Approximating Small Sparse Cuts

Aditya Anand* Euiwoong Lee† Jason Li‡ Thatchaphol Saranurak§

Abstract

We study polynomial-time approximation algorithms for (edge/vertex) Sparsest Cut and Small Set Expansion in terms of k, the number of edges or vertices cut in the optimal solution. Our main results are $\mathcal{O}(\text{polylog}\, k)$-approximation algorithms for various versions in this setting.

Our techniques involve an extension of the notion of sample sets (Feige and Mahdian STOC'06), originally developed for small balanced cuts, to sparse cuts in general. We then show how to combine this notion of sample sets with two algorithms, one based on an existing framework of LP rounding and another new algorithm based on the cut-matching game, to get such approximation algorithms. Our cut-matching game algorithm can be viewed as a local version of the cut-matching game by Khandekar, Khot, Orecchia and Vishnoi and certifies an expansion of every vertex set of size s in $\mathcal{O}(\log s)$ rounds. These techniques may be of independent interest.

As corollaries of our results, we also obtain an $\mathcal{O}(\log opt)$-approximation for min-max graph partitioning, where opt is the min-max value of the optimal cut, and improve the bound on the size of multicut mimicking networks computable in polynomial time.

*University of Michigan, Ann Arbor, USA

†University of Michigan, Ann Arbor, USA. Supported in part by NSF grant CCF-2236669 and Google

‡Carnegie Mellon University, Pittsburgh, USA

§University of Michigan, Ann Arbor, USA. Supported by NSF grant CCF-2238138

Contents

1 Introduction

Given a graph G and a cut $(X, \overline{X})$, the *sparsity* of the cut $(X, \overline{X})$ is given as $\frac{|\delta_G(X)|}{\min(|X|,|\overline{X}|)}$, where $\delta_G(X)$ denotes the set of edges with exactly one endpoint in X. Sparsest Cut is the problem whose goal is to find a cut with minimum sparsity. It is one of the most fundamental and well-studied problems in multiple areas of algorithms. In approximation algorithms, this problem has been the focus of the beautiful connection between algorithms, optimization, and geometry via metric embeddings [42, 44, 5, 6, 38, 50]. Through expander decomposition [34, 59], finding sparse cuts is also one of the most fundamental building blocks in designing algorithms for numerous graph problems in many areas, including graph sketching/sparsification [3, 31, 15], Laplacian solvers [62, 17], maximum flows [35], Unique Games [63] and dynamic algorithms [48, 67, 49]. Over the years, there has been a lot of progress in approximating sparse cuts [42, 6, 21, 37] culminating in the seminal paper of Arora, Rao and Vazirani [6], which gave an $\mathcal{O}(\sqrt{\log n})$-approximation for sparsest cut.

A closely related problem, which can be viewed as a generalization of Sparsest Cut, is Small Set Expansion, introduced by Raghavendra and Steurer [56]. In Small Set Expansion, we are additionally given a size parameter s, and the goal is to find a cut $(Y, \overline{Y})$ with minimum sparsity satisfying $\min(|Y|, |\overline{Y}|) \leq s$.[1] Closely related to the Unique Games Conjecture, it is one of the central problems in the hardness of approximation literature with applications to graph partitioning, continuous optimization, and quantum computing [57, 4, 7, 58, 8, 45, 10, 26, 41]. Using the hierarchical tree decomposition of Racke [54] one obtains an $\mathcal{O}(\log n)$-approximation for Small Set Expansion.

Suppose the optimal cut $(S, \overline{S})$ for Sparsest Cut/Small Set Expansion in a graph cuts k edges, while separating $|S| = s$ vertices from the rest of the graph (so that it is $\phi = \frac{k}{s}$-sparse). Most approximation algorithms studied for both these problems have an approximation ratio that is a function of n, the number of vertices. However, it is natural to expect that for an instance we may have $k \ll n$. Thus a natural question is: can we obtain a *parametric approximation* - an approximation algorithm whose approximation ratio is a function of k alone?

Clearly $k \leq n^2$, and such an approximation can be much better if the sparse cut cuts only a few edges.

The notion of parametric approximation is closely related to that of Fixed Parameter Tractability (FPT), where given a parameter ℓ, we seek exact algorithms that run in time $f(\ell)n^{\mathcal{O}(1)}$. Besides the fact that both these notions seek to improve performance when the parameter ℓ is small, it is easy to see that if a minimization problem P is fixed parameter tractable with respect to a parameter ℓ, then it also admits an $g(\ell)$ approximation in polynomial time for some function g. On the other hand, parametric approximations have often been used as the starting point in the design of polynomial kernels - see for instance the kernels for Chordal Vertex Deletion [32, 1] and $\mathcal{F}$-minor deletion [23].

The result of Cygan et al. [18] for Minimum Bisection shows that Small Set Expansion admits an FPT algorithm running in time $2^{\mathcal{O}(k \log k)}n^{\mathcal{O}(1)}$. Combining this with the known $\mathcal{O}(\log n)$ approximation for Small Set Expansion and the $\mathcal{O}(\sqrt{\log n})$ approximation for Sparsest Cut gives an $\mathcal{O}(k \log k)$ approximation for Small Set Expansion and an $\mathcal{O}(\sqrt{k \log k})$ approximation for Sparsest Cut in polynomial time.

This leads to the natural question: can we obtain an $\mathcal{O}(\text{polylog}(k))$ approximation for these problems in polynomial time? This question was also raised by Wahlström [65, 66], who showed

[1] For Small Set Expansion, we are typically interested in (bi-criteria) approximation algorithms which only violate the size constraint by a constant factor.

that this would have implications in designing quasipolynomial kernels for various cut problems. The work of Feige et al. [21] shows that one can obtain an $\mathcal{O}(\sqrt{\log k})$ approximation when $s = \Omega(n)$, but no such guarantee is known for general s.

In this work, we systematically study parametric approximation algorithms for Sparsest Cut and Small Set Expansion and their vertex analogs Vertex Sparsest Cut and Small Set Vertex Expansion, whose approximation ratio is a function of k, the number of edges/vertices cut. It turns out that these four variants show very different behaviour in terms of algorithms and hardness of approximation in terms of k. We also study fast algorithms for some of these problems and obtain algorithms that run in almost-linear time.

1.1 Our results

Edge cuts. We begin by considering approximation algorithms for Sparsest Cut and Small Set Expansion. For the sake of clarity we define both these problems formally.

Sparsest Cut	**Parameter:** ϕ
Input: Graph G	
Question: Find a set of vertices S' with $\lvert S'\rvert \leq \frac{n}{2}$ such that $\lvert\delta_G(S')\rvert \leq \phi\lvert S'\rvert$, or report that no such set exists.	

Small Set Expansion	**Parameter:** ϕ,s
Input: Graph G	
Question: Find a set of vertices S' with $\lvert S'\rvert \leq s$ and $\lvert\delta_G(S')\rvert \leq \phi\lvert S'\rvert$, or report that no such set exists.	

Definition 1.1 ((α,β)-approximation for Small Set Expansion)**.** *We say that an algorithm is an (α,β)-approximation for* Small Set Expansion *if given parameters ϕ, s, it outputs a set $S' \subseteq V(G)$ with $|S'| \leq \beta s$ and $|\delta_G(S')| \leq \alpha\phi|S'|$, or reports that no $S \subseteq V(G)$ satisfies $|S| \leq s$ and $|\delta_G(S)| \leq \phi|S|$. An $(\mathcal{O}(\alpha),\mathcal{O}(1))$-approximation will be simply called an $\mathcal{O}(\alpha)$-approximation.*

We show the first $\mathcal{O}(\log k)$-approximation for Small Set Expansion. This improves upon the $\mathcal{O}(\log n)$-approximation using [54] and answers the question of [65], directly improving the result (discussed in detail in Section 8.1).

Theorem 1.2. *Let k be the smallest cut-size $|\delta_G(S)|$ among all sets S satisfying $|S| \leq s$ and $|\delta_G(S)| \leq \phi|S|$. Then* Small Set Expansion *admits an $(\mathcal{O}(\log k),\mathcal{O}(1))$-approximation in polynomial time.*

Note that in particular this implies an $\mathcal{O}(\log k)$-approximation for Sparsest Cut. Our next result gives an almost-linear algorithm for Sparsest Cut using a new cut-matching game approach. We show an $\mathcal{O}(\log^2 k)$-approximation algorithm for Sparsest Cut which runs in time $m^{1+o(1)}$ using a new cut-matching game approach to construct *small set expanders*, that we discuss in Section 6. These techniques may be of independent interest.

Theorem 1.3. *Let k be the smallest cut-size $|\delta_G(S)|$ among all sets S satisfying $|\delta_G(S)| \leq \phi|S|$. Then there is an $\mathcal{O}(\log^2 k)$-approximation algorithm for* Sparsest Cut *which runs in time $m^{1+o(1)}$, where m denotes the number of edges in the graph.*

We leave as an open problem the question of if we can obtain an $\mathcal{O}(\text{polylog}(k))$-approximation for Small Set Expansion in almost-linear time: we remark that even in terms of n, to the best of our knowledge, the best known approximation ratio for Small Set Expansion in almost-linear time is $\mathcal{O}(\log^4 n)$ by using dynamic programming on tree cut-sparsifiers obtained using [55].

Vertex cuts. We consider approximation algorithms for Vertex Sparsest Cut and Small Set Vertex Expansion. Again we define these problems formally. Given a graph G, we denote a vertex cut by a triplet (L, C, R) where they form a partition of $V(G)$, and after removing C there are no edges between L and R. The vertex sparsity of (L, C, R) is defined to be $\frac{|C|}{|C|+\min(|L|,|R|)}$.

Vertex Sparsest Cut	**Parameter:** ϕ
Input: Graph G	
Question: Find a vertex cut (L, C, R) with $\|R\| \geq \|L\|$ and $\|C\| \leq \phi\|L \cup C\|$, or report that no such cut exists.	

Small Set Vertex Expansion	**Parameter:** ϕ,s
Input: Graph G	
Question: Find a vertex cut (L, C, R) with $\|R\| \geq \|L\|$, $\|C\| \leq \phi\|L \cup C\|$ and $\|L\| \leq s$ or report that no such cut exists.	

Like in the case of edge cuts, we investigate approximation in terms of k, where k is the number of cut vertices in the optimal cut. We begin by observing that Small Set Vertex Expansion is at least as hard to approximate as a well-known problem of Densest k-Subgraph: Given a graph and an integer $k \in \mathbb{N}$, Densest k-Subgraph asks to find a subset of k vertices that induce the most number of edges. Relationships between Densest k-Subgraph and problems related to Small Set Vertex Expansion (e.g., Minimum k-Union, Bipartite Small Set Vertex Expansion) were discussed in previous works [27, 16, 14].

Theorem 1.4. *Suppose* Densest k-Subgraph *is α hard to approximate under some assumption A. Then there is no bi-criteria $(\mathcal{O}(\sqrt{\alpha}), \mathcal{O}(1))$-approximation for* Small Set Vertex Expansion *under A.*

Since Densest k-Subgraph is believed to be hard to approximate within a factor $n^{1/4-\epsilon}$ for any constant $\epsilon > 0$ [9, 33], one already has a strong contrast between Small Set Expansion and Small Set Vertex Expansion in terms of n.

Surprisingly, a much stronger hardness can be proved in terms of k: we show that there is no $2^{o(k)}$-approximation even when we allow a parameterized running time of $f(k)\text{poly}(n)$, assuming the Strongish Planted Clique Hypothesis (SPCH) [46]. This is in contrast to the edge version Small Set Expansion, where we obtained an $\mathcal{O}(\log k)$-approximation.

Theorem 1.5. *Assuming the SPCH, for $\alpha(k) = 2^{o(k)}$ and any functions $f(k), \beta(k)$, there is no $f(k) \cdot \text{poly}(n)$-time $(\alpha(k), \beta(k))$-approximation algorithm for* Small Set Vertex Expansion.

The 2^k-approximation is tight; using the treewidth reduction technique of Marx et al. [47], we give a matching upper-bound that gives a 2^k-approximation in time $2^{2^{\mathcal{O}(k^2)}}\text{poly}(n)$, under a mild assumption which we call maximality. A vertex cut (L, C, R) is *maximal* if for all partitions of the connected components of $G \setminus C$ into L', R' with $|L'| \leq |R'|$, we have $|L| \geq |L'|$. That is, L and R are chosen to obtain the sparsest cut possible after removing C.

Theorem 1.6. *Suppose that there is a maximal vertex cut (L, C, R) with $|C| = k$ that separates $s = |L| \leq |R|$ vertices. Then we can find a cut (L', C', R'), with $|C'| \leq k$ and $|L| \geq |L'| \geq \frac{|L|}{2^k}$ in time $2^{2^{\mathcal{O}(k^2)}} n^{\mathcal{O}(1)}$. In particular, there is a $(2^k, 1)$-approximation algorithm for* Small Set Vertex Expansion *in time $2^{2^{\mathcal{O}(k^2)}} n^{\mathcal{O}(1)}$.*

We now turn our attention to Vertex Sparsest Cut, where we do not have the additional small set requirement. First, we note that Theorem 1.6 also gives a $g(k)$-approximation for Vertex Sparsest Cut in (non-parameterized) polynomial time for some function g. To see this, consider the following algorithm. Whenever $k \geq \frac{\sqrt{\log \log n}}{C}$, for some large enough constant C, we use a standard poly-time $\mathcal{O}(\sqrt{\log n})$-approximation (see for example [21]), and otherwise we use the above algorithm. It follows that this is a polynomial time algorithm with approximation ratio $g(k) = 2^{\mathcal{O}(k^2)}$.

The natural question is then if one can beat $2^{\text{poly}(k)}$-approximation for Vertex Sparsest Cut. We (almost) show that this is indeed the case: we obtain an $\mathcal{O}(\log k + \log \log n\phi)$-approximation in polynomial time, if the optimal cut has sparsity ϕ and cuts k vertices. The additional $\log \log n\phi$ term as compared to the edge version is due to the weaker guarantee on our technique of sample sets in the vertex version (see Section 4). We show that at least by using sample sets, we cannot hope to improve this result (Section 4.3).

Theorem 1.7. *Let k be the minimum possible value of $|C|$ among all vertex cuts (L, C, R) that are ϕ-sparse. Then* Vertex Sparsest Cut *admits an $\mathcal{O}(\log k + \log \log n\phi)$-approximation in polynomial time.*

Again, we give a fast algorithm using our cut-matching game: we obtain an $\mathcal{O}(\log^2 k + \log^2 \log n\phi)$-approximation in time $m^{1+o(1)}$.

Theorem 1.8. *Let k be the minimum possible value of $|C|$ among all vertex cuts (L, C, R) that are ϕ-sparse. Then* Vertex Sparsest Cut *admits an $\mathcal{O}(\log^2 k + \log^2 \log n\phi)$-approximation in time $m^{1+o(1)}$.*

We note that the only regime in which these results are not necessarily an $\mathcal{O}(\text{polylog}(k))$-approximation is when $k \leq (\log n)^{o(1)}$. Table 1 summarizes our results.

Edge/Vertex	Approximation ratio	Small set	Time
Edge	$\mathcal{O}(\log k)$	yes	poly(n)
Edge	$\mathcal{O}(\log^2 k)$	no	almost-linear
Vertex	$\mathcal{O}(\log k + \log \log n\phi)$	no	poly(n)
Vertex	$\mathcal{O}(\log^2 k + \log^2 \log n\phi)$	no	almost-linear
Vertex	No n^ϵ-approximation for some $\epsilon > 0$	yes	poly(n)
Vertex	No $2^{o(k)}$-approximation	yes	$f(k)$poly(n)
Vertex	2^k	yes[2]	$f(k)$poly(n)

Table 1: Summary of our results

Applications. Next, we describe two applications of our result and techniques for Small Set Expansion to multicut mimicking networks and Min-Max Graph Partitioning.

Multicut mimicking networks. We show an improvement in the size of the best multicut mimicking network that can be computed in polynomial time. This follows directly from the reduction in [66] to Small Set Expansion and our $\mathcal{O}(\log k)$-approximation for it. We start by defining multicut mimicking networks.

[2]Under maximality.

Definition 1.9. *Given a graph G and a set of terminals $T \subseteq V(G)$, let G' be another graph with $T \subseteq V(G')$. We say that G' is a multicut mimicking network for G if for every partition $\mathcal{T} = T_1 \cup T_2 \cup \ldots \cup T_s$ of T, the size of a minimum multiway cut for $\mathcal{T}$ is the same in G and G'.*

We show the following theorem using our $\mathcal{O}(\log k)$-approximation for Small Set Expansion. Given a graph G and a set of terminals $T \subseteq V(G)$, let $cap_G(T)$ denote the total degree of the vertices in T.

Theorem 1.10. *Given a terminal network (G, T) with $cap_G(T) = k$, one can find in randomized polynomial time a multicut mimicking network of size $k^{\mathcal{O}(\log^2 k)}$.*

This improves the result in [66] which shows a bound of $k^{\mathcal{O}(\log^3 k)}$. This in turn improves the best known kernels for many problems in polynomial time, which are based on multicut mimicking networks as considered in [66]. In particular, Multiway Cut parameterized by the cut-size and Multicut parameterized by the number of pairs and the cut-size together, admit $k^{\mathcal{O}(\log^2 k)}$ size kernels in polynomial time.

Min-Max Graph Partitioning: In Min-Max Graph Partitioning, we are given a graph G and an integer r, and the goal is to partition the vertex set into r parts of size $\frac{n}{r}$ each, while minimizing the maximum cut-size among all parts. Bansal et al. [7] obtain an $(\mathcal{O}(\sqrt{\log n \log r}), \mathcal{O}(1))$-approximation algorithm. Extending our notion of sample sets for sparse cuts to a weighted setting, combined with LP-rounding techniques from [29] and [7] we obtain the following theorem.

Theorem 1.11. *Min-max graph partitioning admits an $(\mathcal{O}(\log opt), \mathcal{O}(1))$-approximation algorithm, where opt is the optimal min-max value.*

1.2 Technical highlight

We use two main techniques to obtain our results. The first is that of sample sets for sparse cuts discussed in Section 4. Feige and Mahdian [22] introduced the concept of sample sets, which are a small set of terminals which represent every set with a (vertex or edge) cut of size at most k in proportion to its size, upto an additive deviation of ϵn, where n is the number of vertices in the graph. They used it to obtain a $2^{\mathcal{O}(k)}\text{poly}(n)$ time algorithm for finding balanced separators of size k. We extend this concept to sparse cuts, and show that there is a small set of terminals that represents all sparse cuts. We show essentially tight bounds for such sample sets, and then combine this with two other techniques — (a) existing LP-based methods, and (b) our new cut-matching game — to obtain our approximation algorithms. Essentially, sample sets for sparse cuts enable us to convert $\mathcal{O}(\log s)$-approximation algorithms to $\mathcal{O}(\log k)$-approximation.

The second main technique we develop is the cut-matching game for obtaining small set expanders, introduced in Section 6. The cut-matching game [37, 36, 51] is a very flexible framework that yields one of the fastest approximations to Sparsest Cut, and hence is used extensively in designing fast graph algorithms. The cut-matching game approach of [36] shows that there is a cut player strategy so that for any matching player, we can construct an expander in $\mathcal{O}(\log n)$ rounds. We show that there is a cut player strategy to construct s-small set expanders in $\mathcal{O}(\log s)$ rounds: these are graphs in which every set with size at most s is expanding. We show how to achieve this in time $\tilde{\mathcal{O}}(m)$, where m is the number of edges. Together with the almost-linear time max-flow algorithm of [13] which we use for the matching player strategy, this forms the basis of all our almost-linear time algorithms.

1.3 Related work and connections to FPT algorithms

Vertex cuts. Apart from a number of aforementioned works on approximating the edge version of Small Set Expansion and Sparsest Cut, there is a large literature on approximating Small Set Vertex Expansion and Vertex Sparsest Cut. For Vertex Sparsest Cut, Feige et al. [21] used the ARV algorithm to obtain an $\mathcal{O}(\sqrt{\log n})$-approximation. As we discussed, Small Set Vertex Expansion is much harder to approximate. Louis and Makarychev [45] obtained $\tilde{\mathcal{O}}(\frac{n}{s}\sqrt{\log n})$ and $\tilde{\mathcal{O}}(\frac{n}{s}(\sqrt{\psi \log d} + \psi))$ approximations where ψ is the vertex conductance of the optimal cut and d is the maximum degree.

Approximation algorithms with respect to other parameters. Approximation algorithms where the approximation ratio is a function of a parameter rather than the input size have been studied before for various graph partitioning problems. s For instance, Garg et al. showed an $\mathcal{O}(\log t)$-approximation algorithm for Multicut

where t is the number of terminal pairs that need to be separated [25]. Even et al. [20] studied the approximation of Feedback Vertex/Arc Sets in directed graphs, and obtained $\mathcal{O}(\log opt \log\log opt)$-approximations, where opt is the size of the optimal feedback set. Even et al. [19] obtained $\mathcal{O}(\log opt)$-approximations for many graph partitioning problems using spreading metrics, where opt is the size of the optimal cut. Calinescu et al. [11] obtained an $\mathcal{O}(\log k)$-approximation for the 0-Extension problem with k terminals. Lee [40] obtained an $\mathcal{O}(\log k)$-approximation for k-Vertex Separator and other related problems. As previously mentioned, Feige et al. [21] gave an $\mathcal{O}(\sqrt{\log k})$-approximation for Vertex Sparsest Cut when the optimal solution is balanced, which was an important ingredient in approximating treewidth in polynomial time.

There are previous results for Small Set Expansion whose approximation guarantee depends on s, the size of the set that we seek to find. Bansal et al. [7] gave an approximation algorithm for Small Set Expansion in terms of both n and s: their algorithm has an approximation ratio of $\mathcal{O}(\sqrt{\log n \log \frac{n}{s}})$.

Hasan and Chwa [29] obtained an $\mathcal{O}(\log s)$-approximation for Small Set Expansion. Finally, the conductance of the optimal cut $\psi \in (0, 1)$ is another important parameter related to the performance of many algorithms for Sparsest Cut and Small Set Expansion; the well-known Cheeger inequality gives an $\mathcal{O}(\sqrt{\psi^{-1}})$-approximation algorithm. For Small Set Expansion, Raghavendra et al. [57] gave an $\mathcal{O}(\sqrt{\psi^{-1}\log(n/s)})$-approximation.

Connection to FPT algorithms. Cygan et al. [18] showed that Minimum Bisection admits an FPT algorithm that runs in time $2^{\mathcal{O}(k \log k)}\text{poly}(n)$. In fact, given a size parameter $s < \frac{n}{2}$, their algorithm can find a cut with minimum number of cut-edges among all cuts $(X, \overline{X})$ with $|X| = s$. This essentially means we can solve Sparsest Cut and Small Set Expansion exactly when $k \leq \frac{\log n}{C \log\log n}$ for some constant C. But when $k \geq \frac{\log n}{C \log\log n}$, Sparsest Cut admits an $\mathcal{O}(\sqrt{\log n}) = \mathcal{O}(\sqrt{k \log k})$-approximation, and Small Set Expansion admits an $\mathcal{O}(\log n) = \mathcal{O}(k \log k)$-approximation. Thus FPT algorithms imply some $f(k)$-approximations for the edge versions in polynomial time, but our algorithms significantly improve upon these guarantees.

2 Overview

In this section, we describe on a high level our ideas for obtaining approximation algorithms in terms of k. For simplicity, we will focus on edge cuts, and only give an overview of two results: how

we obtain an $\mathcal{O}(\log k)$-approximation for Small Set Expansion, and an $\mathcal{O}(\log^2 k)$-approximation for Sparsest Cut in almost-linear time.

2.1 Approximating Small Set Expansion.

In Small Set Expansion, given that there exists a set S with $|S| = s$ and $|\delta_G(S)| = k$ (so that S is $\phi = \frac{k}{s}$-sparse), we wish to find a set which has size at most $\mathcal{O}(s)$ and has sparsity $\phi' = \mathcal{O}(\phi \log k)$.

First, we may assume $s \gg k \log k$, otherwise, the $\mathcal{O}(\log s)$-approximation of [29] is already an $\mathcal{O}(\log k)$-approximation.

Is there a way to still use (an extension of) an $\mathcal{O}(\log s)$-approximation algorithm even when $s \gg k \log k$? Our basic approach is to carefully choose a *sample set* $T \subseteq V$ that sparsifies the vertex set in the following sense. For $S \subseteq V$, define the *terminal sparsity* of S as $\phi_T(S) := \frac{|\delta_G(S)|}{|S \cap T|}$; intuitively, we pretend that the size of the set S is $|S \cap T|$ instead of $|S|$. Ideally,

(i) if every $S' \subseteq V$ satisfies $|S' \cap T| \approx |S'| \cdot \frac{|T|}{n}$ (which implies $\phi_T(S') \approx \phi(S') \cdot \frac{n}{|T|}$), and

(ii) the $O(\log s)$-approximation algorithm for $\phi(.)$ can be strengthened to an $O(\log t)$-approximation for $\phi_T(.)$ with $t = |S \cap T|$,

one can expect to find a set S' with $|S' \cap T| \leq \mathcal{O}(t)$ and $\phi_T(S') \leq \mathcal{O}(\log t \cdot \phi_T(S))$, which implies that $|S'| \leq \mathcal{O}(s)$ and $\phi(S') \leq \mathcal{O}(\log t \cdot \phi(S))$. If we could set t to be sufficiently small so that $O(\log t) = O(\log k)$, we will achieve our goal.

Of course, (i) is impossible to have for every $S \subseteq V$ (e.g., when $S \subseteq T$), but one can hope to satisfy it for every sparse cut. The following definition requires that every S sparse in either $\phi(S)$ or $\phi_T(S)$ satisfies $|T \cap S| \approx |S| \cdot \frac{|T|}{n}$; in other words, any set S sparse with respect to ϕ should be sparse with respect to ϕ_T, and vice versa.

Definition 2.1 (Simplification of Definition 4.1)**.** *Given a graph $G = (V(G), E(G))$ and parameters ϵ, ϕ', an (ϵ, ϕ') edge sample set for G is a non-empty set of terminals $T \subseteq V(G)$ which has the following property. Consider a set of vertices $W \subseteq V(G)$ which satisfies either (a) $\phi(W) \leq \phi'$ or (b) $\phi_T(W) \leq \frac{n}{|T|}\frac{\phi'}{10}$. Then we have $|\frac{n}{|T|}|W \cap T| - |W|| \leq \epsilon |W|$.*

This definition is a refined notion of the previously defined sample sets (e.g., [22]), which allows an additive ϵn error and preserves only balanced cuts that have $\Omega(n)$ vertices on each side. Based on Steiner decomposition, we prove in Lemma 4.2 that there exists a sample set of size $\min\{n, \Theta(\frac{n\phi'}{\epsilon^2})\}$. Compute such a sample set T with $\phi' = \mathcal{O}(\phi \log k) = \mathcal{O}(\frac{k \log k}{s})$ as our target sparsity and $\epsilon = 1/10$. Since we assumed $s \gg k \log k$, we must have that T is of size $\Theta(\frac{n\phi'}{\epsilon^2}) < n$.

Since our target set S is ϕ-sparse and therefore ϕ'-sparse, we must have $t := |S \cap T| = \Theta\left(\frac{|T|}{n}|S|\right) = \Theta(k \log k)$, so that the terminal sparsity of S is $\mathcal{O}(\frac{1}{\log k})$. If the condition (ii) above is true, we are able to find a set S' with $|S' \cap T| = \mathcal{O}(t)$ and $\phi_T(S') = \mathcal{O}(\log k \cdot \frac{1}{\log k}) = \mathcal{O}(1)$. Notice that $\frac{n}{10|T|}\phi' = \Theta(1)$. By carefully choosing constants one can ensure that S' is $\frac{n}{10|T|}\phi'$-terminal sparse, and hence must satisfy the sample condition above. It implies that $|S'| = \Theta(\frac{n}{|T|}|S' \cap T|) = \mathcal{O}(s)$. Since S' is $\mathcal{O}(1)$-terminal sparse, we have $\phi(S') = \frac{|\delta_G(S')|}{|S'|} = \mathcal{O}\left(\frac{|T|}{n}\frac{|\delta_G(S')|}{|S' \cap T|}\right) = \mathcal{O}\left(\frac{\phi'|\delta_G(S')|}{|S' \cap T|}\right) = \mathcal{O}(\phi') = \mathcal{O}(\phi \log k)$, and it follows that we obtain an $\mathcal{O}(\log k)$-approximation for the sparsity.

Thus, it suffices to achieve (ii) above, which can be considered as a "terminal version" of Small Set Expansion: Given a set S with $|\delta_G(S)| = k$, $|S \cap T| = t = \Theta(k \log k)$ such that S is

$\psi = \mathcal{O}(\frac{1}{\log k})$-terminal sparse, can we find a set with at most $\mathcal{O}(t)$ terminals that is $\psi' = \mathcal{O}(\psi \log t)$-terminal sparse? Note that this is a generalization of the original Small Set Expansion problem when $T = V(G)$.

Hasan and Chwa [29] handled this special case by giving an $\mathcal{O}(\log s)$-approximation by extending the techniques of [7]. We show that the techniques from [29] and [7] extend to the more general version with terminals, and obtain an $\mathcal{O}(\log t)$-approximation, giving an $\mathcal{O}(\log k)$-approximation for Small Set Expansion.

2.2 Cut-matching game for small set expanders.

Our $\mathcal{O}(\log^2 k)$-approximation algorithm for Sparsest Cut (Theorem 1.3) is based on our new cut-matching game for constructing small set expanders. Given a graph G, the cut-matching game frameworks of [37, 36] start with an empty graph H with the same vertex set. Let ϕ be a sparsity parameter. In every round, each player takes the following action.

- The cut player finds a bisection $(B, \overline{B})$ and gives it to the matching player.
- The matching player either finds a ϕ-sparse cut in G (so we are done), or finds a perfect matching M between B and $\overline{B}$ so that $\phi \cdot M \preceq^{\text{flow}} G$, which means that G (with unit capacities) admits a flow simultaneously sending amount ϕ between the endpoints of every $e \in M$. The matching M is added to H.

Khandekar, Khot, Orecchia and Vishnoi [36] proved that there is a cut player strategy that, against any matching player always returning a matching, ensures after $\mathcal{O}(\log n)$ rounds that H is an $\Omega(1)$-expander.[3] This implies that G is an $\Omega(\frac{\phi}{\log n})$-expander as $\phi H \preceq^{\text{flow}} \mathcal{O}(\log n)G$. Therefore, by executing the strategy of both players, one can either obtain a ϕ-sparse cut in G or certify that G is a $\Omega(\frac{\phi}{\log n})$-expander. This gives an $\mathcal{O}(\log n)$-approximation for Sparsest Cut. However, we note that the cut player strategy of [36] is not a poly-time strategy, hence this does not directly give us a polynomial time approximation algorithm.

Cut player strategy for small set expanders. We show that in a similar framework, there is a cut player strategy that, against any matching player always returning a matching, can ensure in $\mathcal{O}(\log s)$ rounds that H is a s-small set $\Omega(\frac{1}{\log s})$-expander; every $S \subseteq V$ with $|S| \leq s$ has $\phi_H(S) \geq \Omega(\frac{1}{\log s})$. Furthermore, we implement the strategy in near-linear time. Given exponential time, we can even ensure that H is a s-small set $\Omega(1)$-expander, thus generalizing the result of [36].

The basic idea is the following. Let S be a set of at most s vertices in H. If we can somehow force the cut player, within $\mathcal{O}(\log s)$ rounds, to output a bisection $(X, \overline{X})$ which is "sufficiently unbalanced" with respect to S, we would be done. Concretely, suppose that $||X \cap S| - |\overline{X} \cap S|| \geq \epsilon |S|$ for some constant ϵ, then in this round, at least $\epsilon|S|$ edges must be added across the cut $(S, \overline{S})$ in H via the matching between X and Y, and hence S becomes expanding.[4]

This motivates our approach. First, we start by finding a $\Omega(1)$ balanced cut $(W, \overline{W})$ in H that is $\frac{1}{C}$-sparse for some large constant $C \gg 1$. If no such cut exists, we show that we can be done using just one more round — we skip this detail in this overview. Next, we use our key subroutine called ImproveCut, which given a cut $(W, \overline{W})$ in H that is $\frac{1}{C}$-sparse, uses a single max-flow call to obtain another balanced cut $(Q, \overline{Q})$ such that for every subset $S \subseteq V(H)$ that is $\frac{1}{K}$-sparse for

[3] A graph H is $(\Omega(1))$-expanding when every $S \subseteq V$ with $|S| \leq n/2$ has $\phi(S) \geq \Omega(1)$. Note that this is with respect to sparsity, not conductance.

[4] A subset $S \subseteq V$ is $(\Omega(1))$-expanding when $\phi(S) \geq \Omega(1)$. Note that this is with respect to sparsity, not conductance.

some constant $K \gg C$, we have $|\delta_{H[S]}(Q \cap S)| \leq \frac{|S|}{D}$, for some large constant D. In other words, the number of edges cut *inside* every significantly sparse set S is at most $\frac{|S|}{D}$.

The cut player then simply outputs the cut $(X = Q, \overline{X} = \overline{Q})$. (Let us assume that $(Q, \overline{Q})$ is an exact bisection in this overview; the actual framework is slightly more involved.)

Fix a set S of size at most s. After some number of rounds, if S is already $\frac{1}{K}$-expanding, we are already done and hence we assume that this is not the case. Now for the cut $(Q, \overline{Q})$ found by the cut player in the current round, if it turns out that $||S \cap Q| - |S \cap \overline{Q}|| \geq \Omega(|S|)$, the matching player is forced to add $\Omega(|S|)$ edges across $(S, V(H) \setminus S)$ in this round, and hence S becomes expanding. Now suppose this does not happen. Hence we may now assume that the cut $(Q, \overline{Q})$ is balanced with respect to S.

Also ImproveCut guarantees that $|\delta_{H[S]}(Q \cap S)| \leq \frac{|S|}{D}$. Since, $|Q \cap S| = |\overline{Q} \cap S| = \Omega(|S|)$, the sparsity of the cut $Q \cap S$ *inside* $H[S]$ must be $\mathcal{O}(\frac{1}{D})$. Thus the cut $(Q, \overline{Q})$ cuts the set S in a balanced and sparse way. If S was the whole graph, this is exactly what the cut player in previous cut-matching games does; finding a sparse balanced cut and giving it to the matching player!

We show that if this happens for more than $\mathcal{O}(\log s)$ rounds, then there must exist a round in which the matching player must add $\Omega(|S|)$ edges across the cut $(S, V(H) \setminus S)$ (see Lemma 6.10). This analysis is a local version of the potential analysis of [36]. It follows that after $\mathcal{O}(\log s)$ rounds, there must be some round in which the matching player added $\Omega(|S|)$ edges across the cut $(S, V(H) \setminus S)$ (or find a sparse cut in G), and thus we obtain our result.

We remark that the key idea behind the ImproveCut subroutine has been used in previous work on related problems. Lang and Rao [39] and Andersen and Lang [2] gave algorithms to improve quotient cuts using maximum flows. Similar ideas were used in local flow algorithms [53, 30]. Saranurak and Wang [59] used a similar algorithm for expander pruning.

Near-linear time implementation: We describe the main technical ingredients in making the above algorithm run in time $\tilde{\mathcal{O}}(m)$. First is the task of finding an $\Omega(1)$ balanced cut in H that is $\frac{1}{C}$-sparse, or to certify that there is no such cut. While we cannot hope to solve this problem exactly in polynomial time, it is possible to check if there is an $\Omega(1)$ balanced cut in H that is $\frac{1}{C}$-sparse, or certify that every balanced cut in H is $\Omega(\frac{1}{\log s})$-expanding. But even though this looks like an $\mathcal{O}(\log s)$-approximation for Sparsest Cut by itself, we can indeed solve this problem as follows. Since our cut-matching game only runs for $\mathcal{O}(\log s)$ rounds, the maximum degree Δ in H is only $\mathcal{O}(\log s)$. We further ensure that the graph H is regular. Then it is enough to find a balanced cut $(W, \overline{W})$ with *conductance* $\frac{|\delta_G(W)|}{\mathrm{vol}(W)} \leq \frac{1}{C\Delta}$, or certify that every balanced cut has conductance at least $\Omega(\frac{1}{C^2\Delta^2})$.

But this is exactly a "Cheeger-type" approximation for balanced low conductance cuts. We use the algorithm of [52] to accomplish this: while their result is slightly different from the notion described above, it will be enough for us to obtain our result (see Theorem 6.6).

Once we have the cut player strategy for small set expanders, it is easy to show that one can use a matching player similar to that of the standard cut-matching game [37] as discussed above and obtain the following result: Given a graph G and a parameter ϕ', either find a ϕ'-sparse cut, or certify that the graph is a s-small set $\Omega(\frac{\phi'}{\log^2 s})$-expander in time $m^{1+o(1)}$. This gives an $\mathcal{O}(\log^2 s)$-approximation for Sparsest Cut.

It is not difficult to show that this extends to terminal sparsity as well: Given a graph G and a parameter ϕ', either find a ϕ'-terminal sparse cut, or certify that every set with at most s terminals is $\Omega(\frac{\phi'}{\log^2 s})$-expanding. Armed with this result for the terminal version and the technique of sample sets used similarly as described for Small Set Expansion, we can obtain an $\mathcal{O}(\log^2 k)$-

approximation. We show that this entire procedure can be accomplished in time $m^{1+o(1)}$, and hence we obtain our algorithm for Sparsest Cut.

Organization of the paper: We start with preliminaries and definitions in Section 3. In Section 4, we define our new notion of sample sets and give both upper and lower bounds for sample sets for both edge and vertex cuts. Section 5 describes how to combine sample sets with LP rounding algorithms to obtain approximation algorithms. In Section 6 we describe our new cut-matching game for small set expanders, and describes how to use sample sets together with this new cut-matching game to obtain fast parametric approximation algorithms. Section 7 proves both our hardness and algorithmic results for Small Set Vertex Expansion. Section 8 discusses the applications of our results and techniques to Min-Max Graph Partitioning and Multicut Mimicking Networks. In Section 9 we state some open problems that arise naturally from this work. Appendix A consists of our generalizations of some existing LP rounding algorithms, which enable us to combine them with the technique of sample sets in Section 5.

3 Preliminaries and notation

We start with basic graph terminology and define the problems which we will consider throughout this paper. All graphs G are connected and undirected unless otherwise stated.

Edge cuts. We will denote an (edge) cut in a graph G by $(U, \overline{U})$ where $U \subseteq V(G)$ and $\overline{U} = V(G) \setminus U$. For the sake of simplicity, we sometimes also simply refer to this cut as U. The set of cut edges is denoted by $\delta_G(U)$. We say that the cut $(U, \overline{U})$ is *b-balanced with respect to a set* $X \subseteq V(G)$ if $\min(|U \cap X|, |\overline{U} \cap X|) \geq b|X|$. A cut is *b-balanced* if it is b-balanced with respect to $V(G)$. The sparsity of a cut $(U, \overline{U})$ is defined as $\frac{|\delta_G(U)|}{\min(|U|,|\overline{U}|)}$. We will denote the sparsity of the cut $(U, \overline{U})$ by $\phi^G(U)$, and omit the superscript when the context is clear. A cut is ϕ'-sparse if its sparsity is at most ϕ', and ϕ'-expanding otherwise. We say that G is a ϕ'-expander if every cut is ϕ'-expanding. We call a cut $(U, \overline{U})$ ϕ'-sparse with respect to a set X if $|\delta_{G[X]}(U)| \leq \phi' \min(|U \cap X|, |\overline{U} \cap X|)$, where $G[X]$ is the graph induced on the vertex set X.

The *volume* of a set X, denoted by $\mathrm{vol}_G(X)$, is the sum of the degrees of the vertices inside X. The *conductance* of a cut $(X, \overline{X})$ is defined as $\frac{|\delta_G(X)|}{\min\{\mathrm{vol}_G(X), \mathrm{vol}_G(\overline{X})\}}$.

Given a set of terminals $X \subseteq V(G)$, we define the *terminal-sparsity* of a cut $(U, \overline{U})$ as $\phi^G_X(U) = \frac{|\delta_G(U)|}{\min(|U \cap X|, |\overline{U} \cap X|)}$. Notice that this is different from sparsity with respect to X, since we count every edge in $\delta_G(U)$ as opposed to only counting the edges $\delta_{G[X]}(U)$. Similarly, we say that $(U, \overline{U})$ is ϕ'-terminal sparse if $\phi^G_X(U) \leq \phi'$, and ϕ'-terminal expanding otherwise. We say that G is a ϕ'-terminal expander if every cut is ϕ'-terminal expanding. We omit the superscript G when the graph is clear from the context.

Vertex cuts. We denote a vertex cut by a partition (L, C, R) of $V(G)$ such that there are no edges between L and R after removing C. The sparsity of the cut is given as $\phi_G(L, C, R) = \frac{|C|}{\min(|L \cup C|, |R \cup C|)}$.

Given a set of terminals X, the (vertex) terminal-sparsity of the cut (L, C, R) is defined as $\phi^G_X(L, C, R) = \frac{|C|}{\min(|(L \cup C) \cap X|, |(R \cup C) \cap X|)}$. Again we omit the superscript when the graph is clear from the context. For a set L, we will denote by $N_G(L)$ the neighbours of the vertices in L which are not in L. We will also talk about the vertex sparsity of a set L: this will be defined as

$\phi^G(L) = \frac{|N_G(L)|}{|L \cup N_G(L)|}$. Similarly, given a set of terminals we define the vertex terminal sparsity of the set L as $\phi^G_T(L) = \frac{|N_G(L)|}{|(L \cup N_G(L)) \cap T|}$. We say that L is ϕ'-(vertex) terminal sparse if $\phi^G_T(L) \leq \phi'$, and ϕ'-(vertex) terminal expanding otherwise. G is a ϕ'-(vertex) terminal expander, if every set $L \subseteq V(G)$ is ϕ'-(vertex) terminal expanding. We also remark that for the sake of simplicity, our notation for vertex and edge sparsity is similar, but the difference shall be clear from the context.

Flow embeddings. Given a flow f on an undirected weighted graph G, with capacity $c(e)$ for each edge $e \in E(G)$, the congestion of the flow f is defined as $\max_e \frac{f(e)}{c(e)}$. If G is unweighted, we have $c(e) = 1$ for each edge e, so that the congestion is $\max_e f(e)$. For graphs H and G, we write $H \preceq^{\text{flow}} cG$, or H flow-embeds in G with congestion c, if one unit of demand can be routed between the vertices corresponding to every edge of H simultaneously in G, with congestion c.

Algorithms and runtime. Given a connected graph G with n vertices and m edges, we say that an algorithm A runs in near-linear time if it runs in time $\tilde{\mathcal{O}}(m)$ where $\tilde{\mathcal{O}}$ hides polylogarithmic factors. We say that A runs in almost-linear time if it runs in time $m^{1+o(1)}$.

4 Sample sets for sparse cuts

In this section, we extend the technique of sample sets in [22] to the setting of sparse cuts. We begin by defining the notion of sample sets for sparse cuts that we will need. Our notion of sample sets will be a small set of terminals which represent every sparse cut in proportion to its size in the actual graph.

Definition 4.1. *Given a graph $G = (V(G), E(G))$ and parameters ϵ, ϕ', an (ϵ, ϕ') edge (vertex) sample set for G is a non-empty set of terminals $T \subseteq V(G)$ which has the following property. Consider a set of vertices $W \subseteq V(G)$ which is either (a) a connected component after removing an edge (vertex) cut (b) the union of all connected components except one after removing an edge (vertex) cut. Further suppose that W satisfies either (a) $\phi(W) \leq \phi'$ or (b) $\phi_T(W) \leq \frac{n}{|T|}\frac{\phi'}{10}$. Then we have $|\frac{n}{|T|}|W \cap T| - |W|| \leq \epsilon |W|$.*

We remark that the number 10 is arbitrary and chosen large enough for clarity and ease of exposition.

Comparison with sample sets in [22]. [22] defined an (ϵ, k) sample: this was a set of terminals $T \subseteq V(G)$, such that for every set W that is either a connected component or the union of all but one connected components after removing an edge/vertex cut, we have $|\frac{n}{|T|}|W \cap T| - |W|| \leq \epsilon n$. They obtained an edge sample set with size $\mathcal{O}(\frac{k}{\epsilon^2})$ and a vertex sample set of size $\mathcal{O}(\frac{k}{\epsilon^2}\log\frac{1}{\epsilon})$. However, this sample set is clearly useful only when W itself is large (at least ϵn). Thus, one can view our notion of sample sets as a strengthening: we preserve all sparse cuts within ϵ multiplicative deviation, instead of just preserving large sets with a small cut-size.

Sample sets using random sampling. Before we proceed further, we note that one can easily obtain a sample set of size $(\min\{\Theta(\frac{n\phi'}{\epsilon^2}\log n), n\})$ with high probability using random sampling for both edge and vertex cuts. We describe this on a high level for edge cuts. Pick a random subset $T \subseteq V(G)$ of $\frac{Cn\phi'}{\epsilon^2}\log n$ terminals. If $\frac{Cn\phi'}{\epsilon^2}\log n > n$, we can simply return the whole vertex set. Consider a ϕ'-sparse set W with $\delta_G(W) = k'$. The expected number of terminals from W is

then $\frac{C\phi'|W|}{\epsilon^2}\log n \geq \frac{Ck'\log n}{\epsilon^2}$, where the inequality follows from the fact that W is ϕ'-sparse. By a Chernoff bound, the number of terminals is within an ϵ multiplicative error with probability $e^{-Ck'\log n}$. However, there are only $n^{\mathcal{O}(k')}$ sets to consider, and applying a union bound on them and on all possible values of k', every such W must satisfy the sample condition with probability at least $1-\frac{1}{n}$. Similarly, we can show that every set W with $\phi_T(W) \leq \frac{n}{|T|}\frac{\phi'}{10}$ must satisfy the sample condition with probability at least $1-\frac{1}{n}$. Applying a union bound then gives us our result.

However, we show that one can significantly improve these results, both for edge and vertex cuts, using techniques from [22]. The rest of this section is organized into three subsections. In the first, we show an upper bound for the size of sample sets for edge cuts. In the second, we show an upper bound for vertex cuts. Finally, we show a matching lower bound for vertex cuts using a connection to Ramsey numbers.

4.1 Edge cuts – upper bound

The next result is our improved upper bound for edge cuts.

Lemma 4.2. *For every $\epsilon \in (0,1)$, and any sparsity parameter ϕ' there is an (ϵ,ϕ') edge sample set for every graph G of size $\min\{\Theta(n\frac{\phi'}{\epsilon^2}), n\}$. Further, such a set can be computed by a deterministic algorithm which runs in near-linear time.*

In fact, our result will be stronger than the notion defined in Definition 4.1 in that it will hold for any ϕ'-sparse set W, not just a connected component or the union of all connected components except one after removing an edge cut.

Proof. Our proof is similar to the construction of the deterministic sample set in [22] for edge separators. We will use the notion of Steiner-t-decomposition introduced in [22].

Lemma 4.3 (Lemma 5.2, [22])**.** *Given any graph G and a value $1 \leq t \leq n$, one can compute in near-linear time a partition $V_1, V_2 \ldots V_\ell$ of $V(G)$ and a partition of the edge set $E_1, E_2 \ldots E_\ell, E'$ such that*

1. *$G[E_i]$ is connected for all $i \in [\ell]$.*
2. *$V(G[E_i]) = V_i$ or $V(G[E_i]) = V_i \cup \{u\}$ for some vertex u for each $i \in [\ell]$.*
3. *$|V_i| \leq 2t$ for all $i \in [\ell]$.*
4. *$|V_i| \geq t$ for all $i > 1$.*

We remark here that Lemma 5.2 of [22] only claims a polynomial run-time, but it is quite easy to make this work in time $\tilde{\mathcal{O}}(m)$. For a formal proof, we refer to the proof of Claim 3.3 from Fomin et al. [24], which is a generalization of Lemma 4.3.

We will refer to each vertex set V_i a *bag* in the decomposition. Compute a Steiner-t-decomposition with $t = \frac{\epsilon}{100\phi'}$. Clearly, $\frac{100n\phi'}{\epsilon} \geq \ell \geq \frac{50n\phi'}{\epsilon}$. We now construct a sample set T as follows: arbitrarily pick $\lfloor \frac{|V_i|}{t\epsilon} \rfloor$ vertices from the bag V_i for each $i \in \{2, 3 \ldots l\}$. Note that we may assume that V_i has at least $\lfloor \frac{|V_i|}{t\epsilon} | \rfloor$ vertices, for if not, we must have $\phi' \geq \frac{\epsilon^2}{200}$, and in that case, we can simply return the whole vertex set as our sample set.

Lemma 4.4. *The set T is a $(4\epsilon,\phi')$ sample set for all $\epsilon \in (0, \frac{1}{100})$.*

Proof. We first show that every bag is represented proportional to its size in the terminal set. Observe that from every bag V_i, $i > 1$, we pick $\frac{|V_i|}{t\epsilon} \geq \lfloor \frac{|V_i|}{t\epsilon} \rfloor \geq \frac{|V_i|}{t\epsilon} - 1$ vertices. Since $t \leq |V_i|$ for all $i > 1$, this means that $|V_i \cap T| \in [\frac{|V_i|}{t\epsilon} - \epsilon\frac{|V_i|}{t\epsilon}, \frac{|V_i|}{t\epsilon}]$ for all $i > 1$. Adding, and noting that $|V_1| \leq 2t$ we get $|T| \in [\frac{(n-2t)}{t\epsilon}(1-\epsilon), \frac{n}{t\epsilon}]$. This means that $\frac{n}{|T|}|V_i \cap T| \in [|V_i|(1-\epsilon), \frac{|V_i|}{1-\epsilon}\frac{n}{n-2t}]$ for all $i > 1$. We know that $t = \frac{\epsilon}{100\phi'} \leq \frac{\epsilon}{4}n$, which in turn gives $n - 2t \geq (1 - \frac{\epsilon}{2})n$, and hence we have $|T| \in [(1-\frac{\epsilon}{2})(1-\epsilon)\frac{n}{t\epsilon}, \frac{n}{t\epsilon}]$ or equivalently, $|T| \in [(1-2\epsilon)\frac{n}{t\epsilon}, \frac{n}{t\epsilon}]$ and $\frac{n}{|T|}|V_i \cap T| \in [|V_i|(1-\epsilon), \frac{|V_i|}{1-2\epsilon}]$ which implies that $\frac{n}{|T|}|V_i \cap T| \in [|V_i|(1-\epsilon), |V_i|(1+3\epsilon)]$ for small enough ϵ, for all $i > 1$.

Next, we show that every set with $\phi(W) \leq \phi'$ satisfies the sample condition. Fix such a set W.

Observe that the number of edges cut, $|\delta_G(W)|$ is at most $\phi'|W|$. Mark a bag V_i of the decomposition as "bad" if there is a cut edge which belongs to the set E_i. Additionally, we always mark the bag V_0 as bad. Clearly, the number of bad bags is at most $\phi'|W| + 1$. The total number of vertices in these bags is at most $2t\phi'|W| + t \leq \frac{\epsilon|W|}{4}$ (note that $t \leq \frac{\epsilon|W|}{100}$ since $|W| \geq \frac{|\delta_G(W)|}{\phi'} \geq \frac{1}{\phi'}$). Thus at least $(1 - \frac{\epsilon}{4})|W|$ vertices of W are present in good bags. Observe that by the definition of a good bag, if W intersects a good bag V_i, it must include the whole bag V_i, since there is a way to reach every vertex of the bag using the edges E_i, and none of the edges in E_i are cut.

Let G' be the set of vertices of the good bags contained in W. From the preceeding argument, we know that $|G' \cap W| \geq (1 - \frac{\epsilon}{4})|W|$. Since each good bag V_i has at least $\frac{|V_i|}{t\epsilon}(1-\epsilon)$ vertices of T, we must have $|G' \cap T| \geq (1-\epsilon)(1-\frac{\epsilon}{4})\frac{|W|}{t\epsilon} \geq (1-2\epsilon)\frac{|W|}{t\epsilon}$. This gives $|W \cap T| \geq |G' \cap T| \geq (1-2\epsilon)\frac{|W|}{t\epsilon}$. Noting that $|T| \in [1-2\epsilon, 1]\frac{n}{t\epsilon}$, this gives that $\frac{n}{|T|}|W \cap T| \geq (1-2\epsilon)|W|$.

We now show the proof of the other direction that W does not have too many terminals. As before, the number of bad bags is at most $\phi|W| + 1$ and the total number of vertices in bad bags is at most $\frac{\epsilon|W|}{4}$. Let B be the union of vertices in all the bad bags. Observe that by construction we must have $|B \cap T| \leq \frac{|B|}{t\epsilon}$. Noting that $|T| \geq (1-2\epsilon)\frac{n}{t\epsilon}$, this gives $\frac{n}{|T|}|B \cap T| \leq \frac{|B|}{1-2\epsilon} \leq \frac{\epsilon|W|}{4(1-2\epsilon)} \leq \frac{\epsilon|W|}{2}$ for small enough ϵ. By construction, we must have $|G' \cap T| \leq \frac{|W|}{t\epsilon}$ which gives $\frac{n}{|T|}|G' \cap T| \leq \frac{|W|}{1-2\epsilon} \leq |W|(1+3\epsilon)$. Thus, $\frac{n}{|T|}|W \cap T| \leq \frac{n}{|T|}(|G' \cap T| + |B \cap T|) \leq |W|(1+3\epsilon) + \frac{\epsilon|W|}{2} \leq |W|(1+4\epsilon)$. This completes the proof.

Next, we show that every set with $\phi_T(W) \leq \frac{\epsilon^2}{200}$ satisfies the sample condition (Note that $\frac{\epsilon^2}{200} > \frac{n}{10|T|}\phi'$). We will do this by showing that $\phi_T(W) \leq \frac{\epsilon^2}{200}$ implies $\phi(W) \leq \phi'$ - since we already showed that any set with $\phi(W) \leq \phi'$ satisfies the sample condition, we would be done. Again, we have $\delta_G(W) \leq \phi_T(W)|W \cap T| \leq \frac{\epsilon^2}{200}|W \cap T|$, and this upper bounds the number of bad bags. Define G' and B as before. Recall that from bag V_i we pick at most $\frac{|V_i|}{t\epsilon} \leq \frac{2}{\epsilon}$ terminals. This means that the total number of terminals in bad bags $|T \cap B| \leq \frac{2}{\epsilon} \cdot \frac{\epsilon^2}{200}|W \cap T| \leq \epsilon|W \cap T|$. Thus we must have $|W \cap T \cap B| \leq \epsilon|W \cap T|$. This implies that $|G' \cap T| \geq (1-\epsilon)|W \cap T|$. Since G' is a collection of bags, as in the previous analysis, we must have $|G' \cap T| \in [\frac{|G'|}{t\epsilon}(1-\epsilon), \frac{|G'|}{t\epsilon}]$. This gives

$$|W| \geq |G'| \geq t\epsilon|G' \cap T| \geq t\epsilon(1-\epsilon)|W \cap T| \geq t\epsilon(1-\epsilon)\frac{200}{\epsilon^2}\delta_G(W) \geq \frac{2(1-\epsilon)}{\phi'}\delta_G(W) \geq \frac{1}{\phi'}\delta_G(W).$$

But this means that W is ϕ'-sparse in G, and the previous analysis now directly proves the result. □

Since T clearly has size $\Theta(\frac{n\phi'}{\epsilon^2})$ and the algorithm runs in near-linear time, we are done. □

4.2 Vertex cuts – upper bound

We next show our result for vertex cuts. We will use the following standard Chernoff bounds.

Lemma 4.5 (Chernoff Bounds). *Let $X = X_1 + X_2 \dots X_n$ be the sum of n identically distributed indicator random variables with mean p, and suppose that for any subset $S \subseteq [n]$, we have $\Pr[\bigwedge_{i \in S} X_i = 1] \leq p^{|S|}$. Let $E[X] = \mu = np$.*

Then we have

- $\Pr[X > (1+\delta)\mu] \leq e^{\frac{-\delta^2}{2+\delta}\mu}$ *for all* $\delta > 0$. *In particular, for* $\delta \in (0,1)$, *we have* $\Pr[X > (1+\delta)\mu] \leq e^{\frac{-\delta^2}{3}\mu}$ *and for* $\delta > 1$ *we have* $\Pr[X > (1+\delta)\mu] \leq e^{\frac{-\delta}{3}\mu}$.
- $\Pr[X < (1-\delta)\mu] \leq e^{\frac{-\delta^2\mu}{3}}$ *for* $\delta \in (0,1)$.

We need the following well-known result about set families with small VC-dimension.

Lemma 4.6 (Sauer-Shelah Lemma, [60, 61]). *Let $\mathcal{S}$ be a family of sets on a universe U with VC-dimension d, and let $T \subseteq U$ be given. Then the size of the set family $r = \{S \cap T \mid S \in \mathcal{S}\}$ is at most $\sum_{i=0}^{d-1} \binom{|T|}{i} \leq d\binom{|T|}{d}$.*

Lemma 4.7. *For every $\epsilon \in (0,1)$ and $\phi' < \frac{1}{2}$, there is an (ϵ, ϕ') vertex sample set for every graph G of size $\min\{\Theta(n\frac{\phi'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})), n\}$. Further, there is a randomized algorithm that computes such a set with constant probability in near-linear time.*

Proof. The proof follows the VC-dimension union bound recipe for ϵ-nets and is along the lines of Lemma 2.6 from [22]. We also refer to the proof of Theorem 5.3.4 of [28]. Before we proceed further with the proof, we note that by Lemma 3.4 of [22], the family of sets $W \subseteq V(G)$ with $|N_G(W)| = k'$ which is either a connected component or the union of all connected components except one after removing $N_G(W)$, has VC-dimension at most dk', for some constant d. Also, recall that if a set W is ϕ'-sparse, we have $\frac{|N_G(W)|}{|W \cup N_G(W)|} \leq \phi'$, but since $\phi' < \frac{1}{2}$, we also have $|N_G(W)| \leq |W|$ so that in fact $|N_G(W)| \leq 2\phi'|W|$.

Pick a random subset $T \subseteq V(G)$ of terminals with $|T| = C \cdot \frac{n\phi'}{\epsilon^2}\log\frac{n\phi'}{\epsilon^3}$ for some large constant C that shall be chosen later. If $C \cdot \frac{n\phi'}{\epsilon^2}\log\frac{n\phi'}{\epsilon^3} > n$ we can simply return the whole vertex set as the terminal set, so we assume that this does not happen. We will now show that T is a $(20\epsilon, \phi')$-sample set.

We define families of sets $\mathcal{S}_1^{k'}$ and $\mathcal{S}_2^{k'}$ for each $k' \in \{1, 2 \dots n\}$ as follows. Let $\mathcal{S}_1^{k'}$ be the set of all sets W which are (a) ϕ'-sparse, (b) are either a connected component or the union of all connected components except one after removing a vertex cut and (c) satisfy $|N_G(W)| = k'$. Let $\mathcal{S}_2^{k'}$ be the set of all sets W which are (i) $\frac{\epsilon^2}{10C\log(\frac{n\phi}{\epsilon^3})}$-terminal sparse, (ii) are either a connected component or the union of all connected components except one after removing a vertex cut and (iii) satisfy $|N_G(W)| = k'$.

If T is not a $(20\epsilon, \phi')$ sample set, one of the following two cases must happen. The first is that there exists a $k' \in [n]$ and a set $W \in \mathcal{S}_1^{k'}$ that violates the sample condition, so that $|\frac{n}{|T|}|W \cap T| - |W|| \geq 20\epsilon|W|$. The second case is that there exists a $k' \in [n]$ and a set $W \in \mathcal{S}_2^{k'}$ that violates the sample condition. We will bound the probability of both these "bad" events. Henceforth, we fix a value k' and omit the superscripts and simply denote the set families by $\mathcal{S}_1$ and $\mathcal{S}_2$. First, let E be the event that there exists a set $W \in \mathcal{S}_1$ which violates the sample condition: $|\frac{n}{|T|}|W \cap T| - |W|| \geq 20\epsilon|W|$. Now pick another set of random terminals T' with $|T'| = \frac{|T|(1-\epsilon)}{\epsilon}$. Let F_X be the event that for a fixed set X we have $|X \cap T'| \in (1-\epsilon, 1+\epsilon)\frac{|X|}{n}|T'|$. Note that $\mathbb{E}[|X \cap T'|] = \frac{|X|}{n}|T'|$, so by a Chernoff bound we must have $\Pr[F_X] \geq 1 - 2e^{-\frac{\epsilon^2}{3}\frac{|X|}{n}|T'|}$ for any fixed

set X. For any $W \in \mathcal{S}_1$, since W is ϕ'-sparse, $\frac{|W|}{n}|T'| \geq |W|\phi'\frac{C(1-\epsilon)}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3}) \geq \frac{Ck'}{4\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$. It follows that we must have $\Pr[F_W] \geq \frac{1}{2}$ for a large enough choice of C.

Now let E' be the event that there exists a set $W \in \mathcal{S}_1$ with $N_G(W) = k'$ which (i) violates the sample condition and (ii) also satisfies F_W. Clearly, $\Pr[E'] \geq \Pr[E'|E]\Pr[E]$, and thus $\Pr[E] \leq \frac{\Pr[E']}{\Pr[E'|E]}$. The denominator is at least $\Pr[F_W]$ for any W that violates the sample condition, and thus $\Pr[E] \leq 2\Pr[E']$.

We now bound $\Pr[E']$. For the sake of analysis, imagine that we pick $X = T \cup T'$ together, and then randomly select a subset of X of size $|T|$ and assign it as T, and the rest as T'. We can write

$$\Pr[E'] = \sum_X \Pr[E'|X]\Pr[X]$$

Henceforth, we fix X and bound $\Pr[E'|X]$. Define $R = \{W \cap X \mid W \in \mathcal{S}_1\}$. By the Sauer-Shelah Lemma, $|R| \leq dk'\binom{|X|}{dk'}$. Define E'' to be the event that there exists an $r \in R$ that satisfies $r \cap T \notin (1-10\epsilon, 1+10\epsilon)\ell\epsilon$ and $r \cap T' \in (1-\epsilon, 1+\epsilon)\ell$ for some $|T| = \frac{Cn\phi'}{\epsilon^3}\log\frac{n\phi'}{\epsilon^3} \geq \ell \geq \frac{Ck'}{4\epsilon^3}\log\frac{n\phi'}{\epsilon^3}$. Observe that E' implies E'', since we can set $\ell = |W|\frac{|T'|}{n}$ in E''. Thus it is enough to bound $\Pr[E''|X]$.

Fix $r \in R$ and a value of ℓ. First, observe that we may assume $|r| \geq (1-5\epsilon)\ell$. For if not, we must have $|T' \cap r| \leq |r| \leq (1-5\epsilon)\ell$ and hence $\Pr[E''] = 0$. Now there are two cases. First suppose that $|r| \geq (1+5\epsilon)\ell$. Then $\mu = \mathbb{E}[|T' \cap r|] = \frac{|T'|}{|T|+|T'|}|r| = (1-\epsilon)|r| \geq (1+3\epsilon)\ell$ (for small enough ϵ). But in order to have $\Pr[E'']$ non-zero, we must have $|T' \cap r| \in (1-\epsilon, 1+\epsilon)\ell$. But this means that $|T' \cap r|$ deviates from its mean μ by a multiplicative ϵ (for otherwise E'' is impossible). Using the standard Chernoff Bound this probability is at most $e^{-\frac{\epsilon^2}{3}\ell} \leq e^{-Ck'\log\frac{n\phi'}{\epsilon^3}}$, where the last inequality follows from the fact that $\ell \geq \frac{Ck'}{4\epsilon^3}\log\frac{n\phi'}{\epsilon^3}$.

Alternatively, suppose that $|r| \in (1-5\epsilon, 1+5\epsilon)\ell$. This means that $\mathbb{E}[|T \cap r|] = \frac{|T|}{|T|+|T'|}|r| \in (1-5\epsilon, 1+5\epsilon)\epsilon\ell$. But we have $|T \cap r| \notin (1-10\epsilon, 1+10\epsilon)\ell\epsilon$, hence we must have a multiplicative deviation of 2ϵ from the mean μ. For a multiplicative deviation of 2ϵ from this mean, using the Chernoff bound, the probability is at most $2e^{\frac{-\epsilon^3}{3}\ell}$. Since $\ell \geq \frac{Ck'}{4\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$ this probability is at most $2e^{-\frac{C}{12}k'\log\frac{n\phi'}{\epsilon^3}}$.

Since the VC-dimension is at most dk', by the Sauer-Shelah Lemma, $|R|$ is at most $dk'\binom{|X|}{dk'}$. Using the fact that $\binom{|X|}{dk'} \leq (\frac{|X|e}{dk'})^{dk'}$, we obtain that this is at most $(\frac{n\phi'}{\epsilon^3})^{10dk'}$. Together, after applying a union bound on these sets, and on the $\frac{Cn\phi'}{\epsilon^3}\log\frac{n\phi'}{\epsilon^3}$ choices of ℓ, the failure probability is at most $10\frac{Cn\phi'}{\epsilon^3}\log\frac{n\phi'}{\epsilon^3}e^{-\frac{C}{12}k'\log\frac{n\phi'}{\epsilon^3}}(\frac{n\phi'}{\epsilon^3})^{10dk'} < \frac{1}{100k'^2}$ for a large enough choice of C.

For the other case, let E_2 be the event that there is a set $U \in \mathcal{S}_2^{k'}$ which is not ϕ'-sparse. We will bound the probability of the event E_2. Recall that we already showed the sample set guarantee for all ϕ'-sparse sets, so if E_2 does not occur, every such set U will satisfy the sample set guarantee. First, we start by noting that for any such U which is $\frac{\epsilon^2}{10C\log(\frac{n\phi'}{\epsilon^3})}$-terminal sparse, we must have $|(U \cup N_G(U)) \cap T| \geq \frac{10Ck'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})$, which in turn implies $|U \cap T| \geq \frac{9Ck'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})$ since $|N_G(U)| = k'$.

Now suppose that U is not ϕ'-sparse. Again, we pick a second set of terminals T' as before. After picking the set T', we have $\mathbb{E}[U \cap T'] = \frac{C\phi'|U|}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3}) \leq \frac{Ck'}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$ where the inequality holds since we assumed that U is not ϕ'-sparse. This in turn means that with probability at least $\frac{1}{2}$ we have $|U \cap T'| \leq \frac{2Ck'}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$. As in the previous analysis, we have $\Pr[E_2] \leq 2\Pr[E_2']$, where E_2' is the event that there exists such a set U which satisfies $|U \cap T| \geq 9\frac{Ck'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})$ and $|U \cap T'| \leq \frac{2Ck'}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$.

We now bound $\Pr[E_2']$. We analyze this in a similar way, imagine picking $X = T \cup T'$ together and randomly deciding which elements form T and which form T'. We will bound $\Pr[E_2'|X]$ for a fixed X. Define $R_2 = \{W \cap X \mid W \in \mathcal{S}_2\}$. By the Sauer-Shelah Lemma, $|R_2| \leq dk'\binom{|X|}{dk'}$. Define E_2'' to be the event that there exists an $r \in R_2$ that satisfies $r \cap T \geq 9\frac{Ck'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})$ and $|r \cap T'| \leq \frac{2Ck'}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$. As in the previous case, we have $\Pr[E_2'] \leq \Pr[E_2'']$, so it is enough to bound $\Pr[E_2'']$.

Again, there are two cases. First, suppose that $|r| \geq \frac{5Ck'}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$. Then $\mathbb{E}[|T' \cap r|] = \frac{|T'|}{|X|}|r| = (1-\epsilon)|r| \geq \frac{5Ck'}{2\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$. It follows that if E_2'' occurs then $|T' \cap r|$ deviates from its mean by more than a multiplicative factor of $\frac{1}{10} \geq \epsilon$, and hence this happens with probability at most $2e^{-\frac{\epsilon^2}{3}\frac{Ck'}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})}$.

Otherwise, we must have $|r| \leq \frac{5Ck'}{\epsilon^3}\log(\frac{n\phi'}{\epsilon^3})$. Now we obtain $\mathbb{E}[|T \cap W|] \leq \frac{5Ck'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})$. However, for E_2'' to occur we must have $|T \cap r| \geq \frac{9Ck'}{\epsilon^2}\log\frac{n\phi'}{\epsilon^3}$. This implies that $|T \cap r|$ deviates from its expectation by a multiplicative factor $\delta > 1$. By the Chernoff bound, we have that this happens with probability at most $e^{-\frac{\delta\mu}{3}} \leq e^{-\frac{4Ck'}{3\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})}$.

By the Sauer-Shelah Lemma $|R_2|$ is at most $dk'\binom{|X|}{dk'}$. Using the fact that $\binom{|X|}{dk'} \leq (\frac{|X|e}{dk'})^{dk'}$, we obtain that this is at most $(\frac{n\phi'}{\epsilon^3})^{10dk'}$. Together, after applying a union bound on these sets, the failure probability is at most $10e^{-\frac{C}{3}k'\log\frac{n\phi'}{\epsilon^3}}(\frac{n\phi'}{\epsilon^3})^{10dk'} < \frac{1}{100k'^2}$ for a large enough choice of C.

Finally, combining both the cases and applying a union bound on all sizes k' between 1 and n, we obtain that the probability of failure is at most $\frac{1}{10}$, and hence T is a $(20\epsilon, \phi')$ sample set with probability at least $\frac{9}{10}$. □

4.3 Vertex cuts – lower bound

A natural question is if the $\log(\frac{n\phi'}{\epsilon^3})$ factor in the size of the sample set is avoidable for vertex cuts, just as in the case of the edge version. We show that this is not the case. The next lemma shows that this result for vertex sample sets is essentially tight using a connection with Ramsey numbers.

Lemma 4.8. *There exists a family of graphs $\mathcal{F}$ and parameters ϕ such that for any constant $\epsilon \in (0,1)$ any (ϵ, ϕ) vertex sample set T must satisfy $T = \Omega(N\phi\log(N\phi))$ for an N-vertex graph from $\mathcal{F}$.*

Proof. Consider the complete graph K_n on n vertices. Now construct the vertex-edge incidence graph of this graph. Formally, consider the bipartite graph H with vertex set $L \cup R$. The set R contains a vertex x_w for every vertex $w \in V(K_n)$, and the set L contains a vertex y_{uv} all pairs of vertices $u, v \in V(K_n)$. For every vertex y_{uv}, add the edges (x_u, y_{uv}) and (x_v, y_{uv}) to $E(H)$. Let $k = \lfloor\frac{\log n}{2}\rfloor$, $\phi = \frac{k}{\binom{k}{2}+k} = \frac{2}{k+1}$ and choose $s = \binom{k}{2}$. We will let $R(k,k)$ denote the (Ramsey) number such that any graph on $R(k,k)$ vertices must have either a clique of size k or an independent set of size k. It is well known that $R(k,k) \leq 4^k$ while a recent breakthrough lowered this upper bound to $(4-\epsilon)^k$ for some absolute constant $\epsilon > 0$ [12].

Let T be a (ϵ, ϕ) sample set for H. Notice that H has $N = \binom{n}{2} + n$ vertices. We will show that $|T| = t \geq \frac{1}{2}\binom{n}{2}$. Clearly $\frac{1}{2}\binom{n}{2} = \Omega(N\phi\log(N\phi))$ since $\phi = \mathcal{O}(\frac{1}{\log n})$. For the sake of contradiction suppose that $t \leq \frac{1}{2}\binom{n}{2}$. In K_n, T corresponds to a set of vertices and edges. Let G' be the resulting graph after removing from K_n the "edges" of T. Since G' has $n \geq 4^k \geq R(k,k)$ vertices, it follows that there is a set of k vertices in G' that form either an independent set of size k or a clique of size k. In H, this corresponds to a set $Y \subseteq R$ of k right vertices. Observe that after removing Y, the $\binom{k}{2}$

left vertices (call them U) corresponding to the edges of K_n between vertices of Y, are disconnected from the rest of the graph. Also note that $H \setminus (U \cup Y)$ is connected. Thus U is the union of all connected components except one after removing Y. Further, U has sparsity $\frac{k}{\binom{k}{2}+k} = \phi$. It follows that U must satisfy $|\frac{N}{|T|}|U \cap T| - |U|| \leq \epsilon|U|$.

There are two cases. First, suppose that Y corresponds to a clique in G'. This means that no vertex of U is in T. But since $|\frac{N}{|T|}|U \cap T| - |U|| \leq \epsilon|U|$ this cannot happen for any T with $|T| > 0$ and hence we obtain a contradiction. So now suppose that Y corresponds to an independent set in G'. But this means every vertex of U is in T, and hence $|U \cap T| = |U|$. Since $|T| \leq \frac{1}{2}\binom{n}{2} \leq \frac{N}{2}$, U cannot satisfy $|\frac{N}{|T|}|U \cap T| - |U|| \leq \epsilon|U|$, and we have a contradiction again. □

5 Small Set Expansion and Vertex Sparsest Cut via LP rounding

In this section, we will use an extension of an existing LP-based algorithm combined with sample sets for sparse cuts in order to obtain an $\mathcal{O}(\log k)$-approximation algorithm for small set expansion and an $\mathcal{O}(\log k + \log\log n\phi)$-approximation algorithm for Vertex Sparsest Cut.

5.1 Small Set Expansion

We start by defining another problem which we call Small Set Terminal Expansion.

Small Set Terminal Expansion	**Parameter:** ϕ, s
Input: Graph G and set of terminals $T \subseteq V(G)$	
Question: Find a set of vertices S' with $\lvert S' \cap T\rvert \leq s$, and $\lvert\delta_G(S')\rvert \leq \phi\lvert S' \cap T\rvert$, or report that no such set exists.	

Definition 5.1 ((α,β)-approximation)**.** *We say that an algorithm is an (α,β)-approximation for* Small Set Terminal Expansion *if given parameters ϕ, s it outputs a set $S' \subseteq V(G)$ with $|S' \cap T| \leq \beta s$ and $|\delta_G(S')| \leq \alpha\phi|S' \cap T|$, or outputs that there is no set $S \subseteq V(G)$ with $|S \cap T| \leq s$ and $|\delta_G(S)| \leq \phi|S \cap T|$.*

When $T = V(G)$ we get the Small Set Expansion problem. The following result from [29] gave an $\mathcal{O}(\log s)$-approximation for Small Set Expansion.

Theorem 5.2 (see [29])**.** Small Set Expansion *admits an $(\mathcal{O}(\log s), \mathcal{O}(1))$-approximation.*

As described in the introduction, our goal will be to use an "$\mathcal{O}(\log s)$-type"-approximation algorithm for the terminal version, so we need an algorithm for Small Set Terminal Expansion. The ideas remain similar to Small Set Expansion. We follow the ideas of [29] with minor changes to obtain this. For completeness' sake, we give a simple and concise algorithm for Small Set Terminal Expansion based on ideas similar to that in [29].

This is deferred to the Appendix (Theorem A.1). In fact, Theorem A.1 is a stronger result, and the approximation algorithm for Small Set Terminal Expansion will follow directly as a corollary.

Lemma 5.3 (Follows from Theorem A.1)**.** Small Set Terminal Expansion *admits an $(\mathcal{O}(\log s), \mathcal{O}(1))$-approximation.*

Henceforth, we will assume that there indeed exists a set of vertices S with $|S \cap T| = s$ and $\delta_G(S) = \phi|S|$ (if not, it is easy to see that our algorithm will detect that this is the case). Let us denote by $k = \phi s$ the cut-size of S.

We now combine the notion of sample sets with Lemma 5.3 to obtain an $\mathcal{O}(\log k)$-approximation for Small Set Expansion in polynomial time.

Proof of Theorem 1.2. We proceed as follows. Choose an arbitrary constant $\epsilon \in (0, \frac{1}{100})$ and construct an (ϵ, ϕ') sample set T of size $\min\{n, \frac{dn\phi'}{\epsilon^2}\}$ where d is the constant hidden in the big-Θ notation of Lemma 4.2 and $\phi' = \frac{ck \log k}{s} = c\phi \log k$, for some large enough constant c which we will choose later. Henceforth we assume that $\frac{dn\phi'}{\epsilon^2} < n$ since otherwise we would have $s = \mathcal{O}(k \log k)$ and we can simply set the terminal set to be the whole vertex set to obtain an $\mathcal{O}(\log s) = \mathcal{O}(\log k)$-approximation algorithm from Theorem 5.2.

Now recall that we are promised that there exists a set S of size s with expansion ϕ. The sample set guarantee means that $\frac{n}{|T|}|S \cap T| = \Theta(|S|)$. This in turn implies that $|S \cap T| = \Theta(|S|\frac{|T|}{n})$ which gives $|S \cap T| = \Theta(|S|dc\phi \log k) = \Theta(dck \log k) = s'$ (say).

Run the algorithm A for Small Set Terminal Expansion from Theorem 5.2 with the parameter s'.

Notice that $|S \cap T| = \Theta(dck \log k)$, hence $\phi_T(S) \leq \mathcal{O}(\frac{1}{dc \log k})$. Then the algorithm must return a set U that is $\mathcal{O}(\log s')\phi_T(S) = \mathcal{O}(\log k \cdot \frac{1}{dc \log k}) = \mathcal{O}(\frac{1}{dc})$-terminal sparse, and $|U \cap T| \leq \mathcal{O}(s')$. Choose c large enough such that $\phi_T(U) \leq \frac{\epsilon^2}{100d} \leq \frac{n}{10|T|}\phi'$.

The sample set guarantee from Definition 4.1 implies that $|U| \in [1-\epsilon, 1+\epsilon]\frac{n}{|T|}|U \cap T|$. Since $\frac{n}{|T|} = \mathcal{O}(\frac{s}{k \log k})$ and $|U \cap T| \leq \mathcal{O}(s') = \mathcal{O}(k \log k)$, we must have $|U| \leq \mathcal{O}(s)$. Also, U is $(\frac{\epsilon^2}{100d})$-terminal sparse, so by the sample set guarantee, U must be $\mathcal{O}(\frac{\epsilon^2}{100d}\frac{|T|}{n})$-sparse in G. But $\frac{|T|}{n} = \mathcal{O}(\phi') = \mathcal{O}(d\phi \log k)$ and hence it must be the case that U is $\mathcal{O}(\phi \log k)$-sparse in G. ∎

5.2 Vertex Sparsest Cut

In this section we obtain an $\mathcal{O}(\log k + \log\log n\phi)$-approximation for Vertex Sparsest Cut, proving Theorem 1.7. Again, we start with the following theorem which approximates terminal sparsity. The proof again uses techniques from [29] and [7], except that now there is no small set guarantee: the result is only for Vertex Sparsest Cut. We remark that our algorithm is also similar to the algorithm of [40].

Theorem 5.4. *Given a graph G with a set of terminals T, suppose that there is a vertex cut (L, C, R) with $|R \cap T| \geq |L \cap T| = s$, $|C| = k$ and terminal sparsity $\phi = \frac{k}{s+|C \cap T|}$. Further suppose that $s \geq k \log s$ (so that $\phi \leq \mathcal{O}(\frac{1}{\log s})$). Then there is a polynomial time algorithm that finds a cut (L', C', R') with terminal sparsity $\phi' = \frac{|C'|}{\min\{|(L' \cup C') \cap T|, |(R' \cup C') \cap T|\}} \leq \mathcal{O}(\phi \log s)$.*

We postpone the proof of this theorem to the appendix. We are now ready to prove Theorem 1.7.

Proof of Theorem 1.7. We proceed as follows. Suppose there exists a vertex cut (L, C, R) with $|C| = k$, and $|L \cup C| = \frac{k}{\phi} = s + k$, so that $|L| = s$. Let $\phi' = \lambda\phi(\log k + \log\log n\phi)$ where λ is a large constant that will be chosen later. Choose $\epsilon = \frac{1}{100}$ and construct a (ϵ, ϕ') vertex sample set T of size $\min\{n, \frac{Dn\phi'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3})\}$ where $\phi' = \lambda\phi(\log k + \log\log n\phi)$ and D is the constant hidden in the big-Theta notation of Lemma 4.7. Henceforth, we assume $\frac{Dn\phi'}{\epsilon^2}\log(\frac{n\phi'}{\epsilon^3}) < n$, for if not, then it follows that $s = \mathcal{O}(k(\log k + \log\log n\phi)\log(n\phi'))$ - then applying the result of Theorem 5.4 with the terminal set $T = V(G)$ clearly gives us an $\mathcal{O}(\log k + \log\log n\phi)$-approximation.

Observe that by the guarantee of the sample sets, we must have $|L \cap T| = \Theta(\lambda Dk(\log k + \log\log n\phi)\log n\phi')$ (we omit the dependence on ϵ henceforth). It follows that L is $\mathcal{O}(\frac{1}{\lambda D(\log k + \log\log n\phi)\log n\phi'})$-terminal sparse. Now we run the algorithm from Theorem 5.4 to obtain another vertex cut

(L', C', R') with $|R' \cap T| \geq |L' \cap T|$ which is ψ-terminal sparse where

$$\psi = \mathcal{O}(\frac{1}{\lambda D(\log k + \log\log n\phi)\log n\phi'}\log|L \cap T|) = \mathcal{O}(\frac{1}{\lambda D \log n\phi'}).$$

Choose λ large enough so that $\psi \leq \frac{\epsilon^2}{10D\log\frac{n\phi'}{\epsilon^3}} \leq \frac{n}{10|T|}\phi'$. Let $s' = |L' \cap T|$ and $s'' = \frac{n}{|T|}s'$.

There are two cases. Fix a constant $\alpha > 1$ which will be chosen later. First, suppose no component in $G \setminus C'$ has size $> n - \frac{s''}{\alpha}$. Then it is clear that we can partition $V(G) \setminus C'$ into $A \cup B$ such that both A and B have $\Omega(s'')$ vertices.

Now suppose there is indeed a component X which has size $> n - \frac{s''}{\alpha}$. We will obtain a contradiction. Clearly, it must be the case that $V(G) \setminus (C' \cup X)$ has size $< \frac{s''}{\alpha}$. But then $V(G) \setminus (C' \cup X)$ is the set of all but one component after removing a vertex cut. Also clearly, either $L \subseteq V(G) \setminus (C' \cup X)$ or $R \subseteq V(G) \setminus (C' \cup X)$, and hence $V(G) \setminus (C' \cup X)$ contains at least s' terminals. Then since $V(G) \setminus (C' \cup X)$ is $\frac{n}{10|T|}\phi'$-terminal sparse, $V(G) \setminus (C' \cup X)$ must satisfy the sample condition. But this means $|V(G) \setminus (C' \cup X) \cap T| \leq 2\frac{|T|}{n}\frac{s''}{\alpha} \leq \frac{2s'}{\alpha}$, which is a contradiction for $\alpha > 2$.

Thus we must in fact have that every component in $G \setminus C'$ has size $\leq n - \frac{s''}{\alpha}$, and hence we obtain a partition of $V(G) \setminus C'$ into $A \cup B$ such that both A and B have $\Omega(s'')$ vertices. Since (L', C', R') was $\frac{n}{10|T|}\phi'$-terminal sparse, we must have $|C'| \leq \frac{n}{10|T|}\phi'|(L' \cup C') \cap T| \leq \mathcal{O}(\frac{n}{10|T|}\phi'|L' \cap T|) = \mathcal{O}(\frac{n}{|T|}\phi's') = \mathcal{O}(s''\phi')$. It follows that the vertex cut (A, C, B) must be $\mathcal{O}(\phi')$-sparse. Hence we obtain an $\mathcal{O}(\frac{\phi'}{\phi}) = \mathcal{O}(\log k + \log\log n\phi)$-approximation. □

6 Sparsest Cut via a cut-matching game for small set expanders

In this section, we develop a cut-matching game for small set expanders. This will be our main ingredient to obtain an $\mathcal{O}(\log^2 k)$-approximation for Sparsest Cut and an $\mathcal{O}(\log^2 k + \log^2\log n\phi)$-approximation for Vertex Sparsest Cut in time $m^{1+o(1)}$, proving theorems 1.3 and 1.8. We will use the analysis of [36]. On a high level, the framework of [36] constructs an expander using a two-player cut-matching game in $\mathcal{O}(\log n)$ rounds. We give a cut matching game that constructs *s-small set expanders*: these are graphs where every set of size at most s is expanding. We show that there is a cut-matching game that can construct such expanders in only $\mathcal{O}(\log s)$ rounds.

We begin by reviewing the cut-matching framework of [36]. The cut-matching game is a two-player game, with a cut player $\mathcal{C}$ and a matching player $\mathcal{M}$. The cut and matching players take alternate turns. The game proceeds in rounds, where each round is a turn of the cut player followed by the matching player. Initially, the game starts with an edgeless graph H on n vertices. In each round, the cut player $\mathcal{C}$ outputs a bisection $(B, \overline{B})$ with $|B| = |\overline{B}| = \frac{n}{2}$. The matching player then chooses a perfect matching from B to $\overline{B}$, which is then added to the graph H. We have the following theorem from [36].

Theorem 6.1 (Follows from [36])**.** *There is a (exponential time) cut player strategy so that for any matching player strategy, the graph H is an $\Omega(1)$-expander after $\mathcal{O}(\log n)$ rounds.*

On a high level, this strategy is quite simple: in each round, the cut player $\mathcal{C}$ finds a cut $(A, \overline{A})$ with $\frac{n}{4} \leq |A| \leq |\overline{A}|$ in H with expansion (sparsity) at most $\frac{1}{100}$. This cut is then extended to an arbitrary bisection $(B, \overline{B})$ such that $A \subseteq B$.

Now we briefly recall how the cut-matching game is used to approximate Sparsest Cut. As discussed in the Overview, this essentially involves implementing a matching player strategy.

Throughout, we assume that a parameter ϕ' is given. The right value of ϕ' will be found by binary search. The matching player $\mathcal{M}$ then, in each round, given a bisection $(B, \overline{B})$ by the cut player, outputs either (i) a ϕ'-sparse cut in G, or (ii) a perfect matching M matching vertices of B to those of $V(H) \setminus B$ such that $M \preceq^{flow} \frac{G}{\phi'}$ (one unit of demand across each edge of M can be routed simultaneously in G with congestion $\frac{1}{\phi'}$). The matching M is added to $E(H)$.

It follows that if the matching player did not return a ϕ'-sparse cut in any of these rounds, $H \preceq^{flow} \mathcal{O}(\frac{\log n}{\phi'})G$. Since H must be a $\Omega(1)$-expander using Theorem 6.1, we obtain that G must be a $\Omega(\frac{\phi'}{\log n})$-expander. This gives an $\mathcal{O}(\log n)$-approximation.

We use a modified cut-matching game to get an $\mathcal{O}(\log^2 s)$-approximation for (terminal) sparse cuts. In contrast to the [36] cut-matching game which guarantees that the graph H is expanding after $\mathcal{O}(\log n)$ rounds, we give a cut player strategy to construct a *small set expander* that ensures that every *set of size at most* s is expanding after $\mathcal{O}(\log s)$ rounds. We remark that there is a slight technical difference in our cut-matching game. In each round, the cut player, instead of returning a single bisection $(B, \overline{B})$, outputs a collection $\mathcal{C}$ of constantly many pairs of disjoint sets (X, Y) with $|X| = |Y|$, where $X \cup Y$ may not be the whole vertex set $V(H)$. The matching player then finds a perfect matching between X and Y for every $(X, Y) \in \mathcal{C}$. Under this setting, we show the following results.

Theorem 6.2. *There is a near-linear time cut player strategy so that after $\mathcal{O}(\log s)$ rounds, for any matching player strategy, every set with at most s vertices in H has expansion $\Omega(\frac{1}{\log s})$.*

We also show how to improve this result if we are allowed exponential time cut player strategies. We do not use this result for approximating sparsest cut, but just obtain this as a result for the cut-matching game itself, generalizing the result of [36].

Theorem 6.3. *There is a (exponential time) cut player strategy so that after $\mathcal{O}(\log s)$ rounds, for any matching player strategy, every set with at most s vertices in the graph H has expansion $\Omega(1)$.*

Finally, we also show how to use this version of cut-matching game to approximate small sparse cuts. We will need this for the slightly more general case of terminal expansion.

Theorem 6.4. *Given a parameter ϕ', a graph G and a set of terminals T, one can find in almost-linear time either (a) a ϕ'-terminal sparse cut or (b) conclude that every set with $\leq s$ terminals is $\frac{\phi'}{\log^2 s}$-terminal-expanding.*

The next theorem is for the vertex version.

Theorem 6.5. *Given a parameter $\phi' \leq \frac{1}{2}$, a graph G and a set of terminals T, one can find in almost-linear time either (a) a ϕ'-terminal vertex sparse cut or (b) conclude that every set with $\leq s$ terminals is $\frac{\phi'}{\log^2 s}$-terminal-vertex expanding. Formally, for every set L with at most s-terminals, we have $|N_G(L)| \geq \frac{\phi'}{\log^2 s}|(L \cup N_G(L)) \cap T|$.*

The main goal of the rest of this section will be to prove theorems 6.2, 6.3, 6.4 and 6.5.

6.1 Cut player

The goal of this section will be to prove Theorem 6.2 and Theorem 6.3. In this entire section, we will assume that ρ is a large enough constant, which will be chosen based on the analysis - we do not attempt to optimize ρ.

The game starts similarly with the empty graph H on the vertex set $V(H)$. Before we proceed, we note that as we will show later, our cut-matching game will run for at most $d \log s$ rounds for some constant d not depending on ρ. Also, throughout this section, we indicate all dependencies on ρ in big-O and big-Ω notations: hence we will assume that ρ can be chosen bigger than any constant hidden in asymptotic notations. Throughout the algorithm, after every round, we will ensure that the graph H remains regular.

First, we will need the following two results which give a Cheeger-type approximation for finding balanced sparse cuts. On a high level, both these algorithms either find a balanced low conductance cut in H, or find a small set that overlaps all low conductance cuts significantly.

Theorem 6.6 ([52]). [5] *Given parameters $b \in (0, \frac{1}{2})$, ψ and a regular graph H, there exist constants d and d' (which may depend on b) and an algorithm that runs in time $\tilde{\mathcal{O}}(\frac{m}{\psi})$ that with high probability either*

1. *Outputs a $\Omega_b(1)$ balanced cut with conductance at most $d'\sqrt{\psi}$, or*
2. *Outputs a set Y' with $|Y'| \leq \frac{bn}{4}$ such that $\frac{|Y' \cap Z|}{|Z|} \geq \frac{5}{8} - \frac{b}{4}$ for every set of vertices Z with $|Z| \leq \frac{n}{2}$ that has* conductance *smaller than $d\psi$.*

A similar but stronger result can be obtained easily, if we allow exponential time.

Lemma 6.7. *Given parameters $b \in (0, \frac{1}{2})$, ψ and a regular graph H, there exists an algorithm that runs in exponential time that either*

1. *Outputs a $\frac{b}{4}$-balanced cut with conductance at most ψ, or*
2. *Outputs a set Y' with $|Y'| \leq \frac{b}{4}n$ such that $\frac{|Y' \cap Z|}{|Z|} \geq 0.6$ for every set of vertices Z with $|Z| \leq \frac{n}{2}$ that has* conductance *smaller than $\frac{\psi}{3}$.*

Proof. Check if there is a $\frac{b}{4}$-balanced cut with conductance at most ψ. If yes, return it. Otherwise, find the largest $b' < \frac{b}{4}$ for which there is a b'-balanced cut with conductance at most ψ. Suppose this cut is $(Y', \overline{Y'})$ with $|Y'| \leq |\overline{Y'}|$. Now we claim that for any $\leq \frac{\psi}{3}$ conductance cut $(Z, \overline{Z})$, with $|Z| \leq |\overline{Z}|$ it must be the case that $|Z \cap Y'| \geq 0.6|Z|$.

Suppose this is not the case. Consider the union $Z \cup Y'$. We will show that $Z \cup Y'$ has conductance at most ψ, contradicting the fact that Y' is a most balanced cut with conductance at most ψ. First note that $|\delta_H(Z \cup Y')| \leq |\delta_H(Z)| + |\delta_H(Y')|$. We also have $\mathrm{vol}_H(Z \cup Y') = \mathrm{vol}_H(Y') + \mathrm{vol}_H(Z \setminus Y')$. Since $|Y' \cup Z| \leq \frac{bn}{4} + \frac{bn}{4} = \frac{bn}{2} \leq \frac{n}{2}$, we must have $\mathrm{vol}_H(Y' \cup Z) \leq \mathrm{vol}_H(V(H) \setminus (Y' \cup Z))$. Thus the conductance of $Z \cup Y'$ is

$$\frac{|\delta_H(Z \cup Y')|}{\mathrm{vol}_H(Z \cup Y')} \leq \frac{|\delta_H(Z)| + |\delta_H(Y')|}{\mathrm{vol}_H(Z \setminus Y') + \mathrm{vol}_H(Y')} \leq \min\left(\frac{|\delta_H(Y')|}{\mathrm{vol}_H(Y')}, \frac{|\delta_H(Z)|}{\mathrm{vol}_H(Z \setminus Y')}\right)$$

The first term is at most ψ, since Y' has conductance at most ψ. The second term is at most $\frac{\psi}{3 \cdot 0.4} \leq \frac{\psi}{1.2} \leq \psi$, since $\mathrm{vol}_H(Z \setminus Y') \geq 0.4 \mathrm{vol}_H(Z)$. This means that $Z \cup Y'$ has conductance at most ψ, a contradiction. ∎

Before we describe the cut player strategy, we need a key subroutine which we describe next. Let $S \subseteq V(H)$ with $|S| \leq \frac{|V(H)|}{2}$ be any set of vertices of H, which is $\frac{1}{\rho^{10\mathrm{I}}}$-sparse in H. We now describe a procedure, which we call ImproveCut, that when given a sparse cut in H, uses a single

[5] The actual theorem is a little different, but this slightly more precise version follows from the proof.

maxflow call to obtain a cut which is sparse within S, for any such S. We describe this procedure first by using exact s-t min-cut for the sake of clarity: it will subsequently be clear that one can use a slightly relaxed notion of min-cuts as well. This is necessary to get a near-linear time cut-player, since the best known exact s-t max-flow/min-cut algorithm [13] runs in almost-linear time.

Algorithm 1 ImproveCut

Input: A cut $(W, \overline{W})$ in H that is $\Omega(1)$ balanced. Also, $\phi_H(W) \leq \frac{1}{100\rho^3}$.

Output: A cut $(Q, \overline{Q})$ that is $\Omega(1)$ balanced such that $|\delta_{H[S]}(Q \cap S)| \leq \frac{|S|}{100\rho}$ for every S that is $\frac{1}{\rho^{101}}$-sparse in H.

Consider the graph obtained by gluing the edges $\delta_H(W)$ to the induced subgraph $H[W]$. In this graph, for every edge $e = \{u, v\} \in \delta_H(W)$ with $u \in W$, create a new vertex x_e. Remove the edge $\{u, v\}$ and add the edge $\{u, x_e\}$. Denote the resulting graph as H'.

We set up a flow problem on the modified graph H'. Add two new vertices s, t and add the edges (s, x_e) for each vertex x_e, each with capacity 1. Add edges (v, t) for every $v \in W$, each with capacity $\frac{1}{\rho^2}$. Each edge of $E(H')$ has capacity 1.
Find a maximum s-t flow and a corresponding min-cut M, where M is the set of vertices reachable from s in the min-cut. Let $R = M \setminus (\{x_e \mid e \in \delta_H(W)\} \cup \{s\})$ and let $\overline{Q} = \overline{W} \cup R$, $Q = V(H) \setminus \overline{Q}$.
return $(Q, \overline{Q})$

Lemma 6.8. *The procedure* ImproveCut *is sound. That is,* $(Q, \overline{Q})$ *is* $\Omega(1)$ *balanced and* $\delta_{H[S]}(Q \cap S) \leq \frac{|S|}{100\rho}$ *for every* S *that is* $\frac{1}{\rho^{101}}$*-sparse in* H.

Proof. Notice that since $\phi_H(W) \leq \frac{1}{100\rho^3}$ the number of edges in the cut $\delta_H(W)$ is at most $\frac{n}{100\rho^3}$. Hence the max flow is always at most $\frac{n}{100\rho^3}$, and thus the number of edges (v, t) which can be cut for some $v \in V(H')$ is at most $\frac{n}{100\rho}$. This means that both Q and $V(H) \setminus Q$ have $\Omega(n) - \frac{n}{100\rho} = \Omega(n)$ vertices.

Suppose for contradiction that Q is not sparse in S. In particular, suppose that $|\delta_{H[S]}(Q \cap S)| \geq \frac{|S|}{100\rho}$. Observe that we can assume that the min-cut M does not cut any edges of the form (x_e, u) where $u \in W$, for otherwise, we can construct another mincut M' which cuts the edge (s, x_e) instead of the edge (x_e, u). This means that $|\delta_{H[S]}(Q)| = |\delta_{H[S]}(M)|$. Since the min-cut is saturated in any max-flow, the edges in $\delta_{H[S]}(M)$ must all be saturated in any max-flow. If $|\delta_{H[S]}(M)| \geq \frac{|S|}{100\rho}$, since the total capacity of the edges from S to the vertex t (the edges (v, t) for $v \in S$) is at most $\frac{|S|}{\rho^2}$, there is an excess flow of $\frac{|S|}{100\rho} - \frac{|S|}{\rho^2} \geq \frac{|S|}{200\rho}$ which must leave the set $S \cap Q$. However $|\delta_H(S)| \leq \frac{|S|}{\rho^{101}}$, and hence this is impossible. Thus we conclude $|\delta_{H[S]}(Q \cap S)| = |\delta_{H[S]}(M)| \leq \frac{|S|}{100\rho}$. ☐

To obtain an algorithm with nearly linear running time, we modify ImproveCut to use the notion of fair minimum cuts of Li et al. [43] instead of an exact min-cut algorithm. An s-t α-fair min cut in the graph H is an s-t cut Z so that there exists a feasible s-t flow that saturates $\frac{1}{\alpha}$ fraction of each cut edge $\delta_H(Z)$. The result of [43] shows that there exists an algorithm that runs in time $\tilde{\mathcal{O}}(\frac{m}{\epsilon^3})$ and computes a $(1+\epsilon)$- fair s-t min-cut Z. It is easy to check that the above proof still works by replacing the exact minimum cut M with the fair min-cut Z for a small enough choice of ϵ and similarly guarantees $|\delta_{H[S]}(Z)| \leq \frac{|S|}{100\rho}$.

Equipped with this result, we are now ready to describe the cut player strategy in each round. Since our strategies for both the near-linear time cut player and the exponential time cut player are similar, we describe both of them together.

Algorithm 2 Cut player strategy

Input: A regular graph H with degree Δ.

Output: A collection $\mathcal{C}$ of constantly many pairs of disjoint vertices (X, Y) with $|X| = |Y|$.

We start by setting $\psi = \frac{1}{10^4 d'^2 \rho^8 \Delta^2}$ or $\psi = \frac{1}{10^4 d'^2 \rho^8 \Delta}$, $b = \frac{1}{100}$ and invoking Theorem 6.6 or Lemma 6.7 on the graph H (according to whether we use near-linear time or exponential time respectively).

if Theorem 6.6 or Lemma 6.7 return a set Y' such that $\frac{|Y' \cap Z|}{|Z|} \geq 0.6$ for each Z that has conductance at most $d\psi$ or $\frac{\psi}{3}$ respectively **then**

$Q \leftarrow Y'$

else if Theorem 6.6 or Lemma 6.7 return a $\Omega_b(1)$ balanced cut $(W, \overline{W})$ **then**

$(Q, \overline{Q}) \leftarrow \textsc{ImproveCut}(W, \overline{W})$

Suppose that $\alpha n = |Q| \leq |\overline{Q}|$. Find a covering $\mathcal{R}$ of $\overline{Q}$ into $\mathcal{O}(1)$ many sets of size $|Q|$ each as follows - partition $\overline{Q}$ into sets of size $|Q|$ except for one set which may have size less than Q, and then arbitrarily extend the last set to a set of size $|Q|$.

Add into $\mathcal{C}$ the pair of sets (Q, R) for each $R \in \mathcal{R}$. Also add an arbitrary partition (P_1, P_2) of $V(H) \setminus (R \cup Q)$ with $|P_1| = |P_2|$ into $\mathcal{C}$ (This part is just to make sure H remains regular).

end if

(In both cases) Arbitrarily extend the set Q to obtain a bisection $(B, \overline{B})$ with $Q \subseteq B$ and add it to the collection $\mathcal{C}$.

Notice that the collection $\mathcal{C}$ consists of pairs of sets (X, Y) with $|X| = |Y|$. Some of these pairs are bisections with $X \cup Y = V(H)$, while some are not. Broadly, the goal of the cut-player is to simultaneously "make progress" towards making any arbitrary set $S \subseteq V(H)$ with $|S| \leq \frac{V(H)}{2}$ expanding. This can be achieved in one of two ways: by either including in $\mathcal{C}$ a cut that is unbalanced with respect to S, so that the matching player makes S expanding in this round. Alternatively, this can be done by including in $\mathcal{C}$ a cut that is balanced and sparse with respect to S: then one can use a potential analysis similar to [36] to show that this cannot keep happening, so that S becomes expanding after only a few iterations. The next lemma formalizes this notion and shows that the collection $\mathcal{C}$ indeed does satisfy such a property.

Lemma 6.9. *The collection $\mathcal{C}$ output by the cut player satisfies one of the following two conditions.*

1. *For every subset S of $V(H)$ with $|S| \leq \frac{|V(H)|}{2}$ that is $\frac{1}{\rho^{101}}$-sparse in H, we have that at least one of two following statements holds.*

   - *$\mathcal{C}$ includes a bisection $(B', \overline{B'})$ such that $B' \cup \overline{B'} = V(H)$ so that there is a subset $W \subseteq B'$ such that W is balanced and sparse with respect to S. More precisely, the cut $(W, \overline{W})$ in H is such that (a) $|W \cap S|, |W' \cap S| = \Omega(|S|)$ and (b) W is $\frac{1}{100}$- sparse in $H[S]$. Concretely, $|\delta_{H[S]}(W)| \leq \frac{1}{100} \min(|W \cap S|, |W' \cap S|)$.*
   - *There exists a pair of sets $(X, Y) \in \mathcal{C}$ such that $||X \cap S| - |Y \cap S|| = \Omega(|S|)$, so that any perfect matching of X to Y must add at least $\Omega(|S|)$ edges across $(S, V(H) \setminus S)$.*

2. *$\mathcal{C}$ has exactly one pair of sets $(B, \overline{B})$ which is a bisection, and B contains a set $Y' \subseteq V(H)$ with $|Y'| \leq \frac{|V(H)|}{2}$ such that $\frac{|Y' \cap Z|}{|Z|} \geq 0.6$ for every set of vertices Z that has* conductance *smaller*

than $\frac{1}{\rho^{100}\Delta^2}$ *or* $\frac{1}{3\rho^{100}\Delta}$*, depending on whether we use the exponential time or near-linear time algorithm, where* Δ *is the degree of the current (regular) graph* H*.*

Proof. If Lemma 6.7 or Theorem 6.6 returns a set Y' as in above, then note that this set overlaps by at least 60% every set Z with conductance at most $\frac{1}{3\rho^{100}\Delta}$ if we use the exponential time algorithm or every set Z with conductance at most $\frac{1}{\rho^{100}\Delta^2}$ if we use the near-linear time algorithm, and we are done as this satisfies condition 2.

Otherwise, in both cases, we obtain a cut W which is $\Omega(1)$ balanced and has conductance at most $\frac{1}{100\rho^3\Delta}$. Since the maximum degree in the graph H is Δ, it must be the case that this cut has sparsity at most $\frac{1}{100\rho^3}$.

Recall that in this case, the cut player uses the ImproveCut procedure to obtain another cut $(Q, \overline{Q})$. Now if $|Q \cap S| \geq 0.6|S|$, it follows that the bisection $(B, \overline{B})$ with $Q \subseteq B$ satisfies $||B \cap S| - |\overline{B} \cap S|| \geq \Omega(|S|)$, and hence we satisfy condition 1.

Similarly, suppose that $|\overline{Q} \cap S| \geq (1 - \frac{\alpha}{3})|S|$. Notice that since $|Q| = \alpha n$ and $|\overline{Q}| = (1-\alpha)n$, the number of $R \in \mathcal{R}$ is at most $\frac{1-\alpha}{\alpha} + 1 = \frac{1}{\alpha}$. Then it follows that there exists an $R \in \mathcal{R}$ such that $|R \cap S| \geq (1 - \frac{\alpha}{3})\alpha|S| > \frac{\alpha}{2}|S|$. Since in this case $|Q \cap S| \leq \frac{\alpha}{3}|S|$ it follows that the pair of sets (Q, R) satisfies condition 1.

Finally, if none of the above happens, it must be the case that both $|Q \cap S|, |\overline{Q} \cap S| = \Omega(|S|)$. Thus the cut $(Q, \overline{Q})$ is both balanced and sparse inside S since ImproveCut guarantees that $|\delta_{H[S]}(Q \cap S)| \leq \frac{|S|}{100\rho} \leq \frac{1}{100}\min(|W \cap S|, |W' \cap S|)$. In this case, the bisection $(B, \overline{B})$ added by the cut player with $Q \subseteq B$ satisfies condition 1. ☐

Broadly, our next lemma shows that if we keep finding bisections $(B, \overline{B})$ that contain such balanced sparse cuts $(Q, \overline{Q})$ inside S for too many rounds, then the matching player must add a matching that makes the set S $\Omega(\frac{1}{\rho^{101}})$-expanding in H. The proof is similar to the potential analysis in [36].

Lemma 6.10. *Let* S *be a set of size* s *in* H*. Then, if the cut player keeps finding bisections* $(B, \overline{B})$ *such that there is a cut* $(Q, \overline{Q})$ *with* $Q \subseteq B$ *in* H *that is both balanced and* $\frac{1}{100}$*-sparse with respect to* S *for more than* $\mathcal{O}(\log s)$ *rounds, there must exist a round in which at least* $\Omega(\frac{s}{\rho^{101}})$ *edges were added by the matching player across* $(S, V(H) \setminus S)$*. Formally,* ImproveCut *can find a cut* $(Q, \overline{Q})$ *satisfying* $|Q \cap S|, |\overline{Q} \cap S| = \Omega(s)$ *and* $|\delta_{H[S]}(Q)| \leq \frac{1}{100}\min(|Q \cap S|, |\overline{Q} \cap S|)$ *for at most* $\mathcal{O}(\log s)$ *rounds before the matching player adds a matching that makes* S $\Omega(\frac{1}{\rho^{101}})$*-expanding.*

Proof. As in [36], we will consider a random walk. This time, however, the random walk will only involve vertices from S. For $u, v \in S$, let $p(u, v, t)$ denote the amount of mass which was initially at u that is currently at v after t rounds of the cut-matching game. Initially, $p(u, v, 0) = 1$ if and only if $u = v$, and 0 otherwise. We define the random walk as follows: let M_t be the matching (corresponding to the bisection $(B, \overline{B})$) output by the matching player in round t. For every vertex $a \in S$ that is matched in M_t to another vertex $b \in S$, we average the distributions of a and b. That is, we set $p(u, a, t) = \frac{p(u,a,t-1)+p(u,b,t-1)}{2}$, and set $p(u, b, t) = \frac{p(u,a,t-1)+p(u,b,t-1)}{2}$ for every vertex $u \in S$.

We now define a potential function based on these probability distributions - it is simply the sum of the entropies of the distributions induced by each vertex $u \in S$. Formally, we define

$$\Phi(t) = \sum_{u \in S} \sum_{v \in S} -p(u, v, t) \log p(u, v, t)$$

Clearly, $\Phi(0) = 0$, and $\Phi(t) \leq s \log s$ for any t since the entropy of a distribution on s outcomes is at most $\log s$, and $\Phi(t)$ is the sum of s such entropies. We now show that the potential increases

by a lot in each round, provided that there is $Q \subseteq B$ such that the cut $(Q, \overline{Q})$ is $\Omega(1)$ balanced and $\frac{1}{100}$-sparse with respect to S.

Lemma 6.11. *$\Phi(t) - \Phi(t-1) \geq \Omega(s)$ for any $t \geq 1$.*

Proof. Fix a round t. Recall that the matching player, in this round, finds a perfect matching M_t, between vertices of the bisection $(B, \overline{B})$. In particular, M_t matches vertices of Q to some vertices of $\overline{Q}$, saturating Q. We may assume that M_t matches at most $\frac{s}{\rho^{100}}$ vertices of S to vertices outside S, otherwise S is already $\Omega(\frac{1}{\rho^{101}})$-expanding after this round, and we are done. Also recall that $|Q \cap S| = |Q \cap \overline{S}| = \Omega(s)$. Without loss of generality we assume $|Q \cap S| \leq |\overline{Q} \cap S|$. Let $W^* = Q \cap S$, and let $w^* = |W^*|$. We know that the sparsity of the cut W^* in the graph $H[S]$ is $\phi^{H[S]}(W^*) \leq \frac{1}{100}$.

For a vertex $u \in W^*$, let us denote the total mass that was initially on u that is, at the beginning of round t, either (i) outside W^* or (ii) On some vertex $v \in W^*$ which is matched to some vertex outside S in M_t, by $q_u(t)$. Informally, this mass is the "problematic mass" - it does not help us increase the potential. We first claim that $\sum_{u \in W^*} q_u(t) \leq \frac{w^*}{200} + \frac{s}{\rho^{100}}$. This follows from the fact that W^* is a $\frac{1}{100}$-sparse cut in $H[S]$. Since in every round, at most $\frac{1}{2}$ mass can go out of W^* through each edge of W^*, the total mass that went outside W^* in the rounds prior to t is at most $\frac{w^*}{200}$. Also, there are at most $\frac{s}{\rho^{100}}$ vertices of W^* which are matched to some vertex outside S by M_t. The total mass on these vertices is at most $\frac{s}{\rho^{100}}$, since the algorithm maintains the invariant that every vertex has total mass 1. Therefore it must be the case that $\sum_{u \in W^*} q_u(t) \leq \frac{w^*}{200} + \frac{s}{\rho^{100}} \leq \frac{w^*}{100}$, where the last inequality follows from the fact that $w^* = \Omega(s)$.

By averaging, there exist at least $\frac{w^*}{2}$ vertices u for which $q_u(t) \leq \frac{1}{25}$. Fix such a vertex u. We say that a vertex $v \in W^*$ is "good" for u if (a) v is matched to some vertex $M_t(v) \in S$ and (b) $p(u, v) > 2p(u, M_t(v))$. We now show that the total mass of u that is on the bad vertices must be small.

There are two types of "bad" vertices - the ones which violate condition (a) and the ones which violate condition (b). Let us denote by B_1 the set of vertices $v \in W^*$ matched to some vertex outside S by M_t , and by B_2 the set of vertices $v \in W^*$ which are matched to some vertex $M_t(v) \in S$ but having $p(u, v) \leq 2p(u, M_t(v))$. Observe that $p(u, M_t(v))$ contributes to $q_u(t)$, since $M_t(v)$ is outside W^*. This means that the total mass on the vertices $B_1 \cup B_2$ can be bounded by $2q_u(t)$, simply from the definition of $q_u(t)$. The total mass of u inside W^* is at least $1 - q_u(t) \geq 1 - \frac{1}{25}$. Out of these, at most $2q_u(t) \leq \frac{2}{25}$ mass is on the "bad" vertices. This implies that the total mass on the "good" vertices is at least $1 - \frac{3}{25} = \frac{22}{25}$.

Finally, for every matching edge matching $a, b \in S$ such that $2p(u, a) > p(u, b)$, the entropy of the distribution of u must increase at least by $\Omega(p(u, a))$. This is easy to prove but is already proved in [36] so we skip its proof here. Since the total mass on the good vertices is $\Omega(1)$, the entropy of the distribution for u increases by $\Omega(1)$ after this round. Thus, the overall increase in potential is at least $\Omega(s)$ since there are at least $\Omega(s)$ vertices u for which $q_u(t) \leq \frac{1}{25}$. □

Since the potential increases by $\Omega(s)$ in each round and the total potential is at most $s \log s$, such a cut $(Q, \overline{Q})$ can only be found for $\mathcal{O}(\log s)$ many rounds. □

We are now in a position to prove Theorems 6.3 and 6.2.

Proof. Proof of Theorem 6.3, Theorem 6.2

The above analysis implies that after $\mathcal{O}(\log s)$ rounds, for any set S of size at most s if Improve-Cut keeps returning cuts which are balanced and sparse with respect to S, then the matching player must add $\Omega(\frac{|S|}{\rho^{101}})$ edges across $(S, V(H) \setminus S)$. Then after these $\mathcal{O}(\log s)$ many rounds, there are three cases for any such set S based on the properties of the cut player guaranteed in Lemma 6.9.

We will assume that after these $\mathcal{O}(\log s)$ rounds, $|\delta_H(S)| \leq \frac{|S|}{\rho^{101}}$, for otherwise we are done (for both Theorem 6.3 and Theorem 6.2) since S is already expanding (recall that ρ is a constant).

The first case is when in every round, the cut-player returns a cut which is balanced and sparse with respect to S. In this case, the matching player must have added $\Omega(\frac{|S|}{\rho^{101}})$ edges across $(S, V(H) \setminus S)$ in some round. Therefore if this happens, we are done, as after this round S would have become $\Omega(\frac{1}{\rho^{101}})$-expanding.

The second case is when in some round, the collection $\mathcal{C}$ output by the cut player includes a pair of sets (X, Y) with $||X \cap S| - |Y \cap S|| \geq \Omega(|S|)$. This again forces the matching player to add $\Omega(|S|)$ edges across $(S, V(H) \setminus S)$ in this round, which makes S expanding. Finally, in the third case, the cut player could return a set Y' which overlaps all $\frac{1}{\rho^{100}\Delta^2}$ (or $\frac{1}{3\rho^{100}\Delta}$ if exponential time) conductance cuts by more than 0.6 fraction where Δ is the current degree of the graph H. We analyze this case in the next paragraph.

First, we analyze the exponential time cut player: we will show that S must be $\frac{1}{\rho^{101}}$-expanding after the last round in which the cut player finds the set Y'. Suppose S is $\frac{1}{\rho^{101}}$-sparse just before this last round. Then its conductance is at most $\frac{1}{\rho^{101}\Delta}$. Recall that the set Y' overlaps by 60% every set S which has conductance at most $\frac{1}{3\rho^{100}\Delta} \geq \frac{1}{\rho^{101}\Delta}$. Thus we must have $|Y' \cap S| \geq 0.6|S|$. But the cut player, in this round, outputs a bisection $(B, \overline{B})$ such that $Y' \subseteq B$. This means that S becomes $\Omega(1)$-expanding after this round, and this concludes the proof of Theorem 6.3.

We proceed similarly for the near-linear time cut player. Suppose that S is $\frac{1}{\rho^{101}\Delta}$-sparse before the last round. Then S has conductance at most $\frac{1}{\rho^{101}\Delta^2}$. Recall that the set Y' overlaps by 60% every set S which has conductance at most $\frac{1}{\rho^{100}\Delta^2} \geq \frac{1}{\rho^{101}\Delta^2}$. Thus we must have $|Y' \cap S| \geq 0.6|S|$. But the cut player, in this round, outputs a bisection $(B, \overline{B})$ such that $Y' \subseteq B$. This means that S becomes $\Omega(1)$-expanding after this round. Since we never run the game for more than $\mathcal{O}(\log s)$ rounds, and in each round the matching player can increase the degree of each vertex by only a constant, it follows that $\Delta = \mathcal{O}(\log s)$, and therefore S must be $\Omega(\frac{1}{\rho^{101}\log s})$-expanding. Finally, since ρ is just a (large enough) constant which we can set accordingly using our analysis, we obtain our result in Theorem 6.2. This concludes the proof. □

6.2 Matching player: fast algorithms for Sparsest Cut and Vertex Sparsest Cut

Next, we describe the matching player strategy that helps us approximate sparse cuts. Here, we are given a graph G together with a parameter ϕ', and a set of terminals T. The graph H that the cut-matching game operates on now has the vertex set $V(H) = T$.

Algorithm 3 Matching player strategy

Given a collection $\mathcal{C}$ of constantly many pairs of disjoint sets, for each pair $P_1, P_2 \subseteq V(H)$ with $|P_1| = |P_2|$ given by the cut player, find either

1. a perfect matching M of vertices of P_1 to P_2 that can be flow embed in G with congestion $\frac{1}{\phi'}$ or
2. a ϕ'-terminal sparse cut in G.

Next, we show that the matching player strategy can be implemented using a single max flow. This is very similar to the matching player in [37].

Lemma 6.12. *The matching player strategy can be implemented using a single max flow call.*

Proof. We setup a flow problem as follows. We create two new vertices s, t and add edges from s to each vertex of P_1 with capacity 1. Similarly, we add edges from t to each vertex of P_2 with capacity 1. We retain each edge of G with capacity $\frac{1}{\phi'}$. We now find a maximum flow between s and t. For simplicity, we will assume that $\frac{1}{\phi'}$ is an integer (this assumption is also standard in previous cut-matching games).

If the maximum flow is less than $|P_1|$, the minimum cut in this graph will correspond to a ϕ'-sparse cut in G. This can be seen as follows. Suppose $(W, \overline{W})$ is the cut in G corresponding to the $s-t$ min cut. First note that $W, \overline{W} \neq \emptyset$ as the min-cut is strictly has weight strictly less than $|P_1|$. Observe that in any max-flow the flow is saturated across the minimum cut. But the total flow out of the set W is at most $|W \cap T|$. It follows that $|W \cap T| \geq \frac{|\delta_G(W)|}{\phi'}$. One can show similarly that $|\overline{W} \cap T| \geq \frac{|\delta_G(\overline{W})|}{\phi'}$. It follows that W is a ϕ-terminal sparse cut in G.

Otherwise, the flow path decomposition of the maximum flow gives rise to a perfect matching M. We return this matching M. Using the almost-linear time max-flow algorithm of [13], we can accomplish this in time $m^{1+o(1)}$. □

We now proceed to prove Theorem 6.4.

Proof of Theorem 6.4. We run the cut-matching game with the matching player as described above, with parameter s, and the graph H initialized as the empty graph with the vertex $V(H) = T$. If ever the matching player returns a ϕ'-terminal sparse cut, then we are done. Otherwise, the cut player guarantees than in $\mathcal{O}(\log s)$ rounds the graph H is a s-small set $\Omega(\frac{1}{\log s})$-expander. But since each matching flow embeds in G with congestion $\frac{1}{\phi'}$, it must be the case that H itself flow embeds in G with congestion $\frac{\mathcal{O}(\log s)}{\phi'}$. Now consider a set S with at most s-terminals in G, and let $S_T = S \cap T$. Since S_T is $\Omega(\frac{1}{\log s})$-expanding in H, it means that $|\delta_H(S_T)| = \Omega(\frac{|S_T|}{\log s})$. But the edges $\delta_H(S_T)$ flow embed in G with congestion $\frac{\mathcal{O}(\log s)}{\phi}$. This means that one can send a flow of value $\Omega(\frac{|S_T|}{\log s})$ from S to outside S in G with congestion $\frac{\mathcal{O}(\log s)}{\phi}$. This implies that S must be $\Omega(\frac{\phi}{\log^2 s})$-terminal expanding in G. □

Next, we prove our result for the vertex terminal sparse cuts using a simple modification to the matching player. The matching player strategy is almost exactly the same as before, except that for every pair P_1, P_2 given the cut player we want to find a perfect matching M vertices P_1 to P_2 that embeds in G with *vertex congestion* $\frac{1}{\phi'}$ or a ϕ'-(vertex) terminal sparse cut. Accordingly, we change the flow problem so that every *vertex* has capacity $\frac{1}{\phi'}$.

If the maximum flow is less than $|P_1|$, the minimum (vertex) cut in this graph will correspond to a ϕ'-terminal sparse vertex cut in G. This can be seen as follows. Suppose (L', C', R') is the vertex min-cut in G corresponding to the $s-t$ vertex min cut. First note that $L', R' \neq \emptyset$ as the min-cut has weight strictly less than $|P_1|$. Observe that in any max-flow the flow is saturated across the minimum cut. But the total flow out of the set $L' \cup C'$ is at most $|(L' \cup C') \cap T|$. It follows that $|(L' \cup C') \cap T| \geq \frac{|C'|}{\phi'}$. One can show similarly that $|(R' \cup C') \cap T| \geq \frac{|C'|}{\phi'}$. This shows that (L', C', R') is ϕ'- terminal sparse.

Otherwise, the flow path decomposition of the maximum flow gives rise to a perfect matching M. We return this matching M. Again using the almost-linear time max-flow algorithm of [13] we can accomplish this in time $m^{1+o(1)}$.

Proof of Theorem 6.5. We run the cut-matching game with the matching player as described above and the parameter $2s$ instead of s. Again, the graph H is initialized as the empty graph with the

vertex $V(H) = T$. If ever the matching player outputs a ϕ'-terminal sparse vertex cut, we are done. Otherwise, after $\mathcal{O}(\log s)$ rounds, the cut-matching game certifies that H is a $2s$-small set $\frac{1}{\log s}$-expander, and H embeds in G with vertex congestion $\mathcal{O}(\frac{\log s}{\phi'})$.

Now consider a vertex cut (L, C, R) in G, with $|L \cap T| = s$. Suppose this cut is not $\Omega(\frac{\phi'}{\log^2 s})$-terminal expanding. In particular, we have $|C \cap T| \leq |C| \leq \mathcal{O}(\frac{\phi'}{\log^2 s})|(L \cup C) \cap T| \leq \frac{1}{2}|(L \cup C) \cap T|$ since $\phi' \leq \frac{1}{2}$. This gives $|C \cap T| \leq |L \cap T|$ and hence $|(L \cup C) \cap T| \leq 2s$. By the guarantee of the cut-matching game, $(L \cup C) \cap T$ must be $\Omega(\frac{1}{\log s})$-edge expanding in H. This means $|\delta_H((L \cup C) \cap T)| \geq \Omega(\frac{s}{\log s})$. But the edges $\delta_H((L \cup C) \cap T)$ embed in G with vertex congestion $\mathcal{O}(\frac{\log s}{\phi'})$. This means that we can send $|\delta_H((L \cup C) \cap T)| = \Omega(\frac{s}{\log s})$ units of flow from $L \cup C$ to the rest of the graph with vertex congestion only $\mathcal{O}(\frac{\log s}{\phi'})$, and this in turn means that $|C| \geq \mathcal{O}(\frac{\phi'}{\log^2 s})|(L \cup C) \cap T|$, which shows that (L, C, R) must be $\Omega(\frac{\phi'}{\log^2 s})$-terminal expanding, which is a contradiction. □

We are now ready to prove Theorem 1.3 and Theorem 1.8.

Proof of Theorem 1.3. Suppose that there is a $\phi = \frac{k}{s}$-sparse cut S of size k that separates s vertices. We first guess ϕ, $\log k$ upto factor 2. Let $\phi' = D\phi \log^2 k$ for some large constant D to be fixed later. Next, construct a $(\epsilon, D\phi' \log^2 k)$ sample set T with $\epsilon = \frac{1}{100}$, and a constant D which will be chosen later. Note that T has size $\min\{n, \Theta(n\phi')\} = \min\{n, \lambda n\phi'\}$ for some constant λ. We assume that $\lambda n D\phi' \log^2 k < n$, for otherwise we must have $s = \mathcal{O}(k \log^2 k)$ and we can simply run our cut-matching game with the terminal set as the entire vertex set, so that Theorem 6.4 will give us an $\mathcal{O}(\log^2 s)$-approximation which is already an $\mathcal{O}(\log^2 k)$-approximation. It follows that $|S \cap T| = s' = \Theta(\lambda D k \log^2 k)$. Thus S has terminal sparsity $\mathcal{O}(\frac{1}{\lambda D \log^2 k})$. Running the cut-matching game algorithm of Theorem 6.4 then gives us an $\mathcal{O}(\frac{1}{\lambda D \log^2 k} \log^2(s')) = \mathcal{O}(\frac{1}{\lambda D})$-terminal sparse cut, call it S'. Choose D large enough so that S' is $\frac{1}{10\lambda} \leq \frac{n}{10|T|}\phi'$-terminal sparse, which in turn means that S' must satisfy the sample set condition. By the guarantee of sample sets, it follows that $|S'| = \Omega(\frac{n}{|T|}|S' \cap T|)$. Since S' is $\frac{n}{10|T|}\phi'$-terminal sparse, it follows that this cut must be $\mathcal{O}(\phi') = \mathcal{O}(\phi \log^2 k)$-sparse. □

Proof of Theorem 1.8. Suppose that there is a $\phi = \frac{k}{k+s}$-sparse cut (L, C, R) with $|R| \geq |L| = s$ and $|C| = k$ of size k that separates s vertices. Let $\epsilon = \frac{1}{100}$. We first guess ϕ, k, s upto factor 2. Let $\phi' = D\phi(\log^2 k + \log^2 \log n\phi)$ for some large constant D which will be chosen later. Next, construct a (ϵ, ϕ') (vertex) sample set T with $\epsilon = \frac{1}{100}$ (Notice that this can be done even when we guess s, ϕ' upto factor 2, we skip this minor detail). Note that T has size $\min\{n, \mathcal{O}(n\phi' \log n\phi')\} = \min\{n, \lambda n\phi' \log n\phi'\}$, say. We assume that $\lambda n\phi' \log n\phi' < n$, for otherwise we have $s = \mathcal{O}(k(\log^2 k + \log^2 \log n\phi) \log n\phi')$, and by running our cut-matching game with the vertex set as the terminal set, Theorem 6.5 would give us an $\mathcal{O}(\log^2 s) = \mathcal{O}(\log^2 k + \log^2 \log n\phi)$-approximation. Therefore it follows that $|L \cap T| = s' = \Theta(\lambda k D(\log^2 k + \log^2 \log n\phi) \log n\phi')$. Thus S has terminal sparsity $\mathcal{O}(\frac{1}{\lambda D(\log^2 k + \log^2 \log n\phi) \log n\phi'})$.

Now we run the algorithm from Theorem 6.5 to obtain another vertex cut (L', C', R') with $|R' \cap T| \geq |L' \cap T|$ which is ψ-terminal sparse where

$$\psi = \mathcal{O}(\frac{1}{\lambda D(\log^2 k + \log^2 \log n\phi) \log n\phi'} \log^2 |L \cap T|) = \mathcal{O}(\frac{1}{\lambda D \log n\phi'}).$$

Choose λ large enough so that $\psi \leq \frac{\epsilon^2}{10D \log(n\phi')} \leq \frac{n}{10|T|}\phi'$. Let $s' = |L' \cap T|$ and $s'' = \frac{n}{|T|}s'$.

There are two cases. Fix a constant $\alpha > 1$ which will be chosen later. First, suppose no component in $G \setminus C'$ has size $> n - \frac{s''}{\alpha}$. Then it is clear that we can partition $V(G) \setminus C'$ into $A \cup B$ such that both A and B have $\Omega(s'')$ vertices.

Now suppose there is indeed a component X which has size $> n - \frac{s''}{\alpha}$. We will obtain a contradiction. Clearly, it must be the case that $G \setminus (C' \cup X)$ has size $< \frac{s''}{\alpha}$. Also either $L' \subseteq V(G) \setminus (C' \cup X)$ or $R' \subseteq V(G) \setminus (C' \cup X)$, and hence $V(G) \setminus (C' \cup X)$ contains at least s' terminals. But then since $G \setminus (C' \cup X)$ is the set of all but one component after removing a vertex cut, and further $G \setminus (C' \cup X)$ is $\psi \leq \frac{\epsilon^2}{10d} \leq \frac{n}{10|T|}\phi'$-terminal sparse, $G \setminus (C' \cup X)$ must satisfy the sample condition. But this means $|V(G) \setminus (C' \cup X) \cap T| \leq 2\frac{|T|}{n}\frac{s''}{\alpha} \leq \frac{2s'}{\alpha}$, which is a contradiction for $\alpha > 2$.

Thus we must in fact have that every component in $G \setminus C'$ has size $> n - \frac{s''}{\alpha}$, and hence we obtain a partition of $V(G) \setminus C'$ into $A \cup B$ such that both A and B have $\Omega(s'')$ vertices. It follows that since the vertex cut (A, C, B) is $\mathcal{O}(\frac{n}{10|T|}\phi')$-terminal sparse, it must be $\mathcal{O}(\phi')$-sparse. Hence we obtain an $\mathcal{O}(\frac{\phi'}{\phi}) = \mathcal{O}(\log^2 k + \log^2 \log n\phi)$-approximation. □

7 Small Set Vertex Expansion

In this section we study the approximability of Small Set Vertex Expansion and prove Theorems 1.4 to 1.6.

7.1 Hardness

We start by showing two hardness results for Small Set Vertex Expansion. The first is a reduction from Densest k-Subgraph to Small Set Vertex Expansion which shows n^ϵ hardness for some $\epsilon > 0$ assuming standard hardness results for Densest k-Subgraph. The second is a reduction from Densest k-SubHypergraph, a generalization of Densest k-Subgraph, which shows a strong hardness of approximation result, that Small Set Vertex Expansion is hard to approximate beyond a factor $2^{o(k)}$ even using FPT algorithms, that is, algorithms running in time $f(k)n^{\mathcal{O}(1)}$. This is in sharp contrast to Small Set Expansion, for which we obtained an $\mathcal{O}(\log k)$-approximation (Theorem 1.2).

Proof of Theorem 1.4. Given an instance (G, k) of Densest k-Subgraph, consider the graph H with vertex set $L \cup R$. The set R contains a vertex x_w for every vertex $w \in V(G)$, and the set L contains a vertex y_{uv} for every edge $e = \{u, v\} \in E(G)$. For every vertex y_{uv}, add the edges (x_u, y_{uv}) and (x_v, y_{uv}) to $E(H)$. Finally, add a clique on the vertices R to $E(H)$.

First, suppose that the Densest k-Subgraph objective is $\geq \ell$. Let $R' \subseteq R$ be the vertex set corresponding to the Densest k-Subgraph solution in G, and L' be the set corresponding to the edges inside $G[R']$. Observe that $|L'| \geq \ell$. Clearly, $(L', R', V(H) \setminus (L' \cup R'))$ is a vertex cut of size k in H which separates ℓ vertices.

Now suppose there is a vertex cut in H cutting $k\tau$ vertices which separates at least $\ell\tau$ vertices for some $\tau > 1$. Observe that we may assume that such a cut only cuts vertices of R, since the vertices of R form a clique. Let $L' \subseteq L$ be the set of vertices separated. Sample each vertex in the cut with probability $\frac{1}{2\tau}$ into a set $W \subseteq R$. It follows that the expected number of vertices separated from L after deleting W is at least $\frac{\ell}{4\tau^2}$ since each edge has a $\frac{1}{4\tau^2}$ chance that both its endpoints are in W. Using standard concentration inequalities, it must be the case that with constant probability, we obtain a cut in H cutting $\leq k$ right vertices that separates $\Omega(\frac{\ell}{\tau})$ left vertices.

We now show that if we have a (β, γ) bi-criteria approximation algorithm for Small Set Vertex Expansion, then we obtain an $\mathcal{O}(\beta^2\gamma)$ randomized approximation algorithm for Densest

k-Subgraph. First, construct the graph H as above. Guess the optimal value *opt* for the Densest k-Subgraph instance. One can assume $opt = \Omega(k)$, since one can always find k vertices that induce at least $k/2$ edges. Run the (β, γ)-approximation to Small Set Vertex Expansion with set size *opt*. If it returns a set with size $< opt$, re-run the algorithm till the total number of vertices separated is at least *opt*. Since in each iteration the obtained cut must have sparsity at most $\beta \cdot \frac{k}{k+opt}$, it follows that we obtain a set of size $\mathcal{O}(\tau \cdot opt)$ which is separated from the rest of the graph by a cut of size at most $\mathcal{O}(\beta k \tau)$ for some $\gamma \geq \tau > 1$. We now use random sampling as in above to obtain a set of size k vertices that separates $\Omega(\frac{opt}{\beta^2 \tau})$ vertices. It follows that we obtain an $\mathcal{O}(\beta^2 \tau \leq \beta^2 \gamma)$-approximation for Densest k-Subgraph. Setting $\gamma = \mathcal{O}(1)$ and $\beta = \sqrt{\alpha}$, we obtain the hardness for Small Set Vertex Expansion. □

We next obtain a strong hardness of approximation in terms of k based on Densest k-SubHypergraph defined below.

Definition 7.1 (Densest k-SubHypergraph)**.** *Given a bipartite graph G with vertex set $L \cup R$ as the bipartition, find a set S of k vertices in R such that maximum number of vertices in L have neighbours only in S.*

Notice that this definition is equivalent to the standard definition in terms of hyperedges: Each vertex of L represents a hyperedge, and each vertex of R represents a hypergraph vertex. Let an (α, β)-approximation algorithm for Densest k-SubHypergraph be the one that returns $S \subseteq R$ with $|S| \leq \beta k$ that contains the neighbor set of at least $(1/\alpha) \cdot (|S|/k) \cdot opt$ left vertices (hyperedges), so the ratio between the number of hyperedges and the number of vertices is at least $(1/\alpha)$ times that of the optimal solution.

Lemma 7.2. *An $f(k)\text{poly}(n)$ time (α, β)-approximation algorithm for* Small Set Vertex Expansion *implies an $f(k)\text{poly}(n)$ time $(\frac{\alpha}{1-(\alpha-1)(k/s)}, \frac{\alpha\beta}{1-(\alpha-1)(k/s)})$-approximation for* Densest k-Sub-Hypergraph *for the instance when the optimal has k vertices inducing s hyperedges.*

Proof. The reduction is almost the same as the reduction from Densest k-Subgraph. Given an instance (G, k, L, R) of Densest k-SubHypergraph with vertex set $L \cup R$, add a clique among the vertices R. Call the resulting graph H.

Suppose there is a Densest k-SubHypergraph solution that separates s vertices. It is clear that H has a vertex cut (L', C', R') with $|R'| \geq |L'| \geq s$ and $|C'| = k$. Now suppose there is an algorithm that finds a vertex cut (L'', C'', R'') in H with $|R''| \geq |L''|$, $|L''| \leq \beta s$, and $\frac{|C''|}{|L''|+|C''|} \leq \frac{\alpha \cdot k}{k+s}$. Again, it is clear that we may assume that C'' only cuts vertices of R, and that $L'' \subseteq L$. Since $|C''| \leq \frac{\alpha k}{k+s}|L''|/(1 - \frac{\alpha k}{k+s}) = \frac{\alpha k}{s-(\alpha-1)k}|L''| \leq k \cdot \frac{\alpha\beta}{1-(\alpha-1)(k/s)}$, we obtain a $(\frac{\alpha}{1-(\alpha-1)(k/s)}, \frac{\alpha\beta}{1-(\alpha-1)(k/s)})$-approximation for Densest k-SubHypergraph. □

Therefore, when $\alpha k = o(s)$, an (α, β)-approximation for Small Set Vertex Expansion implies a $(\alpha(1 + o(1)), \alpha\beta(1 + o(1)))$-approximation for Densest k-SubHypergraph.

Manurangsi, Rubinstein, and Schramm [46] introduced the Strongish Planted Clique Hypothesis (SPCH) and proved that assuming the SPCH, there is no $f(k)\text{poly}(n)$-time algorithm for Densest k-SubHypergraph that finds k vertices that induce at least $(1/\alpha(k))opt$ hyperedges, for any function $\alpha(k) = 2^{o(k)}$. We slightly generalize their result to show that there is no $f(k)\text{poly}(n)$-time $(\alpha(k), \beta(k))$-approximation algorithm for Densest k-SubHypergraph for any $\alpha(k) = 2^{o(k)}, \beta(k)$, and $f(k)$.

To formally state the SPCH and our result, let $\mathcal{G}(n, 1/2)$ be the distribution of an Erdős-Renyi random graph, and $\mathcal{G}(n, 1/2, n^\delta)$ be the distribution of a graph where a n^δ-size random clique is planted in an Erdős-Renyi random graph.

Hypothesis 7.3 (Strongish Planted Clique Hypothesis)**.** *There exists a constant $\delta \in (0, 1/2)$ such that no $n^{o(\log n)}$-time algorithm $\mathcal{A}$ can satisfy both of the following.*

- *(Completeness)* $\Pr_{G\sim\mathcal{G}(n,1/2,n^\delta)}[\mathcal{A}(G) = 1] \geq 2/3$.
- *(Soundness)* $\Pr_{G\sim\mathcal{G}(n,1/2)}[\mathcal{A}(G) = 1] \leq 1/3$.

Theorem 7.4. *Assuming the SPCH, for any function $f(k), \beta(k)$, and $\alpha(k) = 2^{o(k)}$, there is no $f(k) \cdot \mathrm{poly}(n)$-time $(\alpha(k), \beta(k))$-approximation algorithm for* Densest k-SubHypergraph *when $s = 2^k$.*

Since $k \cdot \alpha(k) = o(s)$, together with Lemma 7.2, it rules out $(\alpha'(k), \beta'(k))$-approximation for Small Set Vertex Expansion for any $\alpha'(k) = 2^{o(k)}$ and $\beta'(k)$ as well, finishing the proof of Theorem 1.5.

Proof. Given a value of k (as a parameter growing arbitrarily slower than n) and δ, we give a reduction from SPCH to Densest k-SubHypergraph. Given an instance $G = (V_G, E_G)$ of SPCH with n vertices, our reduction produces a hypergraph $H = (V_H, E_H)$. Our reduction is parameterized by ℓ, r, N, where $r = r(k) = o(k)$ and $\ell = o(\log n/r)$ will be determined later (so ℓ will be greater than any function of k), and $N = 100kn^{(1-\delta)\ell}$. The hypergraph H will be r-uniform. The reduction is as follows.

- $|V_H| = N$. Each vertex $s \in V_H$ corresponds to a random subset of V_G where each $v \in V_G$ is independently included with probability ℓ/n.
- A r-set $\{s_1, \ldots, s_r\} \subseteq V_H$ becomes a hyperedge if $\cup_{i\in[r]} s_i$ forms a clique in G.
- We will let $N = 100kn^{(1-\delta)\ell}$.
- Reduction time: at most $N^r = n^{O(\ell r)}$.

For completeness, if G contains a clique C of size n^δ, one vertex s_i of H, is a subset of C with probability $n^{(\delta-1)\ell}$. The expected number of such vertices is $100k$, so with a probability at least 0.9, the number of such vertices is at least k. In H, these k vertices induce $\binom{k}{r}$ hyperedges by construction.

For soundness, assume that there exists $S \subseteq V_H$ such that $t := |S| \leq \beta k$ and S induces at least $(1/\alpha)(|S|/k)\binom{k}{r}$ hyperedges, where $\alpha = \alpha(k)$ and $\beta = \beta(k)$ are the functions specified in the theorem. We will show that such S cannot exist with high probability.

Consider a graph G' whose vertex set is V_H and (s, s') is an edge if they belong to a same hyperedge of H. We want to say that each vertex in $s \in S$ has a large degree in $G'[S]$. (This is an analogue of Observation 6 of [46].)

Claim 7.5. *There exists $S' \subseteq S$ such that the degree of every $s \in S'$ in $G'[S']$ is at least $\frac{k}{2(\alpha k)^{1/r}}$.*

Proof. Initially, the *hypergraph density* of $H[S]$, which is simply the number of hyperedges divided by the number of vertices, is at least $(1/\alpha)(|S|/k)\binom{k}{r}/|S| \geq \binom{k}{r}/(\alpha k)$. Greedily, if removing $s \in S$ (and all its incident hyperedges) does not decrease the hypergraph density, do it until there is no such vertex.

Now we want to show that every $s \in S$ has a large degree. Consider $s \in S$ whose degree in $G'[S]$ is d. Then, the number of hyperedges of $H[S]$ containing s is at most $\binom{d}{r-1} \leq \binom{d}{r}$. (Note

that a hyperedge induces a r-clique in G' by construction.) Since removing s from S will decrease the hypergraph density we have,

$$\begin{aligned}
&\frac{\binom{k}{r}}{\alpha k} < \binom{d}{r} \\
\Rightarrow\ &\frac{(k/2)^r}{\alpha k} < d^r \qquad \text{(assuming } r \le k/10 \text{ that will be ensured later)} \\
\Rightarrow\ &d > \frac{k}{2(\alpha k)^{1/r}},
\end{aligned}$$

which proves the claim. □

For notational simplicity, redefine S to be the set satisfying $|S| \le \beta k$ and $\deg_{G'[S]}(s) \ge \frac{k}{2(\alpha k)^{1/r}}$ for every $s \in S$. Now, we show that in the graph G, the vertices in $(\cup_{s\in S} s)$ must have many edges. In order to show this, we use the following lemma [46] directly.

Lemma 7.6 (Lemma 7 of [46])**.** *If $N \le 1000\ell n^{(1-\delta)\ell}$ and $20 \le \ell$, for any $M \subseteq V_H$ with $|M| \le n^{0.99\delta}/\ell$, we have $|\cup_{s\in M} s| \ge 0.01\delta|M|\ell$.*

(Note that this is only over the randomness of selecting V_H and not the randomness of G.)

Let $T = \cup_{s\in S} s \subseteq V_G$, and consider the set of edges that should appear in $G[T]$. For each $v \in T$, fix an arbitrary $s \in S$ that contains v, and let $s_1, \ldots, s_d$ be the neighbors of s in $G'[S]$ (so that $d \ge k/(2(\alpha k)^{1/r})$). By Lemma 7.6, $|\cup_{i\in[d]} s_i| \ge 0.01\delta\ell k/(2(\alpha k)^{1/r})$, which implies that the degree of v in $G[T]$ is at least $0.01\delta\ell k/(2(\alpha k)^{1/r})$. Therefore, we can conclude that the total number of edges that must appear in $G[T]$ is at least $0.01|T|\delta\ell k/(4(\alpha k)^{1/r})$. The probability of all these edges appearing is $2^{-0.01|T|\delta\ell k/(4(\alpha k)^{1/r})} \le 2^{-0.0001\delta^2|S|\ell^2 k/(4(\alpha k)^{1/r})}$, again using Lemma 7.6 for T.

We union bound over all choices of S ($N^{|S|}$ choices for a given size) and all choices for $G'[S]$ ($2^{|S|^2} \le N^{|S|}$ choices since N is larger than any function of $|S|$.

$$\begin{aligned}
&\sum_{p=1}^{\beta k} N^{2p} \cdot 2^{-0.0001\delta^2 p\ell^2 k/(4(\alpha k)^{1/r})} \\
\le &\sum_{p=1}^{\beta k} \left(N^2 \cdot 2^{-0.0001\delta^2\ell^2 k/(4(\alpha k)^{1/r})}\right)^p \\
= &\sum_{p=1}^{\beta k} \left(2^{2(1-\delta)\ell\log(100kn) - 0.0001\delta^2\ell^2 k/(4(\alpha k)^{1/r})}\right)^p \\
= &\sum_{p=1}^{\beta k} \left(2^{2(1-\delta)\log(100kn) - 0.0001\delta^2\ell k/(4(\alpha k)^{1/r})}\right)^{\ell p}.
\end{aligned}$$

We set the parameters ℓ and r depending on $k, \alpha, \beta, f, \delta$. As long as $\ell k/(4(\alpha k)^{1/r}) > 1000000\log n/\delta^2$, the above probability becomes at most 0.01. Choose $\ell = (1000000/\delta^2)\log n/(r \cdot g(k))$, where g is a (slowly) growing function to be determined shortly, which implies that

$$\ell k/(4(\alpha k)^{1/r}) > 1000000\log n/\delta^2 \Leftrightarrow k/(4(\alpha k)^{1/r}) > rg(k) \Leftrightarrow \alpha k < (k/(4rg(k))^r.$$

Letting $r = k/(g(k))$ will ensure that as long as $\alpha k < 2^{k/(g(k))}$, the desired set S does not arise from $G(n, 1/2)$ with probability at least 0.99. Since $\alpha = 2^{o(k)}$, one can always choose a growing function g satisfying this.

Therefore, an (α, β)-approximation $\mathcal{A}$ for Densest k-SubHypergraph implies an algorithm that correctly decides whether a given graph is a random graph or a planted graph with probability at least 0.9. In terms of running time, since H has size at most $O(N^r)$, if $\mathcal{A}$ runs in time $f(k) \cdot N^{c \cdot r}$ for some absolute constant c, then its running time in terms of planted clique is

$$f(k)n^{(1-\delta)\ell r} \leq f(k)n^{1000000((1-\delta)/\delta^2)\log n/g(k)},$$

so for large enough k, the existence of $\mathcal{A}$ refutes the SPCH. □

7.2 Algorithm

For our next result, we prove Theorem 1.6 which shows that we can match this lower bound for Small Set Vertex Expansion using the treewidth reduction technique of Marx et al. [47], under a mild assumption. Previously, this technique was used by van Bevern et al. [64] to show the fixed-parameter tractability of vertex bisection when the number of connected components in the optimal bisection is a constant.

For our result, we will need the following theorem from [47].

Theorem 7.7 (Treewidth reduction, see Lemma 2.11 from [47])**.** *Given a graph G and two vertices s and t, one can in time $\mathcal{O}(f(k)(n+m))$ find a set of vertices U, such that the graph H obtained from G by contracting each connected component of $V(G) \setminus U$ has treewidth at most $2^{\mathcal{O}(k^2)}$. Further, U contains every inclusion-wise minimal s-t separator of size at most k.*

Proof of Theorem 1.6. There are two cases. For the first, assume that every component in $G \setminus C$ has size at most $\frac{3n}{4}$. In this case, C is a balanced separator, and we can simply find a balanced separator of size k using directly the result of [22] in time $2^{\mathcal{O}(k)}n^{\mathcal{O}(1)}$.

For the second case, when at least one component in $G \setminus C$ has size at least $\frac{3n}{4}$ - notice that without loss of generality, we can assume that R is connected, for otherwise we can move all except the largest component in R to L, and obtain a cut which is sparser, contradicting the maximality of (L, C, R).

Group the connected components $C_1, C_2 .... C_\ell$ of $G[L]$ by their neighbourhood in C. Observe that the number of groups is at most 2^k. It follows that there exists at least one group of components, say $\mathcal{A}$ so that $|\bigcup_{A \in \mathcal{A}} A| \geq \frac{|L|}{2^k}$ and each $A \in \mathcal{A}$ has the same neighbourhood in C. Consider an arbitrary component $A \in \mathcal{A}$. Let s be a vertex of A and t be a vertex of R. Consider the inclusion-wise minimal separator $T \subseteq C$ that separates s from t. Clearly, T must separate A from R. Since every component in $\mathcal{A}$ has the same neighbourhood in C, it follows that T in fact separates each component of $\mathcal{A}$ from R.

We now proceed as follows. First, we guess the vertices $s \in A$ and $t \in R$. Next, we invoke Theorem 7.7 to obtain the set U and the graph H with treewidth at most $2^{\mathcal{O}(k^2)}$. We assign a weight to each vertex of H: every vertex of U has weight 1, and every vertex corresponding to a connected component of $G \setminus U$ has weight equal to the number of vertices in that component. It is now clear that the problem reduces to finding a set with weight at least $\frac{|L|}{2^k}$ and at most $|L|$, that is separated from the rest of the graph by a separator of weight k in H. Using Theorem 4 from [64], this can be done in time $2^{\mathcal{O}(\mathbf{tw}(H))}\text{poly}(n)$, completing the proof. □

Note that Theorem 1.6 also gives a 2^k-approximation in time $2^{2^{\mathcal{O}(k^2)}}\text{poly}(n)$ for Small Set Vertex Expansion, when we are promised that the optimal cut is maximal. We remark that maximality is a natural assumption when we want to find the sparsest cut in the graph.

8 Applications

8.1 Multicut mimicking networks

In this section, we show an improvement in the size of the best multicut mimicking network that can be computed in polynomial time. This follows directly from the reduction in [66] to Small Set Expansion. We start by defining multicut mimicking networks.

Definition 8.1. *Given a graph G and a set of terminals T, let G' be another graph with $T \subseteq V(G')$. We say that G' a multicut mimicking network for G if for every partition $\mathcal{T} = T_1 \cup T_2 \cup \ldots T_s, s \geq 1$ of T, the size of a minimum multiway cut for $\mathcal{T}$ is the same in G and G'.*

In other words, a multicut mimicking network preserves the size of the minimum multiway cut for every partition of the terminal set (We recall that a multiway cut for a terminal set is a cut that separates all terminals from each other). Let us denote the total number of edges incident on vertices of T by $cap_G(T)$, and let $cap_G(T) = k$.

Theorem 8.2. *[66]] Let A be an approximation algorithm for* Small Set Expansion *with a bi-criteria approximation ratio of $(\alpha(n,k), \mathcal{O}(1))$, where n is the number of vertices and k is the number of edges cut in the optimal cut. Let (G,T) be a terminal network with $cap_G(T) = k$. Then there is a multicut mimicking network for G of size $k^{\mathcal{O}(\alpha(n,k)\log k)}$ and it can be found in randomized polynomial time.*

Since at the time no polylog(k)-approximation for Small Set Expansion was known, Theorem 8.2 by itself could only be used to show the existence of a multicut mimicking network. [66] however used the approximation algorithm for Small Set Expansion by Bansal et al. [7] who gave an approximation of $\mathcal{O}(\sqrt{\log n \log \frac{n}{s}})$, where s is the bound on the set size and obtained a multicut mimicking network of size $k^{\mathcal{O}(\log^3 k)}$ in polynomial-time using a more careful analysis.

From our results in Section 5, we obtain an algorithm with $\alpha(n,k) = \mathcal{O}(\log k)$. Clearly, Theorem 1.10 follows from Theorem 8.2 and the fact that $\alpha(n,k) = \mathcal{O}(\log k)$. This improves the result to $k^{\mathcal{O}(\log^2 k)}$.

This improves the best known kernels for many problems which are a consequence of multicut mimicking networks as considered in [66]. In particular, Multiway Cut parameterized by the cut-size and Multicut parameterized by the number of pairs and the cut-size together, admit $\mathcal{O}(k^{\mathcal{O}(\log^2 k)})$ size kernels in polynomial time.

8.2 Min-max graph partitioning using weighted sample sets

In min-max graph partitioning, we are given a graph G and an integer r. The goal is to partition the graph into r components of size $\frac{n}{r}$ each such that the maximum cut-size among the r components is minimized. Bansal et. al [7] gave an $\mathcal{O}(\sqrt{\log n \log r})$-approximation, and [29] obtained an $\mathcal{O}(\log \frac{n}{r})$-approximation using their approximation for Small Set Expansion. We show that we can obtain an $\mathcal{O}(\log opt)$-approximation for min-max graph partitioning, where opt is the min-max objective. This follows from an $\mathcal{O}(\log opt)$-approximation for Weighted-ρ-unbalanced cut. [7] shows that an α-approximation for Weighted-ρ-unbalanced cut essentially implies an α-approximation for min-max graph partitioning, so henceforth we will restrict our attention to Weighted-ρ-unbalanced cut.

Definition 8.3 (Weighted-ρ-unbalanced cut [7])**.** *Given a graph G, a weight function $y : V(G) \to \mathbb{Q}^+$, parameters $\tau, \rho \in (0,1)$, find a set S with $|S| \leq \rho n$ such that $y(S) \geq \tau y(V(G))$ such that $\delta_G(S)$ is minimum possible, or report that no such set exists.*

Let us denote the minimum possible $\delta_G(S)$ for some set S satisfying the above by *opt*. We now define an (α, β, γ)-approximation for this problem as in [7].

Definition 8.4 (Approximate Weighted-ρ-unbalanced cut)**.** *Given (G, y, τ, ρ), suppose that there exists a set S with $|S| \leq \rho n$ such that $y(S) \geq \tau y(V(G))$ and $\delta_G(S) = opt$. Find a set S' with $|S'| \leq \beta\rho n$ such that $y(S') \geq \frac{\tau y(V(G))}{\gamma}$ such that $|\delta_G(S')| \leq \alpha\, opt$.*

Theorem 8.5. *There is a $(\alpha = \mathcal{O}(\log opt), \beta = \mathcal{O}(1), \gamma = \mathcal{O}(1))$-approximation for weighted-ρ-unbalanced cut in polynomial time.*

Before we prove this theorem, we will need a generalization of sample sets as defined in Definition 4.1. Given a graph G and a measure μ on the vertices, we define the *μ-sparsity* of a set W as $\phi_\mu(S) = \frac{|\delta_G(S)|}{\mu(S)}$. Given a weight function $w : V(G) \to \mathbb{Q}^+$, we write $w(X) = \sum_{v \in X} w(v)$ for any subset $X \subseteq V(G)$.

Definition 8.6. *Given a graph $G = (V(G), E(G))$ with a measure μ on the vertices, parameters ϵ, ϕ', an (ϵ, ϕ') (edge) sample set for G is a set of terminals $T \subseteq V(G)$ with a weight function α which assigns a weight $\alpha_t \geq 1$ to each terminal t. Suppose that $\mu(V(G)) = K$. Consider a set of vertices $W \subseteq V(G)$ which satisfies either (a) $\phi_\mu(W) \leq \phi'$ or (b) $\phi_\alpha(W) \leq \frac{K}{10\alpha(T)}\phi'$. Then we must have $|\frac{K}{\alpha(T)}\alpha(W \cap T) - \mu(W)| \leq \epsilon\mu(W)$.*

Lemma 8.7. *There is an (ϵ, ϕ') edge sample set for every graph G of total weight $\alpha(V(G)) = \alpha(T) = \Theta\left(\mu(V(G)\frac{\phi'}{\epsilon^2}\right)$. Further, such a set can be computed in polynomial time.*

Proof. Our proof is similar to the construction of the determinstic sample set in [22] for edge separators. We will use the notion of Steiner-t-decomoposition introduced in [22].

Lemma 8.8 (Follows from Lemma 5.2, [22] and Claim 3.3 [24])**.** *Given any graph G, a measure μ' on the vertices, and a value $1 \leq t \leq n$. Suppose every vertex $v \in V(G)$ satisfies $\mu'(v) \leq t$. Then one can compute in polynomial time a partition $V_1, V_2 \ldots V_\ell$ of $V(G)$ and a partition of the edge set $E_1, E_2 \ldots E_\ell, E'$ such that*

1. *$G[E_i]$ is connected for each $i \geq 1$*
2. *$V(G[E_i]) = V_i \cup \{u\}$ for some vertex u.*
3. *$\mu'(V_i) \leq 2t$ for all $i \in [\ell]$.*
4. *$\mu'(V_i) \geq t$ for all $i > 1$.*

We will refer to each vertex set V_i as a *bag* in the decomposition.

We now compute a sample set as follows. Fix $t = \frac{\epsilon}{100\phi'}$. First, for any vertex v with $\mu(v) > t$, which we shall henceforth call a *high vertex*, pick v into the sample set T, and assign it a weight $\alpha_v = \lfloor \frac{\mu(v)}{t\epsilon} \rfloor$. Let V_h denote the set of high vertices. Define a new measure μ' as follows.

$$\mu'(v) = \left\{ \begin{array}{ll} \mu(v), & \text{if } v \notin V_h \\ 0 & \text{otherwise.} \end{array} \right\}$$

Compute a Steiner-t-decomposition for μ' with $t = \frac{\epsilon}{100\phi'}$. Recall that $\mu(V(G)) = K$, and let $\mu'(V(G)) = K'$. We now add to the sample set T as follows: arbitrarily pick 1 vertex from the bag V_i and assign it a weight of $\lfloor \frac{\mu'(V_i)}{t\epsilon} \rfloor$ for each $i > 1$.

Lemma 8.9. *The set T is a $(4\epsilon, \phi')$ sample set for all $\epsilon \in (0, \frac{1}{100})$.*

Proof. We first show that every bag is represented proportional to its measure in the terminal set. Observe that from every bag V_i, we pick one vertex with weight $\frac{\mu'(V_i)}{t\epsilon} \geq \lfloor \frac{\mu'(V_i)}{t\epsilon} \rfloor \geq \frac{\mu'(V_i)}{t\epsilon} - 1$. Since $t \leq \mu'(V_i)$ for all $i > 1$, this means that $\alpha(V_i) \in [\frac{\mu'(V_i)}{t\epsilon} - \epsilon\frac{\mu'(V_i)}{t\epsilon}, \frac{\mu'(V_i)}{t\epsilon}]$ for all $i > 1$. Similarly, we also obtain $\alpha(v_h) \in [\frac{\mu(v_h)}{t\epsilon} - \epsilon\frac{\mu(v_h)}{t\epsilon}, \frac{\mu(v_h)}{t\epsilon}]$ for every vertex $v_h \in V_h$.

Adding and noting that $\mu'(V_1) \leq 2t$ we obtain $\alpha(T \setminus V_h) \in [\frac{(K'-2t)}{t\epsilon}(1-\epsilon), \frac{K'}{t\epsilon}]$. Similarly, we get $\alpha(V_h) \in [\frac{(K-K')}{t\epsilon}(1-\epsilon), \frac{K-K'}{t\epsilon}]$. Together, we obtain $\alpha(V(G)) = \alpha(T) \in [\frac{K-2t}{t\epsilon}(1-\epsilon), \frac{K}{t\epsilon}]$. We know that $t = \frac{\epsilon}{100\phi'} \leq \frac{\epsilon}{4}K$, which in turn gives $K - 2t \geq (1 - \frac{\epsilon}{2})K$, and hence we have $\alpha(T) \in [(1-\frac{\epsilon}{2})(1-\epsilon)\frac{K}{t\epsilon}, \frac{K}{t\epsilon}]$ or equivalently, $\alpha(T) \in [(1-2\epsilon)\frac{K}{t\epsilon}, \frac{K}{t\epsilon}]$. This means that $\frac{K}{\alpha(T)}\alpha(V_i) \in [\mu'(V_i)(1-\epsilon), \frac{\mu'(V_i)}{1-2\epsilon}]$ and hence $\frac{K}{\alpha(T)}\alpha(V_i) \in [\mu'(V_i)(1-\epsilon), \mu'(V_i)(1+3\epsilon)]$ for small enough ϵ, for all $i > 1$. Similarly we also obtain $\frac{K}{\alpha(T)}\alpha(v_h) \in [\mu(V_h)(1-\epsilon), \mu(V_h)(1+3\epsilon)]$ for each $v_h \in V_h$.

First, consider the case where we are given a set W with $\phi_\mu(W) \leq \phi'$. Observe that the number of edges cut, $\delta_G(W)$ is at most $\phi'\mu(W)$. Mark a bag V_i of the decomposition as "bad" if there is a cut edge which has both endpoints in E_i. Additionally, we always mark the bag V_0 as bad. Clearly, the number of bad bags is at most $\phi'\mu(W) + 1$. The total μ-measure in these bags excluding the vertices in V_h is at most $2t\phi'\mu(W) + t \leq \frac{\epsilon\mu(W)}{4}$, since $t = \frac{\epsilon}{100\phi'} \leq \frac{\epsilon\delta_G(W)}{100\phi'} \leq \frac{\epsilon\mu(W)}{100}$. Thus at least $(1-\frac{\epsilon}{4})\mu(W)$ measure of W is either in good bags or the vertices V_h. Observe that by the definition of a good bag, if W intersects a good bag V_i, it must include the whole bag V_i, since there is a way to reach every vertex of the bag using the edges E_i, and none of the edges in E_i are cut edges.

Let G' be the set of vertices of the good bags and the vertices V_h contained in W. From the preceeding argument, we know that $\mu(G') \geq (1-\frac{\epsilon}{4})\mu(W)$. Recall that every good bag V_i satisfies $\frac{K}{\alpha(T)}\alpha(V_i) \in [\mu'(V_i)(1-\epsilon), \mu'(V_i)(1+3\epsilon)]$, and every vertex $v_h \in V(h)$ satisfies $\frac{K}{\alpha(T)}\alpha(v_h) \in [\mu(V_h)(1-\epsilon), \mu(V_h)(1+3\epsilon)]$. It follows that W must satisfy $\frac{K}{\alpha(T)}\alpha(W) \geq (1-\epsilon)(1-\frac{\epsilon}{4})\mu(W) \geq (1-2\epsilon)\mu(W)$. On the other hand, we have $\frac{K}{\alpha(T)}\alpha(G') \leq \mu'(V_i)(1+3\epsilon)$ and $\frac{K}{\alpha(T)}\alpha(v_h) \leq \mu(V_h)(1+3\epsilon)$. Let B be the union of vertices in all the bad bags. Observe that by construction we must have $\alpha(B) \leq \frac{\mu(B)}{t\epsilon}$. Noting that $\alpha(T) \geq (1-2\epsilon)\frac{K}{t\epsilon}$, this gives $\frac{K}{\alpha(T)}\alpha(B) \leq \frac{\mu(B)}{1-2\epsilon} \leq \frac{\mu(W)\epsilon}{4(1-2\epsilon)} \leq \frac{\mu(W)\epsilon}{2}$ for small enough ϵ. Together, we obtain $\frac{K}{\alpha(T)}\alpha(W) \leq \frac{K}{\alpha(T)}\alpha(G') + \frac{K}{\alpha(T)}\alpha(B) + \frac{K}{\alpha(T)}\alpha(V_h) \leq \mu(W)(1+3\epsilon+\frac{\epsilon}{2}) \leq \mu(W)(1+4\epsilon)$.

Now consider the case when we are given a set W with $\phi_\alpha(W) \leq \frac{\epsilon^2}{200}$ (Note that $\frac{\epsilon^2}{200} \geq \frac{K}{10\alpha(T)}\phi'$). Again, we have $\delta_G(W) \leq \phi_T(W)\alpha(W) \leq \frac{\epsilon^2}{200}\alpha(W)$, and this upper bounds the number of bad bags. Define G' and B as before. Recall that each bag V_i satisfies $\alpha(V_i) \leq \frac{\mu'(V_i)}{t\epsilon} \leq \frac{2}{\epsilon}$. This means that the total weight of terminals in bad bags $\alpha(B) \leq \frac{2}{\epsilon} \cdot \frac{\epsilon^2}{200}\alpha(W) \leq \epsilon\alpha(W)$. Thus we must have $\alpha(W \cap B) \leq \epsilon\alpha(W)$. This implies that $\alpha(G') \geq (1-\epsilon)\alpha(W)$. Since G' is a collection of bags (and high vertices), as in the previous analysis, we must have $\frac{K}{\alpha(T)}\alpha(G') \in [\mu(G')(1-\epsilon), \mu(G')(1+3\epsilon)]$. This gives

$$\mu(W) \geq \mu(G') \geq \frac{K}{\alpha(T)}\frac{\alpha(G')}{1+3\epsilon} \geq \frac{K}{\alpha(T)}(1-\epsilon)\frac{\alpha(W)}{1+3\epsilon}$$

It follows that

$$\mu(W) \geq \frac{K}{\alpha(T)}\alpha(W)(1-4\epsilon) \geq \frac{K}{2\alpha(T)}\alpha(W) \geq \frac{K}{\alpha(T)}\frac{100}{\epsilon^2}\delta_G(W).$$

Finally, note that $\frac{K}{\alpha(T)} \geq t\epsilon$, so that we get $\mu(W) \geq \frac{1}{\phi'}\delta_G(W)$, and hence W is ϕ'-sparse. Since we already proved the condition for any ϕ'-sparse set, this concludes the proof of the lemma. □

As shown already, we have $\alpha(T) \leq \frac{K}{t\epsilon} = \mathcal{O}(\frac{K\phi'}{\epsilon^2}) = \mathcal{O}(\frac{\mu(V(G))\phi'}{\epsilon^2})$. □

Proof of Theorem 8.5. We will again combine our notion of sample sets with the LP-rounding framework. First, suppose that we are given a graph G together with a set of terminals T and integral weights α on the terminals, with $\alpha(v) \geq 1$ whenever $\alpha(v) \neq 0$ for any vertex v. Now further suppose that there exists a set $S \subseteq V(G)$ with $|S| = s$, $\alpha(S) = \ell$ and $|\delta_G(S)| = opt$. Then we design an algorithm that finds a set S' with $|S'| \leq 10s$, $\alpha(S') \geq \frac{\ell}{10}$ and $|\delta_G(S')| \leq \mathcal{O}(opt \log \ell)$. The proof combines the ideas behind Theorem 13 of [29] and Theorem 2.1 of [7]. We postpone the proof to the appendix (see Theorem A.1).

Now we proceed as follows. Let $(S, V(G) \setminus S)$ denote the optimal partition where S has size $s \leq \rho n$. First, we guess the size s by trying all choices from 1 to ρn (In fact is sufficient to guess s up to factor 2). Similarly, we also guess the value $y(S)$, suppose it is equal to ℓ. It follows that S is $\phi = \frac{opt}{\ell}$-sparse with respect to the measure y. Choose $\epsilon = \frac{1}{100}$ and construct an $(\epsilon, \phi' = C\phi \log opt)$ sample set T with total weight $\alpha(T) = \Theta(y(V(G))\phi \log opt)$ where C is a large enough constant that will be chosen later. The sample set guarantee implies that $\alpha(S) = \Theta(C opt \log opt)$. Using the result from Theorem A.1, it follows that we obtain a set S' with $|S'| \leq 10s'$, $\alpha(S') \geq \frac{\alpha(S)}{10}$ with $|\delta_G(S')| \leq \mathcal{O}(\log opt) opt$. This set has terminal sparsity $\phi_\alpha(S) = \mathcal{O}(\frac{1}{C})$. Now we choose C large enough so that $\phi_T(S') \leq \frac{\epsilon^2}{200} \leq \frac{K}{10\alpha(T)}\phi'$.

Then S' must satisfy the sample condition. Using the guarantee of sample sets, since $\alpha(S') \geq \frac{C}{10} opt \log opt$ it must be the case that $y(S') \geq \Omega(\ell)$. This completes the proof. □

9 Discussion and Open Problems

Fast $O(\text{polylog}k)$-approximate Small Set Expansion. We show $O(\text{polylog}k)$-approximation algorithms for Small Set Expansion that run in polynomial time based on LPs in Section 5. Via a new cut-matching game, we speed up the running time to almost linear time to solve the easier problem, Sparsest Cut, where we do not require the output set to be small (see Section 6). A natural question is whether there is an almost-linear time $O(\text{polylog}k)$-approximation algorithm for Small Set Expansion.

$O(\text{polylog}k)$-approximate Vertex Sparsest Cut. Can we remove the $\text{poly}(\log \log n\phi)$ term in Theorems 1.7 and 1.8 for approximating Vertex Sparsest Cut and obtain $O(\text{polylog}k)$ approximation algorithms? We have shown a fundamental barrier of our current approach based on sample sets in Section 4.3.

References

[1] Akanksha Agrawal, Daniel Lokshtanov, Pranabendu Misra, Saket Saurabh, and Meirav Zehavi. Feedback vertex set inspired kernel for chordal vertex deletion. *ACM Transactions on Algorithms (TALG)*, 15(1):1–28, 2018. 1

[2] Reid Andersen and Kevin J Lang. An algorithm for improving graph partitions. In *SODA*, volume 8, pages 651–660, 2008. 9

[3] Alexandr Andoni, Jiecao Chen, Robert Krauthgamer, Bo Qin, David P Woodruff, and Qin Zhang. On sketching quadratic forms. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 311–319, 2016. 1

[4] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. *Journal of the ACM (JACM)*, 62(5):1–25, 2015. 1

[5] Sanjeev Arora, James R Lee, and Assaf Naor. Euclidean distortion and the sparsest cut. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 553–562, 2005. 1

[6] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM (JACM)*, 56(2):1–37, 2009. 1

[7] Nikhil Bansal, Uriel Feige, Robert Krauthgamer, Konstantin Makarychev, Viswanath Nagarajan, Joseph Seffi, and Roy Schwartz. Min-max graph partitioning and small set expansion. *SIAM Journal on Computing*, 43(2):872–904, 2014. 1, 5, 6, 8, 18, 34, 35, 37, 42, 43, 46, 47, 49

[8] Boaz Barak, Fernando GSL Brandao, Aram W Harrow, Jonathan Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 307–326, 2012. 1

[9] Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities: an $\mathcal{O}(n^{\frac{1}{4}})$ approximation for densest k-subgraph. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 201–210, 2010. 3

[10] Vijay Bhattiprolu, Euiwoong Lee, and Assaf Naor. A framework for quadratic form maximization over convex sets through nonconvex relaxations. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 870–881, 2021. 1

[11] Gruia Calinescu, Howard Karloff, and Yuval Rabani. Approximation algorithms for the 0-extension problem. *SIAM Journal on Computing*, 34(2):358–372, 2005. 6

[12] Marcelo Campos, Simon Griffiths, Robert Morris, and Julian Sahasrabudhe. An exponential improvement for diagonal ramsey. *arXiv preprint arXiv:2303.09521*, 2023. 16

[13] Li Chen, Rasmus Kyng, Yang P Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 612–623. IEEE, 2022. 5, 22, 27

[14] Eden Chlamtáč, Michael Dinitz, and Yury Makarychev. Minimizing the union: Tight approximations for small set bipartite vertex expansion. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 881–899. SIAM, 2017. 3

[15] Timothy Chu, Yu Gao, Richard Peng, Sushant Sachdeva, Saurabh Sawlani, and Junxing Wang. Graph sparsification, spectral sketches, and faster resistance computation via short cycle decompositions. *SIAM Journal on Computing*, 52(6):FOCS18–85–FOCS18–157, 2023. 1

[16] Julia Chuzhoy, Yury Makarychev, Aravindan Vijayaraghavan, and Yuan Zhou. Approximation algorithms and hardness of the k-route cut problem. *ACM Transactions on Algorithms (TALG)*, 12(1):1–40, 2015. 3

[17] Michael B Cohen, Jonathan Kelner, John Peebles, Richard Peng, Anup B Rao, Aaron Sidford, and Adrian Vladu. Almost-linear-time algorithms for markov chains and new spectral primitives for directed graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 410–419, 2017. 1

[18] Marek Cygan, Paweł Komosa, Daniel Lokshtanov, Marcin Pilipczuk, Michał Pilipczuk, Saket Saurabh, and Magnus Wahlström. Randomized contractions meet lean decompositions. *ACM Transactions on Algorithms (TALG)*, 17(1):1–30, 2020. 1, 6

[19] Guy Even, Joseph Naor, Satish Rao, and Baruch Schieber. Fast approximate graph partitioning algorithms. *SIAM Journal on Computing*, 28(6):2187–2214, 1999. 6

[20] Guy Even, B Schieber, M Sudan, et al. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998. 6

[21] Uriel Feige, MohammadTaghi Hajiaghayi, and James R Lee. Improved approximation algorithms for minimum-weight vertex separators. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 563–572, 2005. 1, 2, 4, 6

[22] Uriel Feige and Mohammad Mahdian. Finding small balanced separators. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 375–384, 2006. 5, 7, 11, 12, 14, 33, 35

[23] Fedor V Fomin, Daniel Lokshtanov, Neeldhara Misra, Geevarghese Philip, and Saket Saurabh. Hitting forbidden minors: Approximation and kernelization. *SIAM Journal on Discrete Mathematics*, 30(1):383–410, 2016. 1

[24] Fedor V Fomin, Daniel Lokshtanov, Saket Saurabh, Michał Pilipczuk, and Marcin Wrochna. Fully polynomial-time parameterized computations for graphs and matrices of low treewidth. *ACM Transactions on Algorithms (TALG)*, 14(3):1–45, 2018. 12, 35

[25] 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. 6

[26] Suprovat Ghoshal and Anand Louis. Approximation algorithms and hardness for strong unique games. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 414–433. SIAM, 2021. 1

[27] Mohammad Taghi Hajiaghayi and Kamal Jain. The prize-collecting generalized steiner tree problem via a new approach of primal-dual schema. In *SODA*, volume 6, pages 631–640, 2006. 3

[28] Sariel Har-Peled. Geometric approximation algorithms. *Lecture Notes for CS598, UIUC*, 2008. 14

[29] Mohammad Khairul Hasan and Kyung-Yong Chwa. Approximation algorithms for the weighted t-uniform sparsest cut and some other graph partitioning problems. *Journal of Computer and System Sciences*, 82(6):1044–1063, 2016. 5, 6, 7, 8, 17, 18, 34, 37

[30] Monika Henzinger, Satish Rao, and Di Wang. Local flow partitioning for faster edge connectivity. *SIAM Journal on Computing*, 49(1):1–36, 2020. 9

[31] Arun Jambulapati and Aaron Sidford. Efficient $O(\frac{n}{\epsilon})$ spectral sketches for the laplacian and its pseudoinverse. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2487–2503. SIAM, 2018. 1

[32] Bart MP Jansen and Marcin Pilipczuk. Approximation and kernelization for chordal vertex deletion. *SIAM Journal on Discrete Mathematics*, 32(3):2258–2301, 2018. 1

[33] Chris Jones, Aaron Potechin, Goutham Rajendran, and Jeff Xu. Sum-of-squares lower bounds for densest k-subgraph. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 84–95, 2023. 3

[34] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3):497–515, 2004. 1

[35] Jonathan A Kelner, Yin Tat Lee, Lorenzo Orecchia, and Aaron Sidford. An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 217–226. SIAM, 2014. 1

[36] Rohit Khandekar, Subhash Khot, Lorenzo Orecchia, and Nisheeth K Vishnoi. On a cut-matching game for the sparsest cut problem. *Univ. California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2007-177*, 6(7):12, 2007. 5, 8, 9, 19, 20, 23, 24, 25

[37] Rohit Khandekar, Satish Rao, and Umesh Vazirani. Graph partitioning using single commodity flows. *Journal of the ACM (JACM)*, 56(4):1–15, 2009. 1, 5, 8, 9, 26

[38] Subhash A Khot and Nisheeth K Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative-type metrics into $l1$. *Journal of the ACM (JACM)*, 62(1):1–39, 2015. 1

[39] Kevin Lang and Satish Rao. A flow-based method for improving the expansion or conductance of graph cuts. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 325–337. Springer, 2004. 9

[40] Euiwoong Lee. Partitioning a graph into small pieces with applications to path transversal. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1546–1558. SIAM, 2017. 6, 18

[41] Euiwoong Lee and Suprovat Ghoshal. A characterization of approximability for biased csps. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 989–997, 2022. 1

[42] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM (JACM)*, 46(6):787–832, 1999. 1

[43] Jason Li, Danupon Nanongkai, Debmalya Panigrahi, and Thatchaphol Saranurak. Near-linear time approximations for cut problems via fair cuts. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 240–275. SIAM, 2023. 22

[44] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–245, 1995. 1

[45] Anand Louis and Yury Makarychev. Approximation algorithms for hypergraph small set expansion and small set vertex expansion. *arXiv preprint arXiv:1404.4575*, 2014. 1, 6

[46] Pasin Manurangsi, Aviad Rubinstein, and Tselil Schramm. The strongish planted clique hypothesis and its consequences. *arXiv preprint arXiv:2011.05555*, 2020. 3, 30, 31, 32

[47] Dáaniel Marx, Barry O'sullivan, and Igor Razgon. Finding small separators in linear time via treewidth reduction. *ACM Transactions on Algorithms (TALG)*, 9(4):1–35, 2013. 3, 33

[48] Danupon Nanongkai and Thatchaphol Saranurak. Dynamic spanning forest with worst-case update time: adaptive, las vegas, and $\mathcal{O}(n^{\frac{1}{2}-\epsilon})$-time. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1122–1129, 2017. 1

[49] Danupon Nanongkai, Thatchaphol Saranurak, and Christian Wulff-Nilsen. Dynamic minimum spanning forest with subpolynomial worst-case update time. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 950–961. IEEE, 2017. 1

[50] Assaf Naor and Robert Young. Vertical perimeter versus horizontal perimeter. *Annals of Mathematics*, 188(1):171–279, 2018. 1

[51] Lorenzo Orecchia, Leonard J Schulman, Umesh V Vazirani, and Nisheeth K Vishnoi. On partitioning graphs via single commodity flows. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 461–470, 2008. 5

[52] Lorenzo Orecchia and Nisheeth K Vishnoi. Towards an sdp-based approach to spectral methods: A nearly-linear-time algorithm for graph partitioning and decomposition. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 532–545. SIAM, 2011. 9, 21

[53] Lorenzo Orecchia and Zeyuan Allen Zhu. Flow-based algorithms for local graph clustering. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1267–1286. SIAM, 2014. 9

[54] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 255–264, 2008. 1, 2

[55] Harald Räcke, Chintan Shah, and Hanjo Täubig. Computing cut-based hierarchical decompositions in almost linear time. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 227–238. SIAM, 2014. 2

[56] Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 755–764, 2010. 1

[57] Prasad Raghavendra, David Steurer, and Prasad Tetali. Approximations for the isoperimetric and spectral profile of graphs and related parameters. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 631–640, 2010. 1, 6

[58] Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *2012 IEEE 27th Conference on Computational Complexity*, pages 64–73. IEEE, 2012. 1

[59] Thatchaphol Saranurak and Di Wang. Expander decomposition and pruning: Faster, stronger, and simpler. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2616–2635. SIAM, 2019. 1, 9

[60] Norbert Sauer. On the density of families of sets. *Journal of Combinatorial Theory, Series A*, 13(1):145–147, 1972. 14

[61] Saharon Shelah. A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, 41(1):247–261, 1972. 14

[62] 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, 2004. 1

[63] Luca Trevisan. Approximation algorithms for unique games. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 197–205. IEEE, 2005. 1

[64] René Van Bevern, Andreas Emil Feldmann, Manuel Sorge, and Ondřej Suchỳ. On the parameterized complexity of computing balanced partitions in graphs. *Theory of Computing Systems*, 57:1–35, 2015. 33

[65] Magnus Wahlström. On quasipolynomial multicut-mimicking networks and kernelization of multiway cut problems. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020. 1, 2

[66] Magnus Wahlström. Quasipolynomial multicut-mimicking networks and kernels for multiway cut problems. *ACM Transactions on Algorithms (TALG)*, 18(2):1–19, 2022. 1, 4, 5, 34

[67] Christian Wulff-Nilsen. Fully-dynamic minimum spanning forest with improved worst-case update time. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1130–1143, 2017. 1

A Appendix

In this section, we extend an $\mathcal{O}(\log s)$-approximation for Small Set Expansion to the terminal version and vertex terminal version.

Theorem A.1. *Suppose we are given a graph G and a set of terminals $T \subseteq V(G)$, and integral weights x such that $x(V(G)) = K$, x is non-zero only on vertices of T, and $x(v) \geq 1$ for each $v \in T$. Suppose there exists a set with $|S| = s$, $x(S) = \ell$ and $|\delta_G(S)| = opt$. Then we can find a set Y satisfying $|Y| \leq 10s$, $3\ell \geq x(Y) \geq \frac{\ell}{10}$ with $|\delta_G(Y)| \leq \mathcal{O}(\log \ell)opt$.*

Proof. We start with the LP in [7] with some modifications. Let us fix $|T| = t$.

$$\min \sum_{e=\{u,v\}\in E(G)} d(u,v)$$

$$\sum_{u\in V(G)} \min\{d(u,v), y_v\} \geq (n-s)y_v \ \forall v \in V(G) \quad \ldots(1)$$

$$\sum_{u\in V(G)} x(u)\min\{d(u,v), y_v\} \geq (K-\ell)y_v \ \forall v \in V(G) \quad \ldots(2)$$

$$\sum_{v\in V(G)} x(v)y_v \geq \ell$$

$$\sum_{v\in V(G)} y_v \leq s$$

$$d(u,v) \leq d(u,w) + d(w,v) \ \forall u,v,w \in V(G)$$

$$d(u,v) = d(v,u) \ \forall u,v \in V(G)$$

$$d(u,v) \geq y_u - y_v \ \forall u,v \in V(G)$$

$$d(u,v), y_v \geq 0 \ \forall u,v \in V(G)$$

It is clear that the LP is a relaxation - to see this, given the integral solution S, set $y_v = 1$ for each $v \in S$, and 0 otherwise. Set $d(u,v) = 1$ if $u \in S$ and $v \notin S$ or $u \notin S$ and $v \in S$, and $d(u,v) = 0$ otherwise. The LP value is at most *opt*, where *opt* denotes the size of the optimal cut $|\delta_G(S)|$.

The rounding: Having obtained the optimal values $d(u,v)$ from the optimal LP solution, we now describe our rounding procedure. Our goal will again be to produce an LP-separator as introduced in [7], but with the improved approximation ratio $\mathcal{O}(\log \ell)$. Before describing our rounding scheme, we begin with a crucial observation that will be used multiple times.

Observation A.2. *For any vertex $v \in V(G)$ and $r < 1$, let $Ball_r(v) = \{u \in V(G) \mid d(u,v) \leq ry_v\}$. Then for every $v \in V(G)$, we must have $|Ball_r(v)| \leq \frac{s}{1-r}$ and $x(Ball_r(v)) \leq \frac{\ell}{1-r}$.*

Proof. Let $|Ball_r(v)| = z$. The LP spreading constraint (1) for vertex v implies that we must have $z \cdot ry_v + (n-z)\cdot y_v \geq (n-s)y_v$. This in turn means that $z \leq \frac{s}{1-r}$.

Similarly, let $x(Ball_r(v)) = z'$. The second spreading constraint (2) for vertex v implies that we must have $z' \cdot ry_v + (K-z')\cdot y_v \geq (K-\ell)y_v$ which gives $z' \leq \frac{\ell}{1-r}$. □

For a set $X \subseteq V(G)$, we define $f(X) = x(X) - \frac{\ell}{10s}|X| - \frac{\ell}{200LP\log\ell}|\delta_G(X)|$ where LP is the optimal LP value.

The algorithm runs for multiple iterations. Algorithm 4 describes the rounding scheme. We start with $Y = \emptyset$. In every iteration, as long as $|Y| \leq s$ and $x(Y) \leq \frac{\ell}{4}$, we proceed with the next iteration to find a cluster C that satisfies $f(C) > 0$. We then set $Y = Y \cup C$, and repeat.

Here we note that at an intermediate step, once the set Y is removed, the LP solution for the vertices $V(G) \setminus Y$ remains "almost-feasible" for the graph $G \setminus Y$. More precisely, every constraint is still satisfied, except potentially the constraint $\sum_{v\in V(G\setminus Y)} x(v) \geq \ell$. However since $x(Y) \leq \frac{\ell}{4}$, it must be the case that $\sum_{v\in V(G\setminus Y)} x(v) \geq \frac{3\ell}{4}$. Therefore we will use this weaker constraint in our analysis at an intermediate step.

Next, we describe the analysis of the rounding algorithm for each iteration, where we find a cluster C.

Algorithm 4 Rounding the LP

$Y \leftarrow \emptyset$
while $|Y| \leq s$ and $x(Y) \leq \frac{\ell}{4}$ **do**
  for $j = 1$ to $\mathcal{O}(n \log n)$ **do**
    Pick a threshold $\delta \in [0, 1]$ uniformly at random.
    Let $X = \{v \in T \mid y_v \in [\delta, 2\delta]\}$ and $\pi : [|X|] \to X$ be a random permutation of the elements of X. Let $U \leftarrow \emptyset$
    Pick $r \in [0.05, 0.1]$ uniformly at random.
    for $i = 1, 2 \ldots |X|$ **do**
      Let $C = \{v \in V(G) \setminus U \mid d(\pi(i), v) \leq r y_{\pi(i)}\}$
      $U \leftarrow U \cup C$
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{C\}$
    end for
    Assign to C one cluster of $\mathcal{S}$ with probability $\frac{1}{n}$, return empty set with probability $1 - \frac{|\mathcal{S}|}{n}$
    if $f(C) > 0$ **then**
      $Y = Y \cup C$
      break
    end if
  end for
end while

Every time we pick a vertex $\pi(i)$ and separate out a cluster C, we will call $\pi(i)$ the *center* of the cluster C. Also, we will say that two vertices u and v are separated while considering $\pi(i) \in T$ if exactly one of u, v is in the cluster with center $\pi(i)$, and neither u nor v is in any cluster whose center is $\pi(1), \pi(2) \ldots \pi(i-1)$.

Lemma A.3. *In each iteration of the outer for loop, the output set C satisfies the following properties.*

1. *For every vertex $v \in T$, we have $v \in C$ with probability at least $\frac{y_v}{2n}$. For any vertex $v \in V(G)$, $v \in C$ with probability at most $\frac{2y_v}{3n}$.*
2. *$|C| \leq 2s$ and $x(C) \leq 2\ell$.*
3. *The expected number of edges cut, $E[|\delta_G(C)|] \leq \frac{\mathcal{O}(\log \ell) LP}{n}$.*

Proof. Clearly, for any $v \in T$ $y_v \in [\delta, 2\delta]$ whenever $\delta \in [\frac{y_v}{2}, y_v]$ which happens with probability at least $\frac{y_v}{2}$. This means that with probability $\frac{y_v}{2}$, v is in some cluster $W \in \mathcal{S}$. But each cluster is picked with exactly probability $\frac{1}{n}$, and hence it must be the case that v is in C with probability at least $\frac{y_v}{2n}$. Now we show the other part. Suppose $v \in V(G)$ is in some cluster W and w is the center of the cluster W. We have $d(u, w) \leq 0.1 y_w$. Using the LP constraints, it is easy to see that $0.9 y_w \leq y_v \leq 1.1 y_w$. But $y_w \in [\delta, 2\delta]$, and hence $0.9\delta \leq y_v \leq 2.2\delta$, or equivalently we must have $\frac{y_v}{2.2} \leq \delta \leq \frac{y_v}{0.9}$. But this happens with probability at most $\frac{2y_v}{3}$. Again, since the cluster containing v is picked with probability exactly $\frac{1}{n}$, the probability that v is in the chosen cluster is at most $\frac{2y_v}{3n}$.

(2) directly follows from Observation A.2 with $r = 0.1$.

Finally, we prove (3). Consider an edge $e = (u, v)$. This edge can be cut only if u or v is in some cluster C of $\mathcal{S}$. Without loss of generality suppose that u is in some such cluster C (the other case is symmetric). Let w be the center of cluster C.

As noted above, we must have $\frac{y_u}{2.2} \leq \delta \leq \frac{y_u}{0.9}$. Thus, the only values of δ for which the edge $\{u,v\}$ can be an edge in $\delta_G(C)$ must satisfy $\frac{y_u}{2.2} \leq \delta \leq \frac{y_u}{0.9}$ or $\frac{y_v}{2.2} \leq \delta \leq \frac{y_v}{0.9}$.

Now, fix a value of δ such that either $\frac{y_u}{2.2} \leq \delta \leq \frac{y_u}{0.9}$ or $\frac{y_v}{2.2} \leq \delta \leq \frac{y_v}{0.9}$.

Let $X' \subseteq X$ be the set of vertices w of X that satisfy $d(w,u) \leq 0.1y_w$ or $d(w,v) \leq 0.1y_w$. Informally, these are the only vertices that can separate u or v from each other, thereby cutting the edge $\{u,v\}$. We show first that $|X'| \leq 10\ell$. Suppose to the contrary that $|X'| > 10\ell$. It is clear that at least $5\ell+1$ vertices $w \in X'$ that satisfy one of the two inequalities $d(w,u) \leq 0.1y_w$ or $d(w,v) \leq 0.1y_w$. Without loss of generality, we assume that $5\ell+1$ vertices $w \in X$ satisfy $d(w,u) \leq 0.1y_w$. Let X'' be the set of such vertices w. Fix some $w' \in X''$. By the triangle inequality, for every $w \in X''$ we have $y_w + 0.1y_w \geq y_u \geq y_w - 0.1y_w = 0.9y_w$. Thus $d(w,u) \leq \frac{y_u}{9}$. Also $d(w',u) \leq \frac{y_u}{9}$. By the triangle inequality on d, we must have $d(w',w) \leq d(w,u) + d(w,w') \leq \frac{2y_u}{9} \leq \frac{2 \cdot 1.1 y_{w'}}{9} \leq \frac{y_w}{3}$. But this means $|Ball_{\frac{1}{3}}(w') \cap T| \geq 5\ell$, which in turn means $x(Ball_{\frac{1}{3}}(w')) \geq 5\ell$ which contradicts Observation A.2.

Order the vertices w of X' in increasing order of $\frac{\min\{d(u,w),d(v,w)\}}{y_w}$, and let $Q = (t_1, t_2 \ldots t_{|X'|})$ denote this sequence. Notice that if some t_i is considered before t_j for some $i < j$ in π, then the edge (u,v) will already be cut/removed in one cluster when t_i is considered and hence we cannot cut this edge while considering t_j.

Also, once we obtain the clusters, the edge (u,v) is cut only if we pick a cluster containing either u or v, which happens with probability at most $\frac{2}{n}$.

$$\begin{aligned}
Pr[u,v \text{ separated}|\delta] &\leq \frac{2}{n}\sum_{i=1}^{|X'|} Pr[u,v \text{ separated while considering } t_i] \\
&\leq \frac{2}{n}\sum_{i=1}^{|X'|} Pr[t_i \text{ appears before } t_1, t_2 \ldots t_{i-1} \\
&\quad \text{and } \min\{d(u,t_i), d(v,t_i)\} \leq Cy_{t_i} \leq \max\{d(u,t_i), d(v,t_i)\}] \\
&\leq \frac{2}{n}\sum_{i=1}^{|X'|} \frac{|d(u,t_i) - d(v,t_i)|}{0.05 y_{t_i} i} \\
&\leq \frac{2}{n}\sum_{i=1}^{|X'|} \frac{d(u,v)}{0.05\delta i} \\
&\leq \frac{2}{n} 20 \log \ell \cdot \frac{d(u,v)}{\delta}
\end{aligned}$$

Here, we used the fact that $|d(u,t_i) - d(v,t_i)| \leq d(u,v)$. The last step follows since $|X'| \leq 10\ell$. Finally, as observed before, the edge $\{u,v\}$ can be separated only when $\frac{y_u}{2.2} \leq \delta \leq \frac{y_u}{0.9}$ or $\frac{y_v}{2.2} \leq \delta \leq \frac{y_v}{0.9}$. We now obtain

$$\begin{aligned}
Pr[u,v \text{ separated}] &\leq \int_{\frac{y_u}{2.2}}^{\frac{y_u}{0.9}} Pr[u,v \text{ separated}|\delta] Pr[\delta] d\delta + \int_{\frac{y_v}{2.2}}^{\frac{y_v}{0.9}} Pr[u,v \text{ separated}|\delta] Pr[\delta] d\delta \\
&\leq 20 \log \ell \cdot \frac{2}{n} \left(\int_{\frac{y_u}{2.2}}^{\frac{y_u}{0.9}} \frac{d(u,v)}{\delta} d\delta + \int_{\frac{y_v}{2.2}}^{\frac{y_v}{0.9}} \frac{d(u,v)}{\delta} d\delta \right) \\
&\leq \frac{100}{n} \log \ell \cdot d(u,v).
\end{aligned}$$

□

The rest of the analysis is similar to [7], however we present it again for completeness' sake. For a set $X \subseteq V(G)$, recall that we defined $f(X) = x(X) - \frac{\ell}{10s}|X| - \frac{\ell}{200LP\log\ell}|\delta_G(X)|$ where LP is the optimal LP value. Then if C is the set returned by the algorithm, we must have $E[f(C)] = E[x(X)] - \frac{\ell}{200LP\log\ell}E[|\delta_G(C)|] - \frac{\ell}{10s}E[|X|] \geq \sum_{v\in T}\frac{y_v}{2n}x(v) - \frac{\ell}{200LP\log\ell}\frac{50\log\ell LP}{n} - \frac{\ell}{10sn}\frac{2s}{3} \geq \frac{\ell}{100n}$, where we use the fact that $\sum_{v\in T}x(v)y_v \geq \frac{3\ell}{4}$, and the fact that $E[|X|] \leq \sum_v \frac{2y_v}{3n} \leq \frac{2s}{3n}$. Also note that $f(C) \leq 2\ell$. It follows that $E[f(C)] \leq 2\ell \cdot Pr[f(C) > 0]$, which means $Pr[f(C) > 0] \geq \frac{E[f(C)]}{2\ell} \geq \frac{1}{200n}$. Thus repeating the algorithm $\mathcal{O}(n\log n)$ times, we are guaranteed to find a set with $f(C) > 0$ with high probability. But $f(C) > 0$ means that C satisfies $\delta_G(C) \leq \mathcal{O}(\log\ell)\frac{x(C)}{\ell}LP \leq \mathcal{O}(\log\ell)\frac{x(C)}{\ell}OPT$. Also $f(C) > 0$ implies $x(C) \geq \frac{\ell}{10s}|C|$.

Recall that the algorithm keeps finding such a set C and sets $Y = Y \cup C$ till we obtain $|Y| \geq s$ or $x(Y) \geq \frac{\ell}{4}$. If $|Y| \geq s$, since $f(C) > 0$ in each iteration, it is clear that $x(Y) \geq \frac{\ell}{10s}s = \frac{\ell}{10}$. Note that we must have $|Y| \leq 4s$, since $|C| \leq 2s$ for every set C that we find.

Finally, note that $x(Y) \leq \frac{\ell}{4} + 2\ell = 3\ell$ since $x(C) \leq 2\ell$ for each C that we find. This gives $|\delta_G(Y)| \leq \mathcal{O}(\log\ell)\frac{x(Y)}{\ell}LP \leq \mathcal{O}(\log\ell)LP \leq \mathcal{O}(\log\ell)opt$ as desired. □

The next theorem adapts the above theorem to the vertex version to obtain Theorem 5.4. Here we only need a simpler version for our results, so we do not have the weight function x. Yet another marked difference is that we no longer return a small set. For convenience, we recall Theorem 5.4.

Theorem 5.4. *Given a graph G with a set of terminals T, suppose that there is a vertex cut (L, C, R) with $|R \cap T| \geq |L \cap T| = s$, $|C| = k$ and terminal sparsity $\phi = \frac{k}{s+|C\cap T|}$. Further suppose that $s \geq k\log s$ (so that $\phi \leq \mathcal{O}(\frac{1}{\log s})$). Then there is a polynomial time algorithm that finds a cut (L', C', R') with terminal sparsity $\phi' = \frac{|C'|}{\min\{|(L'\cup C')\cap T|, |(R'\cup C')\cap T|\}} \leq \mathcal{O}(\phi\log s)$.*

Proof. The proof will essentially be a modified algorithm based on the edge version. Now we have an additional set of variables b_v which indicate if vertex v is in the cut. Thus, for the canonical integral solution, we have $y_v = 1$ for every vertex $v \in L$ and $b_v = 1$ for every vertex $v \in C$, and $d(u, v) = 1$ if $u \in L$ and $v \notin L$, or $u \in R$ and $v \notin R$.

$$\min \sum_{v\in V(G)} b_v$$

$$\sum_{u\in T} \min\{d(u, v), y_v\} \geq (t - s)y_v \ \forall v \in T$$

$$\sum_{v\in T} y_v \geq s$$

$$d(u, v) \leq d(u, w) + d(w, v) \ \forall u, v, w \in V(G)$$

$$d(u, v) \leq d(u, w) + b_v \ \forall u, v, w \in V(G), (w, v) \in E(G)$$

$$d(u, v) = d(v, u) \ \forall u, v \in V(G)$$

$$d(u, v) \geq y_u - y_v \ \forall u, v \in V(G)$$

$$d(u, v), y_v \geq 0 \ \forall u, v \in V(G)$$

It is clear that the LP is a relaxation and the LP value is at most opt, where opt denotes the size of the optimal cut ($opt = |C|$).

The rounding: Having obtained the optimal values $d(u,v), b_v, y_v$ from the optimal LP solution, we now describe our rounding procedure. Our goal will again be to produce an LP-separator as introduced in [7], but with the improved approximation ratio $\mathcal{O}(\log s)$. Before describing our rounding scheme, we begin with a crucial observation that will be used multiple times.

Observation A.4. *For any vertex $v \in T$ and $r < 1$, let $Ball_r(v) = \{u \in V(G) \mid d(u,v) \leq r y_v\}$. Then for every $v \in T$, we must have $|Ball_r(v) \cap T| \leq \frac{s}{1-r}$.*

Proof. Let $|Ball_r(v) \cap T| = z$. The LP spreading constraint for vertex v implies that we must have $z \cdot r y_v + (t-z) \cdot y_v \geq (t-s) y_v$. This in turn means that $z \leq \frac{s}{1-r}$. □

Algorithm 5 Rounding the LP

$C \to \emptyset$.
for $j = 1$ to $\mathcal{O}(n \log n)$ **do**
  Pick a threshold $\delta \in [0,1]$ uniformly at random.
  Let $X = \{v \in T \mid y_v \in [\delta, 2\delta]\}$ and $\pi : [|X|] \to X$ be a random permutation of the elements of X. Let $U \leftarrow \emptyset$
  Pick $r \in [0.05, 0.1]$ uniformly at random.
  for $i = 1, 2 \ldots |X|$ **do**
    Let $C = \{v \in V(G) \setminus U \mid d(\pi(i), v) \leq r y_{\pi(i)}\}$
    $U \leftarrow U \cup C$
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{C\}$
  end for
  if $|U \cap T| < \frac{|T|}{2}$ **then return** $U' \leftarrow U$
  else
    Divide the clusters of $\mathcal{S}$ into two groups, $\mathcal{S}_1$ and $\mathcal{S}_2$, such that each group has $\geq \frac{|(U \cap T)|}{3}$ terminals.
    return the group of vertices U' with at most $\frac{|T|}{2}$ terminals.
  end if
  if $f(U') > 0$ **then**
    break
  end if
end for

Algorithm 5 describes the rounding scheme. For a set $X \subseteq V(G)$ we define $f(X) = |X \cap T| - \frac{s}{2000 LP \log s} |N_G(X)|$ where LP is the optimal LP value.

Everytime we pick a vertex $\pi(i)$ and separate out a cluster C, we will call $\pi(i)$ the *center* of the cluster C. Also, we will say that a vertex u is separated while considering $\pi(i) \in T$ if some neighbour of u is in C but $u \notin C$, and neither u nor its neighbours are in any cluster whose center is $\pi(1), \pi(2) \ldots \pi(i-1)$.

Lemma A.5. *In each iteration, the output set U' satisfies the following properties.*

1. *$E[U' \cap T] \geq \frac{s}{6}$.*
2. *The expected number of vertices cut, $E[|N_G(U')|] \leq \mathcal{O}(\log s) LP$.*

Proof. Clearly, for any $v \in T$ $y_v \in [\delta, 2\delta]$ whenever $\delta \in [\frac{y_v}{2}, y_v]$ which happens with probability at least $\frac{y_v}{2}$. This means that with probability at least $\frac{y_v}{2}$, v is in some cluster $W \in \mathcal{S}$. It follows that $E[U' \cap T] \geq \frac{1}{3} \sum_v \frac{y_v}{2} \geq \frac{s}{6}$.

Next, we prove (2). Consider a vertex u. First, suppose that $b_u \geq \frac{\delta}{100}$. For a given vertex u, the probability that this happens is at most $100b_u$. Now suppose that $b_u < \frac{\delta}{100}$, and let us consider the probability that the vertex u is cut while considering the ball around the terminal vertex w.

In order to cut u while growing the ball from w, there must be a neighbour u' of w inside the ball of w of radius $0.1y_w$. Hence we must have $d(u,w) \leq 0.1y_w + b_w \leq 0.1y_w + \frac{\delta}{100} \leq 0.1y_w + 0.1y_w \leq 0.2y_w$. This means, using the triangle inequalities in the LP constraints, we must have $0.8y_w \leq y_u \leq 1.2y_w$. But $y_w \in [\delta, 2\delta]$, and hence $0.8\delta \leq y_u \leq 2.4\delta$, or equivalently $\frac{y_u}{2.4} \leq \delta \leq \frac{y_u}{0.8}$. Thus, the only values of δ for which the vertex u can be cut must satisfy $\frac{y_u}{2.4} \leq \delta \leq \frac{y_u}{0.8}$.

Now, fix a value of δ such that either $\frac{y_u}{2.4} \leq \delta \leq \frac{y_u}{0.8}$.

Let $X' \subseteq X$ be the set of vertices w of X that satisfy $d(w,u) \leq 0.2y_w$. Informally, these are the only vertices that can separate u, thereby adding u as a cut vertex. We show first that $|X'| \leq 5s$. Suppose to the contrary that $|X'| > 5s$. Then at least $5s+1$ vertices $w \in X'$ that satisfy $d(w,u) \leq 0.2y_w$. Fix some $w' \in X'$. By the triangle inequality, for every $w \in X'$ we have $y_w + 0.2y_w \geq y_u \geq y_w - 0.2y_w = 0.8y_w$. Thus $d(w,u) \leq \frac{y_u}{4}$. Also $d(w',u) \leq \frac{y_u}{4}$. By the triangle inequality on d, we must have $d(w',w) \leq d(w,u) + d(w,w') \leq \frac{2y_u}{4} = \frac{y_u}{2} \leq 0.6y_w$. But this means $|Ball_{0.6}(w')| \geq 5s$, which contradicts Observation A.4.

Order the vertices w of X' in increasing order of $\frac{d(u,w)}{y_w}$, and let $Q = (t_1, t_2 \ldots t_{|X'|})$ denote this sequence. Notice that if some t_i is considered before t_j for some $i < j$ in π, then the vertex u will already be cut/removed in one cluster when t_i is considered and hence we cannot cut this edge while considering t_j.

$$\begin{aligned}
Pr[u \text{ cut}|\delta] &\leq \sum_{i=1}^{|X'|} Pr[u \text{ cut while considering } t_i] \\
&\leq \sum_{i=1}^{|X'|} Pr[t_i \text{ appears before } t_1, t_2 \ldots t_{i-1} \\
&\quad \text{and } d(u,t_i) - b_u \leq C y_{t_i} \leq d(u,t_i)] \\
&\leq \sum_{i=1}^{|X'|} \frac{b_u}{0.05 y_{t_i} i} \\
&\leq 100 \log s \cdot \frac{b_u}{\delta}
\end{aligned}$$

The last step follows since $|X'| \leq 10s$. Finally, as observed before, the vertex u can only be cut when $\frac{y_u}{2.4} \leq \delta \leq \frac{y_u}{0.8}$. We now obtain

$$\begin{aligned}
Pr[u \text{ cut}] &\leq \int_{\frac{y_u}{2.4}}^{\frac{y_u}{0.8}} Pr[u \text{ cut}|\delta] Pr[\delta] d\delta \\
&\leq 100 \log s \cdot \int_{\frac{y_u}{2.4}}^{\frac{y_u}{0.8}} \frac{b_u}{\delta} d\delta \\
&\leq 200 \log s \cdot b_u.
\end{aligned}$$

□

Finally, this above expression only bounds the probability that u is cut assuming that $b_u \leq \frac{\delta}{100}$. To account for the other case, as discussed above, we note that the probability that $b_u \geq \frac{\delta}{100}$ is at

most $100b_u$. Thus the total probability that u is cut is at most $300 \log s \cdot b_u$. The rest of the analysis is similar to [7], however, we present it again for completeness' sake. For a set $X \subseteq V(G)$, recall that we defined $f(X) = |X \cap T| - \frac{s}{2000LP \log s}|N_G(X)|$ where LP is the optimal LP value. Then if U' is the set returned by the algorithm, we must have $E[f(U')] = E[U' \cap T] - \frac{s}{2000LP \log s}E[N_G(U')] \geq \frac{s}{6} - \frac{3s}{20} \geq \frac{s}{60}$. Also note that $f(U') \leq n$. It follows that $E[f(U')] \leq n \cdot Pr[f(U') > 0]$, which means $Pr[f(U') > 0] \geq \frac{E[f(U')]}{n} \geq \frac{s}{60n}$. Thus repeating the algorithm $\mathcal{O}(n \log n)$ times, we are guaranteed to find a set with $f(U') > 0$ with high probability. But $f(U') > 0$ means that U' satisfies $|N_G(U')| \leq \mathcal{O}(\log s)\frac{LP}{s}|U' \cap T| \leq \mathcal{O}(\log s)\frac{k}{s}|U' \cap T|$. Note that since we assumed $s = \Omega(k \log s)$, we must have $|N_G(U')| \leq \frac{1}{2}|U' \cap T|$. Finally, we have $|(U' \cup N_G(U')) \cap T| \leq \frac{1.5|T|}{2}$ and thus $|(V(G) \setminus (U' \cup N_G(U')) \cap T| \geq \frac{|T|}{4} = \Omega(U' \cap T)$. It follows that U' must be $\mathcal{O}(\frac{|N_G(U')|}{|(U' \cup N_G(U')) \cap T|}) \leq \mathcal{O}(\log s)\phi$-terminal sparse. □