(\wedge_2^{(l)}(x)\right) M$, where $l = n/2$, M is the $l \times c_3 n$ generator matrix of a good linear code.

2.2 From Seed-extending PRGs to Extractors for recognizable sources

The Kinne et al. reductions between seed-extending pseudorandom generators and extractors for recognizable distributions were asymmetric. They showed that an extractor with exponentially small error yielded a seed-extending pseudorandom generator with exponentially small error. However, they proved a weak converse.

In this paper, we prove that a seed-extending pseudorandom generator with exponentially small error yields an extractor with exponentially small error. This applies to flip-invariant families of boolean functions, which are invariant under flipping input bits (see Definition 26).

▶ **Theorem 8.** *Let $\mathcal{C}$ be a flip-invariant family of boolean functions over n bits. If G is a seed-extending (d, ϵ)-pseudorandom generator $G : \{0,1\}^d \to \{0,1\}^n$ for $\mathcal{C}$, then for any $\Delta = \Delta(n) > 0$ we can construct an $(n - \Delta, 2^{\Delta}\epsilon)$-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-d}$ for $\mathcal{C}$-recognizable sources. Specifically, if $G(x) = (x, E(x))$ fools any function in $\mathcal{C}$, then $\mathsf{Ext}(x \circ y) = y \oplus E(x)$ is an $(n - \Delta, 2^{\Delta}\epsilon)$-extractor for $\mathcal{C}$-recognizable sources, where $x \in \{0,1\}^d, y \in \{0,1\}^m$, where $m = n - d$.*

In particular, the reduction in [16] requires a tiny $\epsilon \leq 2^{-(m+2\Delta)}$ for the seed-extending PRG to get an $(n-\Delta, 2^{-\Delta})$-extractor. Moreover, the reduction in [16] breaks down for a seed-extending PRG, $G(x) = (x, E(x))$, where $E(x)$ is longer than x. We improve the reduction from seed-extending PRGs to extractors to require only $\epsilon \leq 2^{-2\Delta}$, without depending on the output length m. Furthermore, the new reduction can still work even for a seed-extending PRG, $G(x) = (x, E(x))$, where $E(x)$ is longer than x.

Based on this new reduction, we significantly improve extractors for two important types of recognizable sources as follows.

2.2.1 Circuit-recognizable sources

Kinne et al. proved that the well-known Nisan-Wigderson pseudorandom generator construction [20] can be made seed-extending. Therefore, assuming hardness against small circuits, we can construct an extractor for sources recognizable by small circuits.

▶ **Proposition 9.** *For any $\Delta = \Delta(n) > 0$ and positive integers $l < n$, if there is a function H that is ϵ-hard at input length $\sqrt{l}/2$ for circuits of size $s + (n-l)2^{O(\log(n-l)/\log l)}$ and depth $d+1$, then we can get an $(n-\Delta, (n-l)2^{\Delta}\epsilon)$-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-l}$ for any sources recognizable by circuits of size s and depth d.*

Using the hardness of the parity function against AC^0 [13], we significantly improve Shaltiel's extractor [25] for AC^0-recognizable sources.

▶ **Theorem 10.** *For any $\Delta = \Delta(n) > 0$ and positive integers $l < n$, there exists a polynomial time computable $(n-\Delta, (n-l)2^{\Delta - \Omega(l^{1/(2d+2)})})$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-l}$ for any sources recognizable by circuits of size $2^{n^{1/d}}$ and depth d.*

In particular, for min-entropy $n - n^{1/(\alpha d)}$, our extractor outputs $n - n^{2/\alpha + O(1/d)}$ bits, whereas Shaltiel's extractor outputs only $n^{1/(\alpha d)}$ bits. When $\alpha > 2d/(d-1)$ is a large enough constant, our extractor outputs $n - o(n)$ bits whereas Shaltiel's extractor outputs only $n^{1/(\alpha d)}$ bits. For min-entropy $n - \text{polylog}(n)$ bits, our extractor outputs $n - \text{polylog}(n)$, whereas Shaltiel's extractor outputs only $\text{polylog}(n)$ bits.

Our methods also apply to formulas. Komargodski, Raz and Tal [17] constructed an explicit function $h : \{0,1\}^n \to \{0,1\}$ that is $2^{-\Omega(r)}$-hard for any deMorgan formula of size $n^{3-o(1)}/r^2$. Based on this hardness result, we can construct an efficient extractor for sources recognizable by deMorgan formulas of size close to $n^{3/2}$.

▶ **Theorem 11.** *For any $\Delta, r, \alpha > 0$ and $m \leq (1-\alpha)n$, there exists a polynomial time computable $(n-\Delta, m2^{\Delta - \Omega(r)})$-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ for any sources recognizable by deMorgan formulas of size $n^{3/2-o(1)}/r^2$.*

2.2.2 Sources recognizable by efficient randomized algorithms

Note that there are no efficient seed-extending cryptographic PRGs. Otherwise, with revealed seeds, it is easy to efficiently distinguish the output of an efficient seed-extending PRG, $G(x) = (x, E(x))$, from a random string (x, y), by checking whether y equals $E(x)$.

We show that there is an inefficient seed-extending cryptographic PRG implied by the existence of one-way permutations. By our reduction, we show that a one-way permutation with exponentially small error yields an $(n - n^{\Omega(1)}, 2^{-n^{\Omega(1)}})$ extractor extracting $n - n^{O(1)}$ bits from sources recognizable by BPP algorithms. Formally, this follows by taking $\epsilon = 2^{-cn^{\alpha}}$ and $q(n) = n^{w(1)}$ in the following theorem.

▶ **Theorem 12.** *For any polynomial-time computable functions $t(\cdot)$ and $\epsilon(\cdot)$, assume that $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way permutation with error $\epsilon(\cdot)$ against $t(\cdot)$-bounded inverters. Then for any $\Delta = \Delta(n) > 0$ and a positive constant $\delta < 1$, we can construct an $\left(n - \Delta, O\left(2^{\Delta}\epsilon(n^{\delta})^{c_\delta}\right)\right)$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-n^{\delta}}$ for sources recognizable by randomized algorithms running in time $\left(t(n^{\delta})\right)^{c_\delta}$, where c_δ is a constant depending on δ. The running time of the extractor is a polynomial times the time to compute the inverse function f^{-1} of the one-way permutation f with input length n^{δ}. Due to the space limitation, we prove the following theorem in the full version of this paper.*

Furthermore, the running time of such an extractors will be quasi-polynomial if there exists a sufficiently strong one-way permutations. In particular, by scaling down, we have the following corollary.

▶ **Corollary 13.** *For any constants $a, b, c, \delta > 0$, assume that there exists a one-way permutation invertible in time $O\left(2^{n^a}\right)$ with error 2^{-n^c} against $2^{\delta n^b}$-bounded inverters, Then, for any positive constants α and $\beta < 1$, we can get an $\left(n - c_\beta \log^{c\alpha}(n), O\left(2^{-c_\beta \log^{c\alpha}(n)}\right)\right)$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-n^{\beta}}$ for sources recognizable by randomized algorithms running in time $2^{c_\beta \delta \log^{b\alpha}(n)}$, where c_β is a constant depending on β.The running time of the extractor is $O\left(2^{\log^{a\alpha}(n^{\beta})}\right)$.*

3 Overview of our main constructions and proofs

3.1 From XOR amplification to Extractors

In this subsection, we describe how to construct a seedless extractor for $\mathcal{C}$-recognizable sources if there exists a function $h : \{0,1\}^t \to \{0,1\}$ such that for any $g \in \mathcal{C}$ and $k \le n/t$, $Cor(h^{\oplus k}, g) \le 2^{-\Omega(k)}$. Think of $t = O(1)$.

We start with the statistical XOR lemma[3], usually attributed to Vazirani. We say a random variable Z over $\{0,1\}$ is ϵ-biased if $bias(Z) = Cor(Z, 0) = |\Pr[Z = 0] - \Pr[Z = 1]| \le \epsilon$.

▶ **Lemma 14** (Statistical XOR Lemma)**.** *Let $X_1, \ldots, X_m$ be 0-1 random variables such that for any nonempty $S \subseteq \{1, \ldots, m\}$, the random variable $\bigoplus_{i \in S} X_i$ is ϵ-biased. Then, the distribution of $(X_1, \ldots, X_m)$ is $\epsilon 2^{m/2}$-close to uniform.*

Let $g_i(x)$ be the i-th bit of $\mathsf{Ext}(x)$ for each $i \in [m]$. Thus, to show that the output of Ext is close to uniform, it suffices to show that for any non-empty set $S \subseteq [m]$, $g_S = \sum_{i \in S} g_i$ is low-biased conditioned on $f(x) = 1$ for each $f \in \mathcal{C}$. By XOR amplification, it is enough to guarantee that each g_S is the sum of $\Omega(n)$ independent copies of h, and hence g_S has $2^{-\Omega(n)}$ correlation with any function in $\mathcal{C}$.

A linear code is a natural candidate to guarantee that each g_S is the sum of $\Omega(n)$ independent copies. Let $h^{(l)} : \{0,1\}^{tl} \to \{0,1\}$ denote the concatenation of l copies of h and M be the generating matrix of an asymptotically good $[l, m, r]_2$ code. Our construction is simply

$$\mathsf{Ext}(x) = (g_1(x), \ldots, g_m(x)) = h^{(l)}(x)M.$$

Finally, we observe that the bias of g_S conditioned on $f(x) = 1$ can be bounded by the correlation between g_S and f plus the bias of g_S.

[3] The statistical XOR lemma is unrelated to the XOR amplification used in our proof.

▶ **Lemma 15.** $|\Pr[g_S(X)=1|f(X)=1]-\Pr[g_S(X)=0|f(X)=1]| \le \frac{Cor(g_S,f)+bias(g_S)}{2\Pr[f(X)=1]}$.

That is, if we choose a good linear code, then $\mathsf{Ext}(x) = h^{(l)}(x)M$ is an extractor for $\mathcal{C}$-recognizable sources with exponentially small error.

For details, see Section 5.

3.2 Algebraic extractors over GF(2)

In this subsection, we describe our algebraic extractor construction.

Notice that to construct a degree-d algebraic extractor that outputs only one bit, it is enough to let the extractor have small correlation bounds with degree-d polynomials. This fact is implicitly proved by Dvir [9] and observed by others, e.g., Eshan Chattopadhyay and Avishay Tal (personal communication). Based on this fact, we combine XOR amplification and linear codes to extend the output length form one bit to more.

First we observe that an algebraic source over n bits defined by n-variate polynomials $p_1,\dots,p_k$ is also a source recognizable by the product $\prod_{i\in[k]}(p_i+1)$. Let $\mathcal{V}_d$ denote the set of all products of polynomials of degree at most d. Thus, for any positive integer n, to get an extractor for n-bit algebraic sources of degree d, it suffices to construct an extractor for $\mathcal{V}_d$-recognizable sources over n bits. In particular, by the previous discussion, it suffices to show that XOR amplification holds for $\mathcal{V}_d$.

Second we observe that to show that a function f has low correlations with $\mathcal{V}_d$, it suffices to show that f has low correlation with any d-degree polynomials. This is because the L1 norm of the Fourier transform of the AND function is at most 2.

Viola and Wigderson [30] proved XOR amplification for low-degree polynomials over GF(2). Specifically, if a Boolean function h over $\{0,1\}^{O(d)}$ has correlation at most $1-1/2^d$ with degree-d polynomials, then the correlation between $h^{\oplus l}$ (see Section 1.2) and degree-d polynomials drops exponentially with l. Such h are known.

For details, see Section A.1.

3.3 From seed-extending PRGs to Extractors

We start with a new reduction from pseudorandom generators to seedless extractors. Observe that a seedless extractor $\mathsf{Ext}:\{0,1\}^n\to\{0,1\}^m$ partitions $\{0,1\}^n$ as $\bigcup_{z\in\{0,1\}^m}\mathsf{Ext}^{-1}(z)$. If Ext is a (k,ϵ)-extractor for $\mathcal{C}$-recognizable sources, then for every $f\in\mathcal{C}$ with $|f^{-1}(1)|\ge 2^k$, most intersections $\mathsf{Ext}^{-1}(z)\cap f^{-1}(1)$ should have almost the same size. That is, for most m-bit strings z, the preimage $\mathsf{Ext}^{-1}(z)$ is an ϵ-pseudorandom set against any $f\in\mathcal{C}$ with $|f^{-1}(1)|\ge 2^k$.

Now, given PRGs, how do we construct extractors? From the above observation, converting an ϵ-pseudorandom set into a partition of ϵ-pseudorandom sets is a possible way. If each preimage $\mathsf{Ext}^{-1}(z)$ of Ext is an ϵ-pseudorandom set for $\mathcal{C}$, Ext should be an extractor for $\mathcal{C}$-recognizable sources with a bit worse parameters.

To make $\mathsf{Ext}^{-1}(z)$ an ϵ-pseudorandom set for each z, we need a seed-extending PRG $G(x)$, i.e., $G(x)=x\circ E(x)$ for some function $E:\{0,1\}^d\to\{0,1\}^{n-d}$. By linearly shifting the set $\{(x,E(x))\}$, we can partition $\{0,1\}^n$ as $\bigcup_{z\in\{0,1\}^{n-d}}\{(x,(E(x)\oplus z)):x\in\{0,1\}^d\}$. We therefore define $\mathsf{Ext}(x,z)=E(x)\oplus z$. Since $\mathcal{C}$ is a flip-invariant function family, we have that the set $\mathsf{Ext}^{-1}(z)=\{(x,(E(x)\oplus z)):x\in\{0,1\}^d\}$ fools any function f in $\mathcal{C}$.

For details, see Section 6.

3.4 Algebraic extractors over prime fields

We remark that the main results used in building our algebraic extractor over GF(2) – the XOR amplification, the statistical XOR lemma and the asymptotically linear code – all have been extended to prime fields. Thus, to generalize our algebraic extractor, the remaining technical parts are not hard.

Bogdanov, Kawachi and Tanaka [4] proved XOR amplification for low-degree polynomials over prime fields, i.e., the sum of k independent copies of h was $q^{-\Omega(k)}$-hard for P_d if h was mildly hard. However, besides the sum of copies, we require the same hardness result for linear combinations of k copies of h. We prove this hardness result by using the original proof of Bogdanov, Kawachi and Tanaka with some slight modifications. The main revision of our proof uses the fact that the Gowers norm is multiplicative for functions over disjoint sets of input variables.

Furthermore, over a prime field $\mathbb{F}_q$, an algebraic source over n bits defined by n-variate polynomials $p_1, \ldots, p_k$ is a source recognizable by the product $\prod_{i\in[k]}(1-p_i^{q-1})$. We need to analyze the product of the special form $\prod_{i\in[k]}(1-x_i^{q-1})$, as an analog of the AND function over GF(2).

The reason we assume prime fields in our results is that XOR amplification for polynomials is known only over prime fields.

For details, please check the full version of this paper.

4 Preliminaries

In the following, for any two binary strings x, y, let $x \circ y$ denote their concatenation, and let $x \oplus y$ denote their bitwise XOR when x and y have the same length.

▶ **Definition 16** (Statistical distance). *Let D_1 and D_2 be two distributions over a set S. Define the* statistical distance *between D_1 and D_2 as $|D_1 - D_2| = \frac{1}{2}\sum_{s\in S}|\Pr[D_1 = s] - \Pr[D_2 = s]|$. We say D_1 is ϵ-close to D_2, denoted by $D_1 \approx_\epsilon D_2$, if $|D_1 - D_2| \le \epsilon$.*

▶ **Definition 17** (Recognizable source). *For any boolean function $f : \{0,1\}^n \to \{0,1\}$, define the source recognizable by f, denoted by U_f, as the uniform distribution over $f^{-1}(1)$. For short, we call this distribution the f-recognizable source.*

For any boolean function family $\mathcal{C}$, the set of $\mathcal{C}$-recognizable sources is the set of f-recognizable sources for $f \in \mathcal{C}$.

For $l \in \mathbb{N}$, let U_l denote the uniform distribution on l bits.

▶ **Definition 18** (Extractor for recognizable sources [25]). *Let $\mathcal{C}$ be a class of functions $C : \{0,1\}^n \to \{0,1\}$. We say that $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ is a (k,ϵ)-extractor for $\mathcal{C}$-recognizable sources if for every $f \in \mathcal{C}$ such that $\left|f^{-1}(1)\right| \ge 2^k$, $\mathsf{Ext}(U_f) \approx_\epsilon U_m$.*

Note that when the output length $m = 1$, the extractor is simply a boolean function which has low correlation with any function in $\mathcal{C}$.

4.1 Algebraic sources

An algebraic set is a set of common zeros of one or more multivariate polynomials defined over a finite field $\mathbb{F}$.

▶ **Definition 19** (Algebraic set). *For any s polynomials $f_1, \ldots, f_s \in \mathbb{F}[x_1, \ldots, x_n]$, the set $V(f_1, \ldots, f_s) = \{x \in \mathbb{F}^n | f_i(x) = 0, \forall i \in [s]\}$ is an algebraic set. We say V is an algebraic set of degree d, if each polynomial f_i has degree at most d.*

An *algebraic source* is a random variable distributed uniformly over an algebraic set as initially defined by Dvir [9].

▶ **Definition 20** (Algebraic source). *An algebraic source is the uniform distribution U_V over an algebraic set V. If V is a degree-d algebraic set, then we say U_V is an algebraic source of degree d.*

▶ **Definition 21** (Algebraic extractor). *We say that $\mathsf{Ext} : \mathbb{F}^n \to \mathbb{F}^m$ is a (k, d, ϵ)-algebraic extractor if for any degree-d algebraic source U_V with $|V| \geq |\mathbb{F}|^k$, $\mathsf{Ext}(U_V) \approx_\epsilon U_m$.*

▶ **Definition 22** (Linear codes over prime fields). *For a prime q, a linear code of length n and dimension k is a k-dimensional linear subspace C of the vector space $\mathbb{F}_q^n$. If the distance of the code C is d, i.e., the minimum number of two codewords in which they differ, we say that C is an $[n, k, d]_q$ code. A family of codes $\{C_n\}$ is asymptotically good if there exist constants $0 < \delta_1, \delta_2 < 1$ s.t. $k \geq \delta_1 n$ and $d \geq \delta_2 n$.*

Note that every linear code has an associated generating matrix $M \in \mathbb{F}_q^{k \times n}$, and every codeword can be expressed as vM, for some vector $v \in \mathbb{F}_q^k$. There are explicit constructions of asymptotically good linear codes, such as the Justensen codes over GF(2) constructed in [14] and the expander codes over GF(q) in [1] for any prime q.

▶ **Definition 23** (Correlation over prime fields). *Let $f, g : \mathbb{F}_q^n \to \mathbb{F}_q$ be two functions over n inputs. The correlation between f and g with respect to the uniform distribution is defined as*

$$Cor(f, g) := |Ee_q[f(x) + g(x)]| \in [0, 1],$$

where $e_q[x] := w^x$ for $x \in \{0, 1, \ldots, q-1\}$, where w denotes the q-th root of unity.

For a class $\mathcal{C}$ of functions, we denote by $Cor(f, \mathcal{C})$ the maximum of $Cor(f, C)$ over all $C \in \mathcal{C}$ whose domain is the same as f.

Furthermore, when $q = 2$, we have $e_2[x] = (-1)^x$, and $Cor(f, g) = |\Pr[f(x) = g(x)] - \Pr[f(x) \neq g(x)]|$. We often write $e_2[x]$ as $e[x]$ for convenience.

▶ **Definition 24** ($f^{(m)}, f^v$). *For any function $f : \mathbb{F}_q^n \to \mathbb{F}_q$, let $f^{(m)}$ denote the concatenation of m copies of f, i.e., $f^{(m)}(x_1, x_2, \ldots, x_m) := (f(x_1), \ldots, f(x_m))$, where $x_1, \ldots, x_m \in \mathbb{F}_q^n$. For each $v = (v_1, \ldots, v_m) \in \mathbb{F}_q^m$, let f^v denote the linear combination of m copies of f according to v, i.e., $f^v(x_1, x_2, \ldots, x_m) := \sum_{i \in [m]} v_i f(x_i)$.*

Let $\mathbb{F}_q^* = \mathbb{F}_q \setminus \{0\}$ denotes the set of non-zero elements in $\mathbb{F}_q$. We remark that the statistical XOR lemma has been generalized to prime fields by e.g., Goldreich [11].

▶ **Lemma 25** (Statistical XOR Lemma over $\mathbb{F}_q$). *Let $X = (X_1, \ldots, X_m)$ be random vector over $\mathbb{F}_q^m$ such that for any nonzero vector $v = (v_1, \ldots, v_m) \in \mathbb{F}_q^m \setminus \{0^m\}$, the random variable $v \cdot X = \sum_{i \in [m]} v_i X_i$ is ϵ-biased. Then, the distribution of $(X_1, \ldots, X_m)$ is $\epsilon q^{m/2}$-close to the uniform distribution over $\mathbb{F}_q^m$.*

For example, when $m = 1$, for a random variable X over $\mathbb{F}_q$, to show that $X \approx_\epsilon U_{\mathbb{F}_q}$, we need to show that $bias(\alpha X) \leq \epsilon/\sqrt{q}$ for each $\alpha \in \mathbb{F}_q^*$.

4.2 Seed-extending PRGs

▶ **Definition 26** (Flip-invariant family). *We say a boolean function family $\mathcal{C}$ over n bits is* flip-invariant *if for any string $s \in \{0, 1\}^n$, $f \in \mathcal{C}$ implies $f(x \oplus s) \in \mathcal{C}$.*

▶ **Definition 27** (Seed-extending pseudorandom generator). *A seed-extending pseudorandom generator is a generator G that outputs the seed as part of the pseudorandom string.*

Formally, a seed-extending (d, ϵ)-pseudorandom generator $G : \{0,1\}^d \to \{0,1\}^n$ for a class of functions over n bits, is a seed-extending function, i.e., $G(s) = (s, E(s))$ for some function E, such that

$$|\Pr[f(G(U_d)) = 1] - \Pr[f(U_n) = 1]| \leq \epsilon.$$

5 From XOR Amplification to Extractors for Recognizable Sources

First we define XOR amplification for a boolean function class that contains functions with various input lengths. Recall that $f^{\oplus m}(x_1, \ldots, x_m) = \bigoplus_{i \in [m]} f(x_i)$.

▶ **Definition 28** (α-XOR amplification for a boolean function class). *Let $\mathcal{C} \subseteq \{\{0,1\}^* \to \{0,1\}\}$ be a class of boolean functions. For a positive constant α, we say $\mathcal{C}$ has α-XOR amplification if there exists a function $h : \{0,1\}^t \to \{0,1\}$ such that for any positive integer k, $Cor(h^{\oplus k}, g) \leq 2^{-\alpha k}$, for any $g \in \mathcal{C}$.*

However, for constructing extractors for n-bit recognizable sources, we need to focus on the specific subset $\mathcal{C}_n \subseteq \mathcal{C}$ that contains all n-variate functions in $\mathcal{C}$. We define XOR amplification for $\mathcal{C}_n$ to also allow fixing some input bits.

▶ **Definition 29** ((α, w)-XOR amplification for functions with a fixed input length). *For a set $\mathcal{C}_n$ of n-variate functions $C : \{0,1\}^n \to \{0,1\}$ and a positive constant α, we say $\mathcal{C}_n$ has (α, w)-XOR amplification for a function $h : \{0,1\}^t \to \{0,1\}$ if for any vector $v \in \{0,1\}^{\lfloor n/t \rfloor}$ with at least w ones, $Cor(h^v, \mathcal{C}_n) \leq 2^{-\alpha w}$, where we add dummy variables to the input of h^v if h^v has less than n input variables.*

Moreover, we say C_n has α-XOR amplification for h, if C_n has (α, w)-XOR amplification for h for each positive integer $w \leq \lfloor n/t \rfloor$.

Note that if $\mathcal{C}$ is closed under restrictions, the fact that $\mathcal{C}$ has α-XOR amplification implies that $\mathcal{C}_n$ has also α-XOR amplification for every positive integer n. Formally,

▶ **Lemma 30.** *Let $\mathcal{C} \subseteq \{\{0,1\}^* \to \{0,1\}\}$ be a class of boolean functions closed under restrictions. Let $\mathcal{C}_n \subseteq \mathcal{C}$ denote the set of all n-variate functions in $\mathcal{C}$. If $\mathcal{C}$ has α-XOR amplification for a function $h : \{0,1\}^t \to \{0,1\}$, then $\mathcal{C}_n$ has also α-XOR amplification for h for every positive integer n.*

Proof. Assume that $\mathcal{C}$ has α-XOR amplification for a function $h : \{0,1\}^t \to \{0,1\}$, i.e., $Cor(h^{\oplus k}, \mathcal{C}) \leq 2^{-\alpha k}$ for each positive integer k. Then, we need to prove that for every positive integer n, $\mathcal{C}_n$ has also α-XOR amplification for h. In particular, fix n and let $l = \lfloor n/t \rfloor$. It suffices to prove that for any vector $v \in \{0,1\}^l$ with k ones, $Cor(h^v, \mathcal{C}_n) \leq Cor(h^{\oplus k}, \mathcal{C})$, as $Cor(h^{\oplus k}, \mathcal{C}) \leq 2^{-\alpha k}$.

To prove this, without loss of generality, assume that the first k coordinates of v are all 1's, and the remaining coordinates are all 0's. Thus, h^v depends only on the first kt variables. For any n-variate function $C(x_1, \ldots, x_n) \in \mathcal{C}_n$,

$$\begin{aligned} Cor(h^v, C) &= E_{X\sim U_{kt}, Y\sim U_{n-kt}} e[h^v(X,Y) + C(X,Y)] \\ &= E_{Y\sim U_{n-kt}} \left[E_{X\sim U_{kt}} e[h^v(X,Y) + C(X,Y)]\right] \\ &\le \frac{1}{2^{n-kt}} \sum_{Y_0\in\{0,1\}^{n-kt}} Cor\left(h^{\oplus k}(X), C(X,Y_0)\right) \\ &\le \frac{1}{2^{n-kt}} \sum_{Y_0\in\{0,1\}^{n-kt}} Cor(h^{\oplus k}, \mathcal{C}) \\ &= Cor(h^{\oplus k}, \mathcal{C}). \end{aligned}$$

The last inequality follows since $\mathcal{C}$ is closed under restrictions, i.e., $C(X, Y_0) \in \mathcal{C}$ for any $Y_0 \in \{0,1\}^{n-kt}$. ◀

▶ **Theorem 31.** *Let $\mathcal{C}_n$ be a family of boolean functions over n bits containing the constant function $f(x) = 0$. For any positive integers n, m, t, let M be the $l \times m$ generating matrix of an asymptotically good $[l, m, r_0]_2$ code, where $l = n/t$. Assume that $\mathcal{C}_n$ has (α, r)-XOR amplification for $h : \{0,1\}^t \to \{0,1\}$, where $r \le r_0$. Then, the function $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$,*

$$\mathsf{Ext}(x) = h^{(l)}(x)M,$$

is an $(n - \Delta, 2^{m/2+\Delta-\alpha r})$ extractor for $\mathcal{C}_n$-recognizable sources.

Proof. For convenience, let $(g_1(x), \ldots, g_m(x)) = h^{(l)}(x)M$. To show that the output of Ext is $2^{m/2+\Delta-\alpha r}$-closed to the uniform, by the statistical XOR Lemma, it suffices to show that for any non-empty set $S \subseteq [m]$, $g_S = \sum_{i\in S} g_i$ is $2^{\Delta-\alpha r}$-biased conditioned on $f(x) = 1$ for any $f \in \mathcal{C}_n$ with $|f^{-1}(1)| \ge 2^{n-\Delta}$.

First we observe that the bias of g_S conditioned on $f(x) = 1$ can be bounded by the correlation between g_S and f plus the bias of g_S.

▶ **Lemma 32** (Lemma 15, restated)**.**

$$|\Pr[g_S(X) = 1 | f(X) = 1] - \Pr[g_S(X) = 0 | f(X) = 1]| \le \frac{Cor(g_S, f) + bias(g_S)}{2\Pr[f(X) = 1]}.$$

Proof. By multiplying $2\Pr[f(X) = 1]$ on both sides, it is equivalent to prove that

$$2\,|\Pr[g_S(X) = 1 \wedge f(X) = 1] - \Pr[g_S(X) = 0 \wedge f(X) = 1]| \le Cor(g_S, f) + bias(g_S).$$

Notice that

$$\begin{aligned} Cor(g_S, f) &= |\Pr[g_S(X) = f(X)] - \Pr[g_S(X) \ne f(X)]| \\ &= |\Pr[g_S(X) = 1 \wedge f(X) = 1] + \Pr[g_S(X) = 0 \wedge f(X) = 0] \\ &\quad - \Pr[g_S(X) = 0 \wedge f(X) = 1] - \Pr[g_S(X) = 1 \wedge f(X) = 0]|, \end{aligned}$$

and

$$\begin{aligned} bias(g_S) &= |\Pr[g_S(X) = 1] - \Pr[g_S(X) = 0]| \\ &= |\Pr[g_S(X) = 1 \wedge f(X) = 1] + \Pr[g_S(X) = 1 \wedge f(X) = 0] \\ &\quad - \Pr[g_S(X) = 0 \wedge f(X) = 1] - \Pr[g_S(X) = 0 \wedge f(X) = 0]|. \end{aligned}$$

Thus, by the triangle inequality,

$$\begin{aligned} bias(g_S) + Cor(g_S, f) &\geq |2\Pr[g_S(X) = 1 \wedge f(X) = 1] - 2\Pr[g_S(X) = 0 \wedge f(X) = 1]| \\ &= 2\left|\Pr[g_S(X) = 1 \wedge f(X) = 1] - \Pr[g_S(X) = 0 \wedge f(X) = 1]\right|. \end{aligned}$$ ◀

Then, observe that not only is each g_i a sum of at least r independent copies, but also so is any non-empty sum of the g_i, and hence has exponentially small correlation with degree-d polynomials.

▶ **Lemma 33.** *For any nonempty set $S \subseteq [m]$, $Cor(g_S, \mathcal{C}_n) \leq 2^{-\alpha r}$.*

Proof. Note that

$$g_S(x) = \sum_{i \in S} h^{(l)}(x) M_i = h^{(l)}(x) \left(\sum_{i \in S} M_i \right),$$

where M_i denotes the i-th row of the matrix M. As M is the generating matrix of an $[l, m, r]_2$ code and S is non-empty, $\sum_{i \in S} M_i$ is a codeword and hence has at least r 1's. Thus, g_S is the XOR of at least r_0 independent copies of h. By the assumed (α, r)-XOR amplification, we know $Cor(g_S, \mathcal{C}_n) \leq 2^{-\alpha r}$. ◀

Since the constant function $0 \in \mathcal{C}_n$, we also have that $bias(g_S) = Cor(g_S, 0) \leq 2^{-\alpha r}$. Thus, by Lemma 32, the bias of g_S conditioned on $f(x) = 1$ is at most $2^{-\alpha r}/p$, where $p = \Pr[f(X) = 1]$.

At last, we have $p = \frac{|f^{-1}(1)|}{2^n} \geq 2^{-\Delta}$ by the min-entropy requirement that $|f^{-1}(1)| \geq 2^{n-\Delta}$. Therefore, $g_S(x)$ is $2^{\Delta - \alpha r}$-biased conditioned on $f(x) = 1$. ◀

Combining with an explicit asymptotically good $[l, m, r]_2$ code, we prove the following theorem.

▶ **Theorem 34.** *Let $\mathcal{C} \subseteq \{\{0,1\}^* \to \{0,1\}\}$ be any boolean function class closed under restrictions and α be any positive constant. Let $\mathcal{C}_n$ denote the set of all n-variate functions in $\mathcal{C}$. If $\mathcal{C}_n$ has $(\alpha, \delta n)$-XOR amplification for $h : \{0,1\}^t \to \{0,1\}$, where $\delta < 1/t$ is a positive constant, then there is an explicit $(n - c_1\alpha l,\ 2^{-c_2\alpha l})$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{c_3\alpha l}$ for $\mathcal{C}_n$-recognizable sources, where $l = n/t$ and c_1, c_2, c_3 are some positive constants.*

Moreover, if $\mathcal{C}$ has α-XOR amplification for a function $h : \{0,1\}^t \to \{0,1\}$, then for any positive integer n, there is an explicit seedless $(n - c_1\alpha l,\ 2^{-c_2\alpha l})$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{c_3\alpha l}$ for $\mathcal{C}_n$-recognizable sources, where $l = n/t$ and c_1, c_2, c_3 are some positive constants.

Proof. Note that if $\mathcal{C}$ has α-XOR amplification for a function h, then by Lemma 30, $\mathcal{C}_n$ also has α-XOR amplification for h for every positive integer n, i.e., $\mathcal{C}_n$ also has $(\alpha, \delta l)$-XOR amplification for h by definition. Now, we start with the assumption that $\mathcal{C}_n$ has $(\alpha, \delta l)$-XOR amplification for h. We use an explicit $[l, \delta_1 l, \delta_2 l]_2$ linear code for some constants $\delta_1 > 0$ and $\delta_2 > \delta$ by Justesen [14]. Therefore, Theorem 31 yields an $(n - \Delta,\ 2^{m/2 + \Delta - \alpha\delta_2 l})$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ for $\mathcal{C}_n$-recognizable sources. That is, by setting $\Delta = c_1\alpha l$ and $m = c_3\alpha l$ for some small positive constants c_1, c_3, we get the desired $(n - c_1\alpha l,\ 2^{-c_2\alpha l})$ extractor, where $c_2 = -(c_3/2 + c_1 - \delta_2) > 0$ is also a positive constant. ◀

6 From Seed-Extending PRGs to Extractors for Recognizable Sources

Note that Kinne et al. [16] already showed reductions between extractors for recognizable sources and seed-extending PRGs.

▶ **Lemma 35** ([16, Theorem 7]). *Let $C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ be a function. Let $\Delta = m + \log(1/\epsilon)$ and let $E : \{0,1\}^n \to \{0,1\}^m$ be an $(n-\Delta, 2^{-\Delta})$-extractor for $\mathcal{C}$-recognizable distributions, where each function in $\mathcal{C}$ is of the form $f_r(x) = C(x,r)$ where $r \in \{0,1\}^m$ is an arbitrary string. Then, $G(x) = (x, E(x))$ is ϵ-pseudorandom for $\mathcal{C}$.*

▶ **Lemma 36** ([16, Theorem 8]). *Let $f : \{0,1\}^n \to \{0,1\}$ be a function and let $E : \{0,1\}^n \to \{0,1\}^m$ be a function such that $G(x) = (x, E(x))$ is ϵ-pseudorandom for tests $T(x,r)$ of the form $T_z(x,r) = f(x) \wedge (r = z)$ where $z \in \{0,1\}^m$ is an arbitrary string. For any $\Delta > 0$, if $\epsilon \le 2^{-(m+2\Delta)}$ then E is an $(n-\Delta, 2^{-\Delta})$-extractor for the distribution recognized by f.*

The Lemma 3.2 requires a tiny $\epsilon \le 2^{-(m+2\Delta)}$ for the seed-extending PRG to get an $(n-\Delta, 2^{-\Delta})$-extractor. In the following, we improve the reduction from seed-extending PRGs to extractors to require only $\epsilon \le 2^{-2\Delta}$. Moreover, our extractor is even stronger – the output of our extractor is close to uniform with relative error, which will be defined as follows.

▶ **Definition 37** (Statistical distance with relative error). *We say that a distribution Z on $\{0,1\}^m$ is ϵ-close to uniform with relative error if for every event $A \subseteq \{0,1\}^m$,*

$$|\Pr[Z \in A] - \mu(A)| \le \epsilon \cdot \mu(A), \text{ where } \mu(A) = |A|/2^m.$$

Note that if Z is ϵ-close to uniform with relative error, then it is also ϵ-close to uniform. Next we define extractors with relative error analogously.

▶ **Definition 38** (Seedless extractor with relative error, [2, Definition 1.19]). *Let $\mathcal{C}$ be a class of distributions over $\{0,1\}^n$. A function $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ is a (k,ϵ)-relative-error extractor for $\mathcal{C}$ if for every distribution X in the class $\mathcal{C}$ such that $H_\infty(X) \ge k$, $\mathsf{Ext}(X)$ is ϵ-close to uniform with relative error.*

We remark that the notions of statistical distance and extractors with relative error were introduced by Applebaum, Artemenko, Shaltiel, and Yang [2]. They translate relative-error extractors for distributions recognizable by small circuits into incompressible functions. However, parameters of our relative-error extractors are not strong enough to get incompressible functions.

Now we prove the reduction lemma from seed-extending PRGs to seedless extractors with relative error, which directly implies the reduction from seed-extending PRGs to seedless extractors.

▶ **Lemma 39.** *Let $\mathcal{C}$ be a flip-invariant family of boolean functions over n bits. If G is a seed-extending (d,ϵ)-pseudorandom generator $G : \{0,1\}^d \to \{0,1\}^n$, then we can construct an $(n-\Delta, 2^\Delta\epsilon)$-relative-error extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-d}$ as follows. If $G(x) = (x, E(x))$ fools any function in $\mathcal{C}$, then $\mathsf{Ext}(x \circ y) = y \oplus E(x)$ is an extractor for $\mathcal{C}$-recognizable sources, where $x \in \{0,1\}^d, y \in \{0,1\}^{n-d}$.*

For intuition, observe that a seedless extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ partitions $\{0,1\}^n$ as $\bigcup_{z\in\{0,1\}^m} \mathsf{Ext}^{-1}(z)$. If Ext is a (k,ϵ)-relative-error extractor for $\mathcal{C}$-recognizable sources, then for every $f \in \mathcal{C}$ with $|f^{-1}(1)| \ge 2^k$, all intersections $\mathsf{Ext}^{-1}(z) \cap f^{-1}(1)$ should have almost the same size. That is, for most m-bit strings z, the preimage $\mathsf{Ext}^{-1}(z)$ is an ϵ-pseudorandom set against any $f \in \mathcal{C}$ with $|f^{-1}(1)| \ge 2^k$.

Now, given PRGs, how to construct extractors? From the above observation, converting an ϵ-pseudorandom set into a partition of ϵ-pseudorandom sets is a possible way. If each preimage $\mathsf{Ext}^{-1}(z)$ of Ext is an ϵ-pseudorandom set for $\mathcal{C}$, Ext should be a relative-error extractor for $\mathcal{C}$-recognizable sources with a bit worse parameters, which will be precisely calculated in the following formal proof.

To make $\mathsf{Ext}^{-1}(z)$ an ϵ-pseudorandom set for each z, we need a PRG of the specific form: $G(x) = B(x) \circ E(x)$, for some bijection $B : \{0,1\}^d \to \{0,1\}^d$ and some function $E : \{0,1\}^d \to \{0,1\}^{n-d}$. By linearly shifting the set $\{(B(x), E(x))\}$, we can partition $\{0,1\}^n$ as $\bigcup_{z\in\{0,1\}^{n-d}} \{(B(x), (E(x) \oplus z)) : x \in \{0,1\}^d\}$. Since $\mathcal{C}$ is a flip-invariant function family, we have that the set $\mathsf{Ext}^{-1}(z) = \{(B(x), (E(x) \oplus z)) : x \in \{0,1\}^d\}$ fools any function f in $\mathcal{C}$.

Note that to convert the PRG of the form $(B(x), E(x))$ into an extractor, the above intuition gives $\mathsf{Ext}(x) = E(B^{-1}(x))$. Thus, to get an efficient extractor, we have to assume that $E(B^{-1}(x))$ can be efficiently computed. That is, the PRG of the form $(B(x), E(x))$ also gives an efficient seed-extending PRG $(x, E(B^{-1}(x)))$. Therefore, for constructing extractors from the above intuition, we only need to focus on the seed-extending PRGs.

Proof. For convenience, let $m = n - d$ denote the output length of Ext.

First, we observe that, for any fixed z, $G_z(x) = (x, (E(x) \oplus z))$ fools any function $f(x, y)$ in $\mathcal{C}$. Notice that to prove $G_z(x)$ fools $f(x, y)$, it is equivalent to prove $(x, E(x))$ fools $f(x, y \oplus z)$. Because of the flip-invariant property of $\mathcal{C}$, we know if $f(x, y) \in C$, then $f(x, y \oplus z) \in \mathcal{C}$. So $G(x) = x \circ E(x)$ fools $f(x, y \oplus z)$. That is, $G_z(x)$ fools the function $f(x, y)$.

Note that $\mathsf{Ext}^{-1}(z)$ is the range of G_z. Then, we can get

$$
\begin{aligned}
&\Pr[\mathsf{Ext}(X \circ Y) = z | f(X \circ Y) = 1] \\
=&\frac{\Pr[\mathsf{Ext}(X \circ Y) = z \wedge f(X \circ Y) = 1]}{\Pr[f(X \circ Y) = 1]} \\
=&\frac{\Pr[\mathsf{Ext}(X \circ Y) = z]}{\Pr[f(X \circ Y) = 1]} \Pr[f(X \circ Y) = 1 | \mathsf{Ext}(X \circ Y) = z] \\
=&\frac{\Pr[\mathsf{Ext}(X \circ Y) = z]}{\Pr[f(X \circ Y) = 1]} \Pr[f(G_z(X)) = 1] \\
=&\frac{\Pr[\mathsf{Ext}(X \circ Y) = z]}{\Pr[f(X \circ Y) = 1]} (\Pr[f(X \circ Y) = 1] \pm \epsilon) \\
=&\frac{p \pm \epsilon}{p} \Pr[\mathsf{Ext}(X \circ Y) = z], \text{ where } p = \Pr[f(X \circ Y) = 1], \\
=&\frac{p \pm \epsilon}{p} \frac{1}{2^m}.
\end{aligned}
$$

For any nonempty subset $S \subseteq \{0,1\}^m$, summing over all $z \in S$, we deduce that the output of Ext is $\frac{\epsilon}{p}\mu(S)$-close to the uniform distribution over S. Furthermore, we have $\frac{\epsilon}{p} \le 2^{\Delta}\epsilon$, since $p = \frac{|f^{-1}(1)|}{2^n} \ge 2^{-\Delta}$ by the min-entropy requirement that $|f^{-1}(1)| \ge 2^{n-\Delta}$. Therefore, $\mathsf{Ext}(x \circ y) = y \oplus E(x)$ is an $(n - \Delta, 2^{\Delta}\epsilon)$-relative-error extractor for $\mathcal{C}$-recognizable sources. ◀

References

1 Noga Alon, Jehoshua Bruck, Joseph Naor, Moni Naor, and Ron M Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on information theory*, 38(2):509–516, 1992.

2 Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions. *Computational complexity*, 25(2):349–418, 2016.

3 László Babai, Noam Nisant, and Márió Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, 1992.

4 Andrej Bogdanov, Akinori Kawachi, and Hidetoki Tanaka. Hard functions for low-degree polynomials over prime fields. *ACM Transactions on Computation Theory (TOCT)*, 5(2):5, 2013.

5 Jean Bourgain. On the construction of affine extractors. *GAFA Geometric And Functional Analysis*, 17(1):33–57, 2007.

6 Ashok K Chandra, Merrick L Furst, and Richard J Lipton. Multi-party protocols. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 94–99. ACM, 1983.

7 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

8 Gil Cohen and Avishay Tal. Two Structural Results for Low Degree Polynomials and Applications. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, page 680, 2015.

9 Zeev Dvir. Extractors for varieties. *Computational complexity*, 21(4):515–572, 2012.

10 Zeev Dvir, Ariel Gabizon, and Avi Wigderson. Extractors and rank extractors for polynomial sources. *Computational Complexity*, 18(1):1–58, 2009.

11 O Goldreich. Three XOR-Lemmas – An exposition, 1995.

12 Alexander Golovnev and Alexander S Kulikov. Weighted gate elimination: Boolean dispersers for quadratic varieties imply improved circuit lower bounds. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 405–411. ACM, 2016.

13 Johan Håstad. *Computational limitations of small-depth circuits*. MIT Press, 1987.

14 Jørn Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Transactions on Information Theory*, 18(5):652–656, 1972.

15 Jesse Kamp, Anup Rao, Salil Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. *Journal of Computer and System Sciences*, 77(1):191–220, 2011.

16 Jeff Kinne, Dieter van Melkebeek, and Ronen Shaltiel. Pseudorandom generators, typically-correct derandomization, and circuit lower bounds. *Computational complexity*, 21(1):3–61, 2012.

17 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved Average-Case Lower Bounds for De Morgan Formula Size: Matching Worst-Case Lower Bound. *SIAM Journal on Computing*, 46(1):37–57, 2017.

18 Xin Li. Improved two-source extractors, and affine extractors for polylogarithmic entropy. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 168–177. IEEE, 2016.

19 Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is Eighty*, 1:301–315, 1993.

20 Noam Nisan and Avi Wigderson. Hardness vs. randomness. In *Foundations of Computer Science, 1988., 29th Annual Symposium on*, pages 2–11. IEEE, 1988.

21 Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

22 Ran Raz. Extractors with weak random seeds. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 11–20. ACM, 2005.

23 Zachary Remscrim. The Hilbert Function, Algebraic Extractors, and Recursive Fourier Sampling. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 197–208. IEEE, 2016.

24 Miklos Santha and Umesh V Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33(1):75–87, 1986.

25 Ronen Shaltiel. Weak derandomization of weak algorithms: explicit versions of Yao's lemma. *Computational complexity*, 20(1):87, 2011.

26 Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 77–82. ACM, 1987.

27 Luca Trevisan and Salil Vadhan. Extracting randomness from samplable distributions. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 32–42. IEEE, 2000.

28 Emanuele Viola. Guest Column: correlation bounds for polynomials over {0 1}. *ACM SIGACT News*, 40(1):27–44, 2009.

29 Emanuele Viola. Extractors for circuit sources. *SIAM Journal on Computing*, 43(2):655–672, 2014.

30 Emanuele Viola and Avi Wigderson. Norms, XOR Lemmas, and Lower Bounds for Polynomials and Protocols. *Theory of Computing*, 4(1):137–168, 2008.

31 Andrew C Yao. Theory and application of trapdoor functions. In *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*, pages 80–91. IEEE, 1982.

32 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 209–213. ACM, 1979.

A Application of Theorem 2

A.1 Algebraic extractors over GF(2)

In this subsection, we will show that for any algebraic sources of constant degree over GF(2), there exists an efficient extractor that has linear output for linear min-entropy and exponentially small error. Formally, we will prove the following theorem:

▶ **Theorem 40.** *For any positive integer d, there is an efficient $\left((1-1/c_d)n, d, 2^{-\Omega(n/c_d)}\right)$-algebraic extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$, where $c_d = \Theta(d^2 4^d)$, $m = \Omega(n/c_d)$.*

Let P_d denote the set of all polynomials of degree at most d over GF(2). Let $\mathcal{V}_d$ denote the set of all products of polynomials in P_d and $\mathcal{V}_{d,n}$ denote the set of all products of n-variate polynomials in P_d.

Notice that an algebraic source of degree d over n bits is also a $\mathcal{V}_{d,n}$-recognizable source.

▶ **Lemma 41.** *An n-bit algebraic source of degree d iff it is a $\mathcal{V}_{d,n}$-recognizable source.*

Proof. Let U_V denote an arbitrary algebraic source, where $V = \{x \in \{0,1\}^n | p_i(x) = 0, p_i \in P_d, \forall i \in [k]\}$ is an algebraic set of degree d over n bits. Notice that V can be viewed as the set of 1-inputs of function $\prod_{i\in[k]}(p_i(x)+1)$. That is, the uniform distribution over V is also the source recognizable by $\prod_{i\in[k]}(p_i(x)+1) \in \mathcal{V}_{d,n}$. In other words, an algebraic source of degree d is a $\mathcal{V}_{d,n}$-recognizable source.

For the other direction, let U_f denote an arbitrary $\mathcal{V}_{d,n}$-recognizable source, where $f = \prod_{i\in[k]} p_i \in \mathcal{V}_{d,n}$ with $\deg(p_i) \le d$ for each $i \in [k]$. Note that $f^{-1}(1) = \{x \in \{0,1\}^n | p_i(x) = 1, \forall i \in [k]\} = \{x \in \{0,1\}^n | p_i(x) + 1 = 0, \forall i \in [k]\}$. Hence, $f^{-1}(1)$ is the algebraic set of $p_1(x)+1, \ldots, p_k(x)+1$. Since $\deg(p_i(x)+1) = \deg(p_i) \le d$ for each $i \in [k]$, $f^{-1}(1)$ is an algebraic set of degree d over n bits. Therefore, U_f is an n-bit algebraic source of degree d. ◀

Then, observe that $\mathcal{V}_d$ is closed under restrictions. Thus, by Theorem 31, to get an extractor for $\mathcal{V}_{d,n}$-recognizable sources, it is enough to show that $\mathcal{V}_d$ has α-XOR amplification for some positive constant α.

Note that to show that a function f has low correlations with $\mathcal{V}_d$, it suffices to show that f has low correlation with any polynomial of degree at most d. Recall that the correlation between a function f and a class $\mathcal{C}$ of functions is defined as the maximum of $Cor(f, C)$ over all $C \in \mathcal{C}$ whose input length is the same as f. In particular, to show that a function $f : \{0,1\}^t \to \{0,1\}$ has low correlations with $\mathcal{V}_d$, it suffices to show that f has low correlation with any t-variate polynomial of degree at most d.

▶ **Lemma 42.** *If a function $f : \{0,1\}^t \to \{0,1\}$ is ϵ-correlated with any polynomial of degree at most d in t variables, then f is at most 2ϵ-correlated with any product of polynomials of degree at most d in t variables.*

The lemma follows because the L1 norm of the Fourier transform of the AND function is at most 2.

Proof. We need to show that if for any t-variate $p \in P_d$ $Cor(f,p) = |Ee[f+p]| \leq \epsilon$, then for any product $\prod_{i\in[k]}(p_i+1) \in \mathcal{V}_{d,t}$ where $p_1+1, \ldots, p_k+1 \in P_{d,t}$, we have

$$Cor\left(f, \prod_{i\in[k]}(p_i(X)+1)\right) = \left|Ee\left[f + \prod_{i\in[k]}(p_i(X)+1)\right]\right| \leq 2\epsilon.$$

Consider the Fourier expansion of the function

$$e\left[\prod_{i\in[k]}(y_i+1)\right] = -\sum_{S\neq\emptyset}\frac{e\left[\sum_{i\in S} y_i\right]}{2^{k-1}} + (1-1/2^{k-1}).$$

Now, substituting each y_i by p_i, we have $e\left[\prod_{i\in[k]}(p_i+1)\right] = -\sum_{S\neq\emptyset}\frac{e\left[\sum_{i\in S} p_i\right]}{2^{k-1}} + (1-1/2^{k-1})$.

That is,

$$\left|Ee\left[f + \prod_{i\in[k]}(p_i(X)+1)\right]\right| \leq \sum_{S\neq\emptyset}\frac{\left|Ee\left[f+\sum_{i\in S} p_i(X)\right]\right|}{2^{k-1}}.$$

Notice that for each $S \neq \emptyset$, the sum $\sum_{j\in S} p_i$ is also a polynomial of degree at most d. For the polynomial of degree at most d, $\sum_{j\in S} p_i$, we have that $\left|Ee\left[f+\sum_{j\in S} p_i(X)\right]\right| \leq \epsilon$. In other words, $\left|Ee\left[f+\prod_{i\in[k]}(p_i(X)+1)\right]\right| \leq 2^k\frac{2\epsilon}{2^{k-1}} = 4\epsilon$. ◀

Moreover, Viola and Wigderson [30] proved XOR amplification for GF(2) polynomials, which implies XOR amplification for $\mathcal{V}_d$ by Lemma 42.

▶ **Theorem 43** ([30, Theorem 1.1])**.** *Let $h : \{0,1\}^n \to \{0,1\}$ be a function such that $Cor(h, P_{d,n}) \leq 1 - 1/2^d$. Then $Cor(h^{\oplus m}, P_d) \leq 2^{-\Omega(m/(4^d\cdot d))}$.*

Finally, by brute force search, it is easy to find a function h over $O(d)$ bits such that $Cor(h, P_d) \leq 1 - 1/2^d$ as d is a constant. That is, P_d has $\Omega(\frac{1}{4^d\cdot d})$-XOR amplification for the function $h : \{0,1\}^{O(d)} \to \{0,1\}$. This implies that V_d has $\Omega(\frac{1}{4^d\cdot d})$ -XOR amplification for the function $h : \{0,1\}^{O(d)} \to \{0,1\}$ by Lemma 42. Therefore, Theorem 34 yields our main theorem of this subsection, i.e., constructing an efficient $\left((1-1/c_d)n, d, 2^{-\Omega(n/c_d)}\right)$-algebraic extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$, where $c_d = \Theta(d^2 4^d)$, $m = \Omega(n/c_d)$.

We remark that an explicit example of h is the mod_3 function, which outputs 1 if and only if the number of input bits that are '1' is congruent to 1 modulo 3. Smolensky [26] proved that the mod_3 function over $O(d^2)$ bits is 2/3-hard for P_d (see Viola [28] for a proof), that is, P_d has $\Omega(\frac{1}{4^d \cdot d})$-XOR amplification for the function $\mathrm{mod}_3 : \{0,1\}^{O(d^2)} \to \{0,1\}$. Using the mod_3 function, Theorem 34 yields an efficient $\left((1-1/c'_d)n, d, 2^{-\Omega(n/c'_d)}\right)$-algebraic extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$, where $c'_d = \Theta(d^3 4^d)$, $m = \Omega(n/c'_d)$.

A.2 Sources recognizable by communication protocols

In this subsection, we construct an extractor for sources recognizable by randomized t-party protocols. Formally, we prove the following theorem.

▶ **Theorem 44.** *There exists an explicit seedless $((1-1/c_t)n,\ 2^{-c_1 n/c_t})$ extractor $\mathsf{Ext} : \left(\{0,1\}^{n/t}\right)^t \to \{0,1\}^{c_2 n/c_t}$ for sources recognizable by randomized t-party communication protocols of at most $c_3 n/c_t$ bits, where $c_t = \Theta(t4^t)$ and c_1, c_2, c_3 are some positive constants.*

Let $\mathcal{RCC}_{n,t,w}$ denote the class of n-variate randomized t-party protocols using at most w communication bits. Now, to construct extractors for $\mathcal{RCC}_{n,t,w}$-recognizable sources with exponentially small error, by Theorem 31, it suffices to show $\mathcal{RCC}_{n,t,w}$ has $(\alpha, r)-$XOR amplification for some function h, where $r = \Omega(n)$ is the distance of some good linear code.

Notice that, Babai, Nisan, and Szegedy [3] proved a lower bound for randomized t-party protocols for the Generalized Inner Product (GIP) function, which is the XOR of AND functions. Formally, let $\wedge_t : \{0,1\}^t \to \{0,1\}$ denote the AND function on t variables. Then, the GIP function $\mathrm{GIP}_{kt} : (\{0,1\}^t)^k \to \{0,1\}$ is defined as the function$\wedge_t^{\oplus k}$, i.e., $GIP_{kt}(x_1, \ldots, x_k) := \bigoplus_{i=1}^k \wedge_t(x_i)$. Moreover, let $R_{t,\epsilon}(f)$ denote the complexity of the best randomized t-party protocol correlating f with at least ϵ.

▶ **Theorem 45** ([3, Theorem 2])**.**

$$R_{t,\epsilon}(GIP_n) = \Omega\left(\frac{n}{4^t} - \log(1/\epsilon)\right).$$

Now, for any constant $0 < \delta < 1/t$ and some constant $c_t = \Theta(t4^t)$, we prove that $\mathcal{RCC}_{n,t,O(n/4^t)}$ has $(\Omega(1/c_t), \delta n)-$XOR amplification for $\wedge_t$, which directly yields Theorem 44 by Theorem 34.

▶ **Proposition 46.** *For any constant $0 < \delta < 1/t$, $\mathcal{RCC}_{n,t,c'n/4^t}$ has $(c/c_t, \delta n)-$XOR amplification for $\wedge_t$, where $c_t = \Theta(t4^t)$, $c, c' > 0$ are constants.*

Proof. Assume by contradiction that $\mathcal{RCC}_{n,t,c'n/4^t}$ does not have $(c/c_t, \delta n)-$XOR amplification for $\wedge_t$, where c, c' are some constants to be decided later. That is, there exists some vector $v \in \{0,1\}^{n/t}$ with at least δn ones, $Cor(h^v, \mathcal{RCC}_{n,t,c'n/4^t}) \le 2^{-\frac{c}{c_t}\delta n}$, That is, there exists a $(c'n/4^t)$-bit randomized protocol that approximates h^v within $2^{-\frac{c}{c_t}\delta n}$ error. Furthermore, observe that h^v is the XOR of at least δn copies of $\wedge_t$, i.e, h^v depends on $\ge \delta nt$ variables. Therefore, by Theorem 45, we have

$$R_{t,2^{-\frac{c}{c_t}\delta n}}(h^v) \ge R_{t,2^{-\frac{c}{c_t}\delta n}}(GIP_{\delta n}) = \Omega\left(\delta\frac{n}{4^t} - \frac{c}{c_t}\delta nt\right).$$

That is, letting the constant c be small enough, we know there exists a positive constant c'' such that

$$R_{t,2^{-\alpha\delta n}}(h^v) \ge c''n/4^t.$$

Now letting $c' < c''$ yields a contraction. Therefore, $\mathcal{RCC}_{n,t,c'n/4^t}$ has $(c/c_t, \delta n)-$XOR amplification for $\wedge_t$. ◀

A.3 Halfspace sources

In this subsection, for halfspace sources, we construct an efficient extractor that has linear output for linear min-entropy and exponentially small error. Formally, we will prove the following theorem.

▶ **Theorem 47.** *There exists an explicit seedless* $((1-c_1)n,\ 2^{-c_2 n})$ *extractor* $\mathsf{Ext}: \{0,1\}^n \to \{0,1\}^{c_3 n}$ *for halfspace sources, where* c_1, c_2, c_3 *are some positive small enough constants.*

Note that Nisan already proved an exponentially small correlation bound for Inner Product function against LTFs. Formally, let $IP_n : (\{0,1\}^2)^{n/2} \to \{0,1\}$ denote the inner product function over n variables, i.e., $IP_n(x_1, \ldots, x_{n/2}) = \bigoplus_{i\in[n/2]} \wedge_2(x_i)$. Then, we have the following lemma.

▶ **Lemma 48.** *For any LTF* f *on* n *variables, we have*

$$Cor(IP_n, f) \le 2^{-\Omega(n)}.$$

Proof of sketch. Nisan proved that a LTF on n variables can be approximated within ϵ error by a randomized 2-party protocol of complexity $O(\log(n/\epsilon))$ by [19, Theorem 1]. Moreover, by Chor and Goldreich [7], we know at least $n/2 - \log(1/\epsilon)$ complexity needed for randomized 2-party protocol computing the function IP_n.

Therefore, for any LTF f over n variables, there is a protocol $\mathcal{P}$ of complexity cn bits approximating f within $2^{-\Omega(n)}$ error and $Cor(IP_n, \mathcal{P}) \le 2^{-\Omega(n)}$. That is, replacing f by IP_n in $Cor(IP_n, f)$, we can bound $Cor(IP_n, f) \le 2^{-\Omega(n)} + Cor(IP_n, \mathcal{P}) = 2^{-\Omega(n)}$. ◀

Let $\mathcal{LTF}_n$ denote the class of LTFs over n variables. Then, the above lemma directly yields that $\mathcal{LTF}_n$ has $(\alpha, \delta n)$-XOR amplification for $\wedge_2$ for any positive constant $\delta < 1/2$, where α is some positive constant. Hence Theorem 47 directly follows by Theorem 34.

B Application of Theorem 8

In this section, we construct extractors for sources recognized by several widely used function families. These constructions are all based on Lemma 39 proved in the previous section, which means we can convert seed-extending PRGs into extractors. In the following subsections, the main points are to construct seed-extending PRGs for some specific common function families.

B.1 Circuit-recognizable sources

Recall that we say a function $h : \{0,1\}^t \to \{0,1\}$ is ϵ-hard for $\mathcal{C}$ if $Cor(h, \mathcal{C}) \le \epsilon$.

For any circuit family, Nisan and Wigderson [20] already constructed a hardness-based PRG. Reviewing the NW generator, Kinne et al. [16] proved that it could be made seed-extending, and hence they gave a seed-extending PRG for circuits. In particular, they proved the following lemma.

▶ **Lemma 49** ([16, Lemma 2.9])**.** *Let* l *and* m *be positive integers and* $H : \{0,1\}^{\sqrt{l}/2} \to \{0,1\}$ *a function. If* H *is* $\frac{\epsilon}{m}$*-hard at input length* $\sqrt{l}/2$ *for circuits of size* $s + m \cdot 2^{O(\log m/\log l)}$ *and depth* $d+1$*, then there is a seed-extending* (l, ϵ)*-PRG* $NW_{H;l,m} : \{0,1\}^l \to \{0,1\}^{l+m}$ *for tests* $T : \{0,1\}^{l+m} \to \{0,1\}$ *computable by circuits of size* s *and depth* d*.*

Notice that the set of bounded-size circuits is flip-invariant since flipping the inputs of a circuit does not change its size. Thus, applying Lemma 39, we get an extractor.

▶ **Proposition 50.** *For any positive integer $l < n$, if there is a function H that is ϵ-hard at input length $\sqrt{l}/2$ for circuits of size $s + (n-l) \cdot 2^{O(\log(n-l)/\log l)}$ and depth $d+1$, then for any $\Delta = \Delta(n) > 0$ we can get an $(n - \Delta, (n-l)2^{\Delta}\epsilon)$-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-l}$ for any sources recognizable by circuits of size s and depth d.*

We remark that, in the best case, the above lemma yields an $(n - \tilde{O}(\sqrt{l}), 2^{-\tilde{\Omega}(\sqrt{l})})$-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-l}$, if we can get a function at input length $\sqrt{l}/2$ which is $2^{-\tilde{\Omega}(\sqrt{l})}$-hard for circuits of polynomial size.

B.2 AC^0-ecognizable sources

Hastad [13] proved that the parity function is $2^{-n^{1/(d+1)}}$-hard against any AC^0 circuit of size $2^{n^{1/(d+1)}}$ and depth d. Based on this hardness, Shaltiel [25] constructed extractors for AC^0-recognizable sources.

▶ **Theorem 51** (Corollary 4.25, [25])**.** *For any $\Delta = \Delta(n) > 0$, there is a constant $\alpha > 0$ such that for every sufficiently large n, $m \leq n^{1/(\alpha d)}$, and sources recognizable by circuits of size $2^{n^{1/(\alpha d)}}$ and depth d, we can construct an $(n - n^{1/(\alpha d)}, 2^{-100m})$-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$.*

▶ **Theorem 52** (Theorem 4.21, [25])**.** *For any constants $c, d, e > 1$ there is a constant $d' > 1$ and a uniform family $E = \{E_n\}$ of circuits of polynomial-size and depth d' such that $E_n : \{0,1\}^n \to \{0,1\}^m$ for $m(n) = (\log n)^e$ and E_n is a $(n - 100m(n), 2^{-100m(n)})$-extractor for sources recognizable by circuits of size n^c and depth d.*

However, directly using the Lemma 50 with the hardness of parity function, we can get the following lemma.

▶ **Theorem 53.** *For any $\Delta = \Delta(n) > 0$, there exists a polynomial time computable $(n - \Delta, (n-l)2^{\Delta - \Omega(l^{1/(2d+2)})})$ extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{n-l}$ for any sources recognizable by circuits of size $2^{n^{1/d}}$ and depth d.*

▶ **Proposition 54.** *For any constants $c, d, e > 1$ there is a constant $e' < e$ and a polynomial-time computable uniform family $E = \{E_n\}$ such that $E_n : \{0,1\}^n \to \{0,1\}^m$ for $m(n) = n - (\log n)^e$ and E_n is a $(n - 100(\log n)^{e'}, 2^{-100(\log n)^{e'}})$-extractor for sources recognizable by circuits of size n^c and depth d.*

In particular, for min-entropy $n - n^{1/(\alpha d)}$, our extractor outputs $n - n^{2/\alpha + O(1/d)}$ bits, whereas Shaltiel's extractor outputs only $n^{1/(\alpha d)}$ bits. When $\alpha > 2d/(d-1)$ is a large enough constant, our extractor outputs $n - o(n)$ bits whereas Shaltiel's extractor outputs only $n^{1/(\alpha d)}$ bits. For min-entropy $n - \mathrm{polylog}(n)$ bits, our extractor outputs $n - \mathrm{polylog}(n)$, whereas Shaltiel's extractor outputs only $\mathrm{polylog}(n)$ bits

For circuit sources, Viola [29] also constructed extractors for AC^0-samplable sources, extracting $k(k/n^{1+\gamma})^{O(1)}$ bits with super-polynomially small error from n-bit sources of min-entropy k, for any $\gamma > 0$. Nevertheless, AC^0-samplable sources are different from AC^0-recognizable sources.