

The Moser–Tardos Framework with Partial Resampling

DAVID G. HARRIS, Department of Computer Science, University of Maryland, College Park, MD 20742
ARAVIND SRINIVASAN, Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742

The resampling algorithm of Moser and Tardos is a powerful approach to develop constructive versions of the Lovász Local Lemma. We generalize this to *partial* resampling: When a bad event holds, we resample an appropriately random *subset* of the variables that define this event rather than the entire set, as in Moser and Tardos. This is particularly useful when the bad events are determined by sums of random variables. This leads to several improved algorithmic applications in scheduling, graph transversals, packet routing, and so on. For instance, we settle a conjecture of Szabó and Tardos (2006) on graph transversals asymptotically and obtain improved approximation ratios for a packet routing problem of Leighton, Maggs, and Rao (1994).

CCS Concepts: • **Mathematics of computing** → **Combinatorial algorithms; Probabilistic algorithms;**

Additional Key Words and Phrases: Lovász local lemma, resampling algorithm, discrepancy, packet routing, independent transversal

ACM Reference format:

David G. Harris and Aravind Srinivasan. 2019. The Moser–Tardos Framework with Partial Resampling. *J. ACM* 66, 5, Article 36 (August 2019), 45 pages.

<https://doi.org/10.1145/3342222>

1 INTRODUCTION

The Lovász Local Lemma (LLL) [9] is a fundamental probabilistic tool. The breakthrough of Moser and Tardos [28] gives a constructive approach to the LLL through a very natural resampling procedure, which we summarize as follows. Suppose we have a collection \mathcal{B} of “bad” events, each $B \in \mathcal{B}$ being a Boolean function of a subset of *independent* random variables X_1, X_2, \dots, X_n . Then, assuming that the standard sufficient conditions of the LLL hold, the following resampling algorithm (which we refer to as the *MT algorithm*) quickly converges to a setting of the X_j ’s that simultaneously avoids all the bad events in \mathcal{B} :

- Sample X_1, \dots, X_n independently from their respective distributions;
- **while** some bad event is true, pick one of these, say, B , arbitrarily, and resample (independently) all the variables determining it.

Preliminary versions of parts of this article appeared in two papers by the authors [13, 14].

David G. Harris: Research supported in part by NSF Awards CNS-1010789 and CCF-1422569.

Aravind Srinivasan: Research supported in part by NSF Awards CNS-1010789, CCF-1422569, and CCF-1749864, a gift from Google, Inc., and research awards from Adobe, Inc. and Amazon, Inc.

Authors’ addresses: D. G. Harris and A. Srinivasan, Department of Computer Science, University of Maryland, A. V. Williams Building, 8223 Paint Branch Drive, College Park, MD 20742; emails: davidgharris29@gmail.com, srin@cs.umd.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.

0004-5411/2019/08-ART36 \$15.00

<https://doi.org/10.1145/3342222>

We generalize this to an algorithm that we call the Partial Resampling Algorithm (PRA); the idea is that when a bad event B is true, we randomly select a *subset* D of its variables, according to some carefully designed probability distribution, and then only resample the variables X_j contained in D . This partial-resampling approach leads to algorithmic results for many applications that are not captured by the LLL.

As a starting point, suppose the bad events are defined by non-negative linear threshold functions. In such cases, the constraints (i.e., the complements of the $B \in \mathcal{B}$) have the form

$$\sum_{i,j} a_{k,i,j} [X_i = j] \leq b_k. \quad (1)$$

Here we use the Iverson notation: Given an event \mathcal{E} , the notation $[\mathcal{E}]$ will stand for the indicator variable for \mathcal{E} . Thus, $[X_i = j]$ is the indicator variable that variable X_i takes on value j . The matrix A of coefficients $a_{k,i,j}$ has m rows indexed by k and N columns that are indexed by pairs (i, j) ; by scaling, we will assume throughout that $a_{k,i,j} \in [0, 1]$.

Low-congestion routing is a motivating example of this type of problem: We are given a collection of (source, destination) pairs $(s_1, t_1), \dots, (s_n, t_n)$ in a directed or undirected graph G with edge-set E ; each edge $f \in E$ has a capacity b_f , and we are also given a collection $\mathcal{P}_i = \{P_{i,j}\}$ of possible routing paths for each (s_i, t_i) pair. We aim to choose one path from each collection \mathcal{P}_i to minimize the *relative congestion*: The minimal T such that the maximum load on any edge f is at most $T \cdot c_f$. This problem leads to a simple IP formulation:

$$\text{minimize } T \text{ subject to } \left[\forall i, \sum_j z_{i,j} = 1; \forall f \in E, \sum_{(i,j): f \in P_{i,j}} z_{i,j} \leq T \cdot c_f; z_{i,j} \in \{0, 1\} \right],$$

where $z_{i,j}$ is an indicator variable for selecting path $P_{i,j}$.

There are two broad approaches to such problems, both starting with the natural LP relaxation where the variable $z_{i,j} \in [0, 1]$ represents the fractional assignment of variable X_i to value j . Suppose that z satisfies the constraints $\sum_{i,j} a_{k,i,j} z_{i,j} \leq c_k$. The natural question is as follows:

“What choice of vector b , and what conditions on the matrix A and vector c , ensure that (1) has an integer solution, which, furthermore, can be found efficiently?”

The first major approach to this is polyhedral. Letting D denote the maximum column sum of A , i.e., $D = \max_{i,j} \sum_k a_{k,i,j}$, the rounding theorem of Reference [23] shows constructively that for all k ,

$$b_k = c_k + D \quad (2)$$

suffices; for the low-congestion routing problem, for instance, this would show that if each path has length D , then we can obtain an $O(D)$ -approximate solution.

The second approach is randomized rounding [30]: Independently set $X_i = j$ with probability $z_{i,j}$. The standard “Chernoff bound followed by a union bound over all m rows” analysis of Reference [30] shows that this works for

$$b_k = \begin{cases} C \cdot \frac{\log m}{1 + \log(\frac{\log m}{c_k})} & \text{if } c_k \leq \log m \\ c_k + C \sqrt{c_k \log m} & \text{if } c_k > \log m \end{cases}, \quad (3)$$

where C is some universal constant. In particular, the low-congestion routing problem can be approximated to within $O(\frac{\log m}{\log \log m})$, where m denotes the total number of edges.

Let us compare these known bounds (2) and (3). The former is good when all the c_k are “large” (say, much bigger than, or comparable to, D); the latter is better when D is large compared to

m . Can we do better? Our Theorem 4.7 answers this in the affirmative—we replace m by D in Equation (3), showing constructively that when $c_k = R \geq 1$ we may set

$$b_k = \begin{cases} C \frac{\log D}{1 + \log(\frac{\log D}{R})} & \text{if } R \leq \log D \\ R + C\sqrt{R \log D} & \text{if } R \geq \log D \end{cases}. \quad (4)$$

Thus, for the low-congestion routing problem, this would give an approximation ratio $O(\frac{\log D}{\log \log D})$, beating both the union bound and the polyhedral bounds.

We will show in Appendix A that the MT algorithm cannot directly get “scale-free” bounds such as Equation (4) (that is, b_k is a function of R, D but not of n or m). In such cases, the MT algorithm is no better than random search, requiring exponential time.

There are two other related results that deserve mention here. First, References [13, 24] show a bound similar to Equation (4) but with D' , the maximum number of non-zeroes in any column of A , playing the role of D . Note that $D' \geq D$ always and that $D' \gg D$ is possible. Moreover, the bound of Reference [24] primarily works when all the c_k are close to each other and rapidly degrades when these values can be disparate; the bound of Reference [13] is nonconstructive.

While we have discussed linear threshold functions here for simplicity, much of the power of the PRA comes from the fact that it is flexible enough to handle complex constraints that have both linear and non-linear components and thus behave “almost linearly.” By contrast, approaches such as Reference [23], which depend critically on linear algebra, and results such as Reference [24], based on multiple phases of resampling, are more difficult to adapt to such problems. Let us summarize two non-linear problems where we obtain improved bounds:

Transversals with omitted subgraphs. Given a partition of the vertices of an undirected graph G into blocks, a *transversal* is a subset of the vertices, one chosen from each block. An *independent transversal*, or independent system of representatives, is a transversal that is also an independent set in G . The study of independent transversals was initiated by Bollobás, Erdős and Szemerédi [6] and has received a considerable amount of attention (see, e.g., References [1–3, 19, 21, 22, 27, 35, 36]). Such transversals serve as building blocks for other graph-theoretic parameters such as the linear arboricity and strong chromatic number [2, 3].

We improve (algorithmically) a variety of sufficient conditions for the existence of certain transversals. In particular, Szabó and Tardos present a conjecture on the minimum block size to guarantee the existence of transversals that avoid K_s [35]; we show that this conjecture holds asymptotically for large s .

Packet routing with low latency. Consider an undirected graph G with N packets, in which we need to route each packet i from vertex s_i to vertex t_i along a *given simple path* P_i . The constraints are that each edge can carry only one packet at a time, and each edge traversal takes unit time for a packet; edges are allowed to queue packets. A well-known scheduling problem considered in Reference [25] is to minimize the *makespan* T (the time by which all packets are delivered).

Two natural lower-bounds on T are the *congestion* C (the maximum number of the P_i that contain any given edge of G) and the *dilation* D (the length of the longest P_i); thus, $(C + D)/2$ is a universal lower-bound, and there exist families of instances with $T \geq (1 + \Omega(1)) \cdot (C + D)$ [31]. A seminal result of Reference [25] is that $T \leq O(C + D)$ for all input instances, using constant-sized queues at the edges; the asymptotic notation hides a rather large constant. Building on further improvements [29, 32], our work in Reference [13] developed a nonconstructive $7.26(C + D)$ and a constructive $8.84(C + D)$ bound; we improve these further to a constructive $6.73(C + D)$.

Informal discussion of the Partial Resampling Algorithm. To understand the intuition behind the PRA, consider bad events of the form $[X_{i_1} = j_1] + \dots + [X_{i_k} = j_k] \geq t$, where the expected

value of $[X_{i_1} = j_1] + \dots + [X_{i_k} = j_k]$ is $\mu < t$. Suppose we run the MT algorithm on this problem. We begin by drawing all the variables X_1, X_2, \dots, X_n independently from their original distributions, and we find some such bad event has become true. At this point, the MT algorithm would resample all of the variables affected by this event—that is, all the variables X_{i_1}, \dots, X_{i_k} .

But suppose now that $X_{i_\ell} \neq j_\ell$ for some $\ell \in [k]$. In that case, variable X_{i_ℓ} seems like it is “helpful” in terms of avoiding this bad event. Since the goal of resampling a bad event is to “fix” it, then resampling this X_{i_ℓ} seems counter-productive. Thus it seems more appropriate to only resample the variables with $X_{i_\ell} = j_\ell$. In the original step of the MT algorithm, the expected number of such variables is μ , and we expect that in intermediate stages of the MT algorithm it should also be close to μ . Thus, heuristically, we should only be resampling about μ variables, not all k variables. In fact, even this is somewhat too many variables to resample. Since we expect about μ variables to have $X_{i_\ell} = j_\ell$, it is only the $t - \mu$ “extra” variables that are causing the bad event to occur. Thus, we only sample some of the X_{i_j} ’s, and we make this choice probabilistically.

The power of the partial-resampling approach comes from the fact it makes steady progress toward a solution—when there is a bad event, we make a minimal change to fix it, while preserving as much of the prior solution as possible. For these linear-threshold bad events, it is only these few, “guilty” variables that should be resampled. We thus make much smaller steps to fix any bad event, making steady progress toward a solution that avoids them all.¹

In general, the PRA tends to work well when there are common configurations, which are not actually forbidden, but are nonetheless “bad” in the sense that they increase the probability of a bad event. In the case of a sum of random variables, for example, this occurs whenever $X_{i_\ell} = j_\ell$ holds simultaneously for many values of ℓ . We will see other examples of more complicated types of bad-but-legal configurations.

Organization of the article. The PRA is discussed in detail in Section 2. We give criteria, similarly to the cluster-expansion LLL, asymmetric LLL, and symmetric LLL, for showing that this algorithm terminates in expected polynomial time. Section 3 describes how to apply the PRA when the bad events are complex (such as linear-threshold functions) as opposed to pure atomic events. It also provides a more succinct formulation that allows us to reduce the number of parameters. Section 4 shows how to apply the PRA when the underlying bad events are linear threshold functions. Such events are ubiquitous in combinatorics and algorithms, and the PRA deals with them particularly effectively. We discuss applications to column-sparse packing (Section 5), transversals with omitted subgraphs (Section 6), and packet routing (Section 7).

2 THE PARTIAL RESAMPLING ALGORITHM

2.1 The Variable-assignment Setting

Our algorithms and problems all come from a general class of constraint-satisfaction problems (CSP’s) we refer to as the *variable-assignment setting*. We have n variables X_1, \dots, X_n ; each variable has a finite set F_i of possible values. We define the probability space Ω in which the variables are assigned independently: Namely, for each $i \in [n]$, we set $X_i = j$ with probability $p_{i,j}$, where $j \in F_i$ ranges over the set of valid assignment to variable i . We often omit the set F_i when it is clear from context, e.g., we write simply $\sum_j p_{i,j} = 1$.

¹There is an alternative way to apply the LLL in this context, which is to define a separate bad event for each *atomic* bad configuration, that is, for each subset of variables that exceeds the value of t . This subdivides the original bad event into approximately $\binom{k}{t}$ separate smaller bad events. This approach can be effective in some regimes, especially when $t \gg \mu$, and can lead to scale-free configurations. However, this method suffers from the drawback that the sum of the probabilities of these atomic bad events is much larger than the original probability of the single bad event (because these atomic events have significant positive correlation). The method we develop will be strictly stronger than this approach.

As a starting point for our algorithm, we assume there is a collection \mathcal{B} of bad events to avoid, which are all *atomic events* in that each bad event B can be written as a conjunction of the form

$$B \equiv ((X_{i_1} = j_1) \wedge \cdots \wedge (X_{i_k} = j_k))$$

for $k \geq 1$.

We refer to any ordered pair (i, j) where $j \in F_i$, as an *element* and let \mathcal{X} denote the set of all elements. We may encode these types of atomic bad events by subsets of \mathcal{X} . We thus define the *family of atomic sets* $\mathcal{A} \subseteq 2^{\mathcal{X}}$ to be the collection of all non-empty sets $Y \subseteq \mathcal{X}$ of the form

$$Y = \{(i_1, j_1), \dots, (i_k, j_k)\} \quad \text{where } i_1, \dots, i_k \text{ are distinct and } j_\ell \in F_{i_\ell} \text{ for } \ell = 1, \dots, k.$$

We say that an atomic set $Y \in \mathcal{A}$ *holds* on a configuration X_1, \dots, X_n , if $X_i = j$ for all $(i, j) \in Y$. Thus, an atomic event $B \equiv ((X_{i_1} = j_1) \wedge \cdots \wedge (X_{i_k} = j_k))$ corresponds to the atomic set $A = \{(i_1, j_1), \dots, (i_k, j_k)\}$. Here B is true on X iff A holds on X . So \mathcal{B} can be regarded as subset of \mathcal{A} .

Given any vector $\lambda \in [0, 1]^{\mathcal{X}}$, and any $Y \in \mathcal{A}$, we define

$$\lambda^Y = \prod_{x \in Y} \lambda_x. \quad (5)$$

Thus, for instance, for any $Y \in \mathcal{A}$, we have $P_\Omega(Y \text{ holds on } X) = p^Y$.

2.2 Partial Resampling Algorithm and the Main Theorem

Our algorithm is driven by a parameter, the *fractional hitting-set*, which tells which variables to resample for which bad event.

Definition 2.1 (Fractional Hitting-Set). Let $C : \mathcal{A} \rightarrow [0, 1]$ and let $B \in \mathcal{A}$. We say that C is a *fractional hitting-set* for B if

$$\sum_{\substack{Y \subseteq B \\ Y \neq \emptyset}} C(Y) \geq 1;$$

note that this summation treats Y and B as subsets of \mathcal{X} .

We say that C is a *fractional hitting-set* for \mathcal{B} if C is a fractional hitting-set for every $B \in \mathcal{B}$.

We now introduce our main algorithm, which we refer to as the *Partial Resampling Algorithm (PRA)*; this requires as input a fractional hitting-set Q for the set of bad events \mathcal{B} .

ALGORITHM 1: PRA($p, \mathcal{B}, Q, \mathcal{X}$)

- 1: Draw the variables X_1, \dots, X_n independently, where $X_i = j$ with probability p_{ij} .
- 2: **while** there is a true bad event $B \in \mathcal{B}$ **do**
- 3: Select, arbitrarily, some true bad event $B \in \mathcal{B}$. We refer to this set B as the *violated set*.
- 4: Select exactly one subset $Y \subseteq B$. The probability of selecting a given Y is given by

$$\Pr(\text{select } Y) = \frac{Q(Y)}{\sum_{Z \subseteq B} Q(Z)}.$$

We refer to Y as the *resampled set*.

- 5: For each $(i, j) \in Y$, draw a new value for X_i independently, using the probability distribution p_i . (We refer to this as *resampling* Y).
 - 6: **return** X
-

The main difference between the PRA and the MT algorithms is that the latter would resample *all* the variables of a true bad event. The PRA only resamples a (carefully chosen, random) subset

of these variable. In fact, with an appropriate choice of fractional hitting-set, the *trivial hitting-set*, the PRA essentially degenerates into the MT algorithm.

Definition 2.2 (Trivial Hitting-Set). We define the *trivial hitting-set* for \mathcal{B} by $C(Y) = [Y \in \mathcal{B}]$ (using Iverson notation as before).

We will need to keep track of how bad events (and subsets of bad events) can be interdependent. This is more complicated than the usual LLL setting, because we distinguish two different types of dependencies: Sets Y, Y' could share a variable, or they could both be potential resampling targets for some bad event. The LLL only needs to keep track of the first type of dependency. We introduce the symmetric relations \sim, \bowtie, \approx over \mathcal{A} to account for these dependency types. These definitions all depend on some fixed choice of fractional hitting-set Q ; we omit this from the notation for readability.

Definition 2.3 (Supported Event). We say that a set $Y \in \mathcal{A}$ is *supported* if $Q(Y) > 0$.

Definition 2.4 (Symmetric Relations \sim, \bowtie , and \approx on \mathcal{A}). Let $Y, Y' \in \mathcal{A}$. We say $Y \sim Y'$ iff there exists a triple (i, j, j') such that $(i, j) \in Y$ and $(i, j') \in Y'$: i.e., iff Y and Y' overlap in a variable. We also write $i \sim Y$ (or $Y \sim i$) to mean that $(i, j) \in Y$ for some j .

We say $Y \bowtie Y'$ iff $Y \approx Y'$ and there is some event $B \in \mathcal{B}$ with $Y, Y' \subseteq B$.

We say $Y \approx Y'$ iff either (1) $Y \sim Y'$ or (2) $Y \bowtie Y'$.

We say that \bowtie is *null* if for all supported Y, Y' we have $Y \not\bowtie Y'$.

Note that if $Y, Y' \subseteq B$ for some bad event B , then $Y \approx Y'$.

In Theorem 2.6, we give three conditions for the PRA to terminate. These conditions are analogous to, respectively, the cluster-expansion LLL criterion [5], the asymmetric LLL, and the symmetric LLL. Conditions (b) and (c) follow immediately from (a); however, we include (b) and (c) here, since they are more convenient for typical applications. We begin with a preliminary definition.

Definition 2.5 (Neighbor-Set for Y). For any $Y \in \mathcal{A}$, we say that a set $\mathcal{T} \subseteq \mathcal{A}$ is a *neighbor-set* for Y (and we write $\mathcal{T} \in \text{NS}(Y)$) if the following conditions hold:

- (1) Every $Z \in \mathcal{T}$ satisfies $Z \approx Y$.
- (2) There do not exist distinct $Z, Z' \in \mathcal{T}$ with $Z \sim Z'$.
- (3) There is at most one $Z \in \mathcal{T}$ with $Z \bowtie Y$.

THEOREM 2.6 (MAIN THEOREM). *Let Q be a fractional hitting-set for \mathcal{B} , and let $p \in [0, 1]^X$ be the probability vector for X . In each of the following three cases, the PRA terminates in a feasible configuration avoiding all bad events with probability 1.*

(a) *Suppose that $\mu : \mathcal{A} \rightarrow [0, \infty)$ satisfies, for all $Y \in \mathcal{A}$,*

$$\mu(Y) \geq p^Y Q(Y) \sum_{\mathcal{T} \in \text{NS}(Y)} \prod_{Y' \in \mathcal{T}} \mu(Y').$$

Then, the expected number of resamplings is at most $\sum_Y \mu(Y)$.

(b) *Suppose that $\mu : \mathcal{A} \rightarrow [0, \infty)$ satisfies, for all $Y \in \mathcal{A}$,*

$$\mu(Y) \geq p^Y Q(Y) \left(\prod_{Y' \sim Y} (1 + \mu(Y')) \right) \left(1 + \sum_{Y'' \bowtie Y} \mu(Y'') \right).$$

Then, the expected number of resamplings is at most $\sum_Y \mu(Y)$.

(c) *Suppose that $p^Y Q(Y) \leq P$ for all $Y \in \mathcal{A}$; and suppose that for all supported Y , there are at most D supported Y' with $Y' \approx Y$ (we allow $Y' = Y$ here). And suppose, finally, that $ePD \leq 1$. Then, the expected number of resamplings is at most $e \sum_Y p^Y Q(Y)$.*

The proof of Theorem 2.6 is lengthy; we will spend the next sections showing a number of preliminary lemmas.

2.3 Analyzing the PRA: Witness Trees and the Resampling Table

Our analysis of the PRA is based on *witness trees* and a coupling construction called the *resampling table*, two proof techniques developed by Moser and Tardos [28].

We first describe the resampling table. In the PRA we have described, we generate new values for the variables X_i as they are needed. Instead, we may imagine generating a resampling table $R(i, k)$, where i ranges over $[n]$ and k ranges over positive integers, whose entries are independent random variables where entry $R(i, k)$ is drawn from the distribution p_i . Once we have drawn this table, we use its values in place of resampling. For instance, in step (1) of the PRA, we set $X_i = R(i, 1)$ for every $i \in [n]$. The first time we resample X_i , we set $X_i = R(i, 2)$, and so on. It is clear that pre-generating the randomness does not affect the behavior of the PRA. We also observe that after drawing the resampling table, the only source of randomness remaining in the PRA is the choice of which subset $Y \subseteq B$ to select when resampling a bad event B .

The witness tree $\hat{\tau}^t$ describes the history of the variables involved in the resampling up to some time t . Suppose we run the PRA for t timesteps (not necessarily to completion) and that at each time $k = 1, \dots, t$ the violated bad event is B_k , with resampled set $Y_k \subseteq B_k$. We define U_k to be the ordered pair (Y_k, B_k) for each time k . We will construct an associated random variable $\hat{\tau}^t$, which is a type of rooted labeled tree.

To begin, we recursively define $\hat{\tau}_k^t$ for $k = t, t-1, \dots, 1$. We begin with $\hat{\tau}_t^t$, which has a single root node labeled Y_t . Next, letting $L(v)$ denote the label of a node v , we define M_k^t for each $k < t$ to be the set of nodes $v \in \hat{\tau}_{k+1}^t$ with either of the two properties:

- (1) $L(v) \sim Y_k$; OR
- (2) $L(v) \bowtie Y_k$ and v does not have any child u such that $L(u) \bowtie L(v)$.

If M_k^t is empty, then we update $\hat{\tau}_k^t = \hat{\tau}_{k+1}^t$. Otherwise, we select some node $v \in M_k^t$ of greatest depth in $\hat{\tau}_{k+1}^t$ (breaking ties arbitrarily) and form $\hat{\tau}_k^t$ by adding a new node that is a child of v and is labeled by Y_k . We finish by defining $\hat{\tau}^t = \hat{\tau}_1^t$.

Note that the witness tree $\hat{\tau}^t$ does *not* record information about the violated sets B_k themselves; this is a critical step to cut down the number of possible trees.

By convention, for $s > t$ we define $\hat{\tau}_s^t = \emptyset$ (the null tree, which does not contain any nodes – not even the root node). If the PRA has terminated already by time t , then we also define $\hat{\tau}^t = \emptyset$.

We define a *tree-structure* to be any rooted tree whose nodes are labeled from the set \mathcal{A} . Every $\hat{\tau}^t$ is a tree-structure. We say that a tree-structure τ *appears* if $\hat{\tau}^t = \tau$ for any value of t . We define the *weight* of tree-structure τ by

$$w(\tau) = \prod_{v \in \tau} p^{L(v)} Q(L(v)).$$

Remark. Recall the notation (5) in parsing the value “ $p^{L(v)}$ ” above.

We note a few simple facts about witness trees.

PROPOSITION 2.7. *If $t < t'$ and the PRA has not terminated by time t' , then $\hat{\tau}^t \neq \hat{\tau}^{t'}$.*

PROOF. Let $Y_t, Y_{t'}$ be resampled sets at times t, t' respectively. If $Y_t \neq Y_{t'}$, then the result holds, because the root nodes of $\hat{\tau}^t, \hat{\tau}^{t'}$ have different labels. Otherwise, note that for every time s with $Y_s = Y_t$, we will have $M_s^t \neq \emptyset$ and $M_s^{t'} \neq \emptyset$ (since the root nodes of $\hat{\tau}^t$ and $\hat{\tau}^{t'}$ respectively will be in these sets). Thus, $\hat{\tau}^{t'}$ will contain strictly more nodes with label Y_t than does $\hat{\tau}^t$. \square

PROPOSITION 2.8. *If v, v' are distinct nodes at depth h in $\hat{\tau}^t$, then $L(v) \approx L(v')$. Furthermore, if both v, v' are leaf nodes, then $L(v) \not\bowtie L(v')$.*

PROOF. Suppose that v, v' correspond to resamplings at time s, s' respectively where $s < s' \leq t$. If $L(v) \sim L(v')$, then $v' \in \hat{\tau}_s^t$ and so $v' \in M_s^t$. So v will be eligible to be placed as a child of v' or some other node of greater depth. In particular, v will be placed at depth strictly larger than h . If v, v' are leaf nodes and $L(v) \bowtie L(v')$, then, again $v' \in M_s^t$ so the same argument would apply. \square

2.4 The Witness Tree Lemma

The key to analyzing the PRA is the following lemma:

LEMMA 2.9 (WITNESS TREE LEMMA). *Any tree-structure τ appears with probability at most $w(\tau)$.*

We prove this in two stages. We first connect the behavior of a witness trees to the resampling table, and show certain necessary conditions on the entries of R . This part of the proof is nearly identical to that of Moser and Tardos. Second, we show that even after fixing R , there is still enough randomness remaining in the PRA to further bound the probability of a tree-structure appearing.

Let r be a possible value for the resampling table. We say a tree-structure τ is *compatible* with r if there is a non-zero probability of $\hat{\tau}^t$ appearing, conditioned on $R = r$.

PROPOSITION 2.10 ([28]). *The probability that tree-structure τ is compatible with R is at most $\prod_{v \in \tau} p^{L(v)}$.*

PROOF. Let $i \in [n]$, and consider the set of nodes $v \in \tau$ such that $L(v) \sim i$. By Proposition 2.8, all such nodes are at distinct depths in τ . Let us sort these nodes in decreasing order of depth as v_1, \dots, v_{s_i} ; for $k = 1, \dots, s_i$ let us say that $(i, a_{ik}) \in L(v_k)$.

Suppose that $\hat{\tau}^t = \tau$ for some time t . We claim that $R(i, 1) = a_{i1}$. For, suppose not; consider the first time t' that $L(v_1)$ is the resampled set (this must occur, since $v_1 \in \hat{\tau}^t$). At time t' , we have $X_i = a_1$; so, X_1 must have been resampled earlier, at time t'' . Let Y'' be the resampled set at time t'' ; so $i \sim Y''$. But then when constructing $\hat{\tau}^t$, we would place an additional node labeled Y'' at greater depth than v_1 , leading to a contradiction.

Continuing in this way, we see that $R(i, k) = a_{ik}$ for $i \in [n]$ and $k = 1, \dots, s_i$. Since the entries of R are all independent, the overall probability of this event is

$$\prod_{i=1}^n \prod_{k=1}^{s_i} p_{i, a_{ik}}.$$

If $(i, j) \in L(v)$ for any node $v \in \tau$, then this contributes exactly one factor of p_{ij} to this expression. So, we can rearrange this term as

$$\prod_{i=1}^n \prod_{k=1}^{s_i} p_{i, a_{ik}} = \prod_{v \in \tau} \prod_{(i, j) \in L(v)} p_{ij} = \prod_{v \in \tau} p^{L(v)}. \quad \square$$

Before the second part of proof of the Witness Tree Lemma, let us give some intuition to the role of \bowtie .

Consider a tree-structure τ consisting of a singleton node labeled Y . We want to show that τ appears with probability at most $w(\tau) = p^Y Q(Y)$. We have already seen that the probability that R is compatible with τ is at most p^Y . We would next like to say that the probability that Y was selected as the resampled set is at most $Q(Y)$, giving us our probabilistic bound.

On any given instance in which Y is eligible to be the resampled set, the probability of selecting Y is indeed at most $Q(Y)$. However, there may have been a long sequence of bad events in which Y was eligible yet not selected. With enough of these opportunities, the probability of eventually

selecting Y approaches 1. Thus to obtain a useful bound on the probability of selecting Y , one must distinguish in advance a *specific* instance in which Y is eligible.

Now observe that if $Y \subseteq B$ for some true bad event B , but we instead select some other $Y' \subseteq B$ as the resampled set, then $Y' \approx Y$. So Y' would be added as a child of Y in the witness tree. As τ is a singleton node, then a necessary condition for τ to appear is that Y is *selected the first time it is eligible to be selected*. With some thought, we see that this event has probability at most $Q(Y)$.

The reader should bear this intuition in mind for the remainder of the proof. Proposition 2.11 extends this to larger tree-structures, which can have more complex interactions.

PROPOSITION 2.11. *Suppose the PRA has not terminated by time s , and let $U_s = (Y_s, B_s)$. For some integer $t \geq s$, define J to be the set of leaf nodes v of $\hat{\tau}_s^t$ such that $L(v) \subseteq B_s$. Then:*

- (1) J cannot contain two nodes at the same depth.
- (2) If $J \neq \emptyset$ and v is the (unique) vertex in J of greatest depth, then $L(v) = Y_s$ and $\hat{\tau}_{s+1}^t = \hat{\tau}_s^t - v$.
- (3) If $J = \emptyset$, then $\hat{\tau}_{s+1}^t = \hat{\tau}_s^t$.

PROOF. If $s = t$, then $\hat{\tau}_s^t$ contains a single node v labeled Y_s and $\hat{\tau}_{s+1}^t = \emptyset$ and $J = \{v\}$. In this case all the three parts hold easily. So let assume that $s < t$.

Part (1) follows immediately from Proposition 2.8, noting that J contains only leaf nodes.

For (2), let v be the greatest-depth node of J . Suppose that $L(v) = Y' \neq Y_s$. We must have $v \in \hat{\tau}_{s+1}^t$ (the only node added to $\hat{\tau}_s^t$ has label Y_s). As $Y_s \approx Y'$, we have $v \in M_s^t$, and so $\hat{\tau}_s^t$ would have a new leaf node w labeled Y_s at greater depth than v ; but then $w \in J$ has greater depth than v , a contradiction.

Thus, we have shown that $L(v) = Y_s$. Observe that $\hat{\tau}_s^t$ is either equal to $\hat{\tau}_{s+1}^t$, or has a single leaf node w added to $\hat{\tau}_{s+1}^t$ labeled Y_s . In the latter case, w will have label Y_s and will be at greatest depth in $\hat{\tau}_s^t$, and so $w = v$ and part (2) follows. In the former case, $v \in \hat{\tau}_{s+1}^t$. Since $L(v) = Y_s$, we will have $v \in M_s^t$, so again $\hat{\tau}_s^t$ would have a new leaf node w labeled Y_s at greater depth than v ; but then $w \in J$ has greater depth than v , a contradiction.

To show (3), suppose that $\hat{\tau}_s^t \neq \hat{\tau}_{s+1}^t$. Then $\hat{\tau}_s^t$ has a new leaf node v labeled Y_s . This node v would be in J , contradicting that $J = \emptyset$. \square

We are now ready to prove the Witness Tree Lemma.

PROOF OF LEMMA 2.9. To simplify the notation, let us define $Q(v) := Q(L(v))$ for any node v of τ . In analyzing the PRA, there are two types of random variables: first, there is the resampling table R ; second, there are the random variables U_k . By our coupling construction, the full table R is constructed at time 0; thus, we may view the PRA as a stochastic process that sequentially generates the random variables R, U_1, U_2, \dots .

We will prove a stronger result: for all pairs of integers s, T with $1 \leq s \leq T$, and all tree-structures τ , we have

$$\Pr\left(\bigvee_{t=s}^{T-1} \hat{\tau}_s^t = \tau \mid R, U_1, \dots, U_{s-1}\right) \leq \prod_{v \in \tau} Q(v). \quad (6)$$

We may assume that $\tau \neq \emptyset$, as otherwise the right-hand side of Equation (6) equals 1 and the bound holds vacuously. We prove this by induction backward on s . The base case is $s = T$; in this case left-hand side of Equation (6) is zero so the bound holds vacuously. We move to the induction step. Suppose that we condition on R, U_1, \dots, U_{s-1} . If there are no more true bad events, then (as $\tau \neq \emptyset$) the bound holds, since the left-hand side of Equation (6) equals zero. Otherwise, let us condition as well on the violated set B_s . Define J to be the set of leaf nodes v of τ such that $L(v) \subseteq B_s$.

By Proposition 2.11, to τ to appear, it must either be the case that $J = \emptyset$, or that J contains a unique node at greatest depth.

In the first case, then by Proposition 2.11 we have $\hat{\tau}_{s+1}^t = \hat{\tau}_s^t$. So a necessary condition to have $\tau_s^t = \tau$ is to have $\tau_{s+1}^t = \tau$ for some t in the range $s + 1 \leq t < T$. By the induction hypothesis, this has probability at most $\prod_{v \in \tau} Q(v)$ (even after we condition on the random variable U_s).

In the second case, suppose that the greatest-depth node of J is v , with $L(v) = Y$. Then Proposition 2.11 shows that Y must be the resampled set at time s and $\hat{\tau}_{s+1}^t = \hat{\tau}_s^t - v$. The probability of selecting Y is at most $\frac{Q(Y)}{\sum_{Z \subseteq B} Q(Z)} \leq Q(Y) = Q(v)$.

By the induction hypothesis, the probability that there is some t with $\hat{\tau}_{s+1}^t = \tau - v$ is at most $\prod_{u \in \tau - v} Q(u)$, even after conditioning on R, U_1, \dots, U_s . Because of this conditional probability bound, we can multiply the two probabilities; so the overall probability of both selecting Y and having $\hat{\tau}_{s+1}^t = \tau - v$, is at most $Q(v) \times \prod_{u \in \tau - v} Q(u) = \prod_{u \in \tau} Q(u)$, and the induction again holds.

We now move on to prove the full result. By Proposition 2.10, the resampling table is compatible with τ with probability at most $\prod_{v \in \tau} p^{L(v)}$. We have shown that

$$\Pr\left(\bigvee_{t=1}^T \hat{\tau}^t = \tau \mid R\right) \leq \prod_{v \in \tau} Q(v).$$

So, multiplying these two probabilities gives

$$\Pr\left(\bigvee_{t=1}^T \hat{\tau}^t = \tau\right) \leq \left(\prod_{v \in \tau} Q(v)\right) \left(\prod_{v \in \tau} p^{L(v)}\right) = w(\tau).$$

Taking the limit as $T \rightarrow \infty$:

$$\Pr\left(\bigvee_{t \geq 1} \hat{\tau}^t = \tau\right) = \lim_{T \rightarrow \infty} \Pr\left(\bigvee_{t=1}^T \hat{\tau}^t = \tau\right) \leq \lim_{T \rightarrow \infty} w(\tau) = w(\tau). \quad \square$$

2.5 Finishing the Proof: Counting Witness Trees

To finish the proof of Theorem 2.6 and show the convergence of the PRA, we must count the weight of certain classes of witness trees.

Definition 2.12 (Proper Tree-Structure). A tree-structure τ is *proper* if it has the following property: for every node v with children w_1, \dots, w_s , the labels $L(w_1), \dots, L(w_s)$ are all distinct and $\{L(w_1), \dots, L(w_s)\} \in \text{NS}(L(v))$.

We let Γ denote the set of all proper tree-structures, and $\Gamma(Y)$ the set of non-empty proper tree-structures whose root node has label Y .

It is clear that if τ is a proper tree-structure, all of its subtrees are proper tree-structures as well.

PROPOSITION 2.13. *The tree-structure $\hat{\tau}^t$ is proper.*

PROOF. Consider some node $v \in \hat{\tau}^t$ with children w_1, \dots, w_s . By Proposition 2.8, we have $L(w_i) \approx L(w_j)$ for $i \neq j$. Hence, $L(w_1), \dots, L(w_s)$ are distinct. Let $\mathcal{T} = \{L(w_1), \dots, L(w_s)\}$; we have thus shown that \mathcal{T} satisfies property (2) of the definition of neighbor-set.

To show part (1), suppose that w_i is added as a child of v in forming $\hat{\tau}_k^t$. So $v \in M_k^t$ and $L(w_i)$ is the resampled set Y_j at time k . By definition of M_k^t , we have $Y_k \approx L(v)$ and so $L(w_i) \approx L(v)$.

To show part (3), suppose that $L(w_i) \bowtie L(v)$ and $L(w_j) \bowtie L(v)$, and suppose that w_i, w_j correspond to resamplings at times s, s' , respectively, where $s < s'$. But then note that v already has a child w_j in $\hat{\tau}_s^t$ with $L(w_j) \bowtie L(v)$, so that $v \notin M_s^t$. So w_i cannot be added as a child at time s , a contradiction. \square

PROPOSITION 2.14. *If the function μ satisfies Theorem 2.6(a), then every $Y \in \mathcal{A}$ has*

$$\sum_{\tau \in \Gamma(Y)} w(\tau) \leq \mu(Y).$$

PROOF. For any $Y \in \mathcal{A}$, let $T_h(Y)$ be the total weight of all $\tau \in \Gamma(Y)$ whose height is at most h . We show that $T_h(Y) \leq \mu(Y)$ for all $Y \in \mathcal{A}$ and $h \geq 0$, by induction on h . When $h = 0$, this is vacuously true. Now, consider some $\tau \in \Gamma(Y)$ of height at most h . Suppose that the children of the root node are w_1, \dots, w_s (with possibly $s = 0$), with labels Y_1, \dots, Y_s respectively.

If τ_1, \dots, τ_s are the subtrees of each w_i , then each τ_i has height at most $h - 1$ and $\tau_i \in \Gamma(Y_i)$. Also, note that $w(\tau) = p^Y Q(Y) w(\tau_1) \cdots w(\tau_s)$. So, for a fixed value of Y_1, \dots, Y_s , the total weight of all such proper tree-structures τ is (by induction hypothesis) at most $p^Y Q(Y) T_{h-1}(\tau_1) \cdots T_{h-1}(\tau_s) \leq p^Y Q(Y) \mu(Y_1) \cdots \mu(Y_s)$.

By definition, Y_1, \dots, Y_s are distinct and $\{Y_1, \dots, Y_s\} \in \text{NS}(Y)$. Summing over all such neighbor-sets, we see that

$$T_h(Y) \leq \sum_{\mathcal{T} \in \text{NS}(Y)} p^Y Q(Y) \prod_{Y' \in \mathcal{T}} \mu(Y').$$

By Theorem 2.6(a), the right-hand side of this expression is at most $\mu(Y)$. So we have shown that $T_h(Y) \leq \mu(Y)$ for all integer $h \geq 0$. This shows that $T_\infty(Y) \leq \lim_{h \rightarrow \infty} T_h(Y) \leq \lim_{h \rightarrow \infty} \mu(Y) = \mu(Y)$, completing the proof. \square

PROOF OF THEOREM 2.6. Let us first show Theorem 2.6(a). Suppose that the PRA runs for t or more time-steps. By Proposition 2.7 $\hat{\tau}^1, \dots, \hat{\tau}^t$ are distinct, non-empty, appearing tree-structures. By Proposition 2.13 they are all proper. So the number of resamplings is at most the number of appearing proper tree-structures, i.e.,

$$\begin{aligned} \mathbb{E}[\# \text{ Resamplings}] &\leq \sum_{\tau \in \Gamma} \Pr(\tau \text{ appears}) \leq \sum_{\tau \in \Gamma} w(\tau) && \text{(by Lemma 2.9)} \\ &= \sum_{Y \in \mathcal{A}} \sum_{\tau \in \Gamma(Y)} w(\tau) \leq \sum_{Y \in \mathcal{A}} \mu(Y). && \text{(by Proposition 2.14)} \end{aligned}$$

In particular, since this is finite, the algorithm terminates with probability 1.

We derive Theorem 2.6(b) by using the following method to enumerate neighbor-sets \mathcal{T} . First, we put into \mathcal{T} either one element Y'' with $Y'' \bowtie Y$, or no such elements; this contributes a factor $(1 + \sum_{Y'' \bowtie Y} \mu(Y''))$. Next, we may place any $Y' \sim Y$ into \mathcal{T} . For each $Y' \sim Y$, this contributes the term $1 + \mu(Y')$. As every neighbor-set \mathcal{T} is generated in this way, we have

$$\sum_{\mathcal{T} \in \text{NS}(Y)} \prod_{Y' \in \mathcal{T}} \mu(Y') \leq \left(\prod_{Y' \sim Y} (1 + \mu(Y')) \right) \left(1 + \sum_{Y'' \bowtie Y} \mu(Y'') \right).$$

Finally, we derive Theorem 2.6(c) by setting $\mu(Y) = ep^Y Q(Y)$ for all Y . This satisfies Theorem 2.6(b), as any $Y \in \mathcal{A}$ has

$$\left(\prod_{Y' \sim Y} (1 + \mu(Y')) \right) \left(1 + \sum_{Y'' \bowtie Y} \mu(Y'') \right) \leq e^{\sum_{Y' \sim Y} \mu(Y')} e^{\sum_{Y'' \bowtie Y} \mu(Y'')} = e^{\sum_{Z \bowtie Y} ep^Z Q(Z)} \leq e^{eDP} \leq e.$$

Therefore, we have

$$p^Y Q(Y) \left(\prod_{Y' \sim Y} (1 + \mu(Y')) \right) \left(1 + \sum_{Y'' \bowtie Y} \mu(Y'') \right) \leq ep^Y Q(Y) = \mu(Y),$$

satisfying Theorem 2.6(b). \square

3 EXTENSION TO COMPLEX BAD EVENTS

Many applications of the LLL involve multiple bad events that may be more complex than pure atomic events; for example, they may be defined by linear threshold functions. We can always write a complex bad event as a union of a (possibly large) number of atomic bad events. For example, a bad event of the form $[X_{i_1} = j_1] + \dots + [X_{i_r} = j_r] \geq t$ can be represented as $\binom{r}{t}$ separate atomic events. Thus, we suppose that there are multiple bad events $\mathcal{B}_1, \dots, \mathcal{B}_K$, where each \mathcal{B}_k is a subset of \mathcal{A} ; the sets $\mathcal{B}_1, \dots, \mathcal{B}_K$ are not necessarily disjoint.

It would be natural to apply Theorem 2.6 directly on the bad event set $\mathcal{B} = \mathcal{B}_1 \cup \dots \cup \mathcal{B}_K$. However, there are two technical obstacles to this. The first difficulty is that Theorem 2.6 requires us to bound the connection (in terms of the relation \approx) between subsets of events of \mathcal{B}_k and \mathcal{B}_ℓ for $k \neq \ell$. The linkages due to \sim are relatively easy to handle in this way, and are similar to the usual asymmetric LLL. But the linkages due to \bowtie are much more difficult to enumerate and analyze.

To avoid this first problem, we will “decouple” $\mathcal{B}_1, \dots, \mathcal{B}_K$; we analyze each \mathcal{B}_k separately, deriving an appropriate fractional hitting-set Q_k and computing an appropriate potential function for \mathcal{B}_k . We then sum this potential function over $k = 1, \dots, K$. In particular, we never need to analyze \bowtie -interactions between $\mathcal{B}_k, \mathcal{B}_\ell$.

The second technical difficulty is that Theorem 2.6 requires defining an auxiliary function μ , and checking a condition on it for every $Y \in \mathcal{A}$. Because of our method for decoupling the bad events, we would actually need to check a separate condition for every $Y \in \mathcal{A}$ as well as each $k = 1, \dots, K$. This results in a huge inflation in the number of parameters. To avoid this, we reparametrize in terms of a more compact auxiliary function, consisting of just variables $\lambda_{i,j}$ for each *element* $(i, j) \in \mathcal{X}$. We describe how to encode μ as a function of λ , and how to check a much more manageable set of conditions on it.

3.1 Decoupling Bad-events

Given fractional hitting-sets Q_1, \dots, Q_K for $\mathcal{B}_1, \dots, \mathcal{B}_K$, respectively, we will use the following Algorithm 2. This is a slight modification of the PRA given in Section 2.2.

ALGORITHM 2: PRA-multi($p, \mathcal{B}_1, \dots, \mathcal{B}_K, Q_1, \dots, Q_K, \mathcal{X}$)

- 1: Draw the values X_1, \dots, X_n independently according to the probability distribution p .
- 2: **while** there is some true bad event $B \in \mathcal{B}_1 \cup \dots \cup \mathcal{B}_K$ **do**
- 3: Arbitrarily select some index $k \in [K]$ and some $B \in \mathcal{B}_k$ such that B is true.
- 4: Select exactly one subset $Y \subseteq B$. The probability of selecting a given Y is given by

$$\Pr(\text{select } Y) = \frac{Q_k(Y)}{\sum_{Z \subseteq B} Q_k(Z)}$$

- 5: Resample every variable involved in Y independently according to probability distribution p .
 - 6: Return X
-

We analyze PRA-multi by coupling it to the original version of the PRA when run on a larger set of variables and an appropriate set of bad events $\tilde{\mathcal{B}}$. In effect, we encode $\mathcal{B}_1, \dots, \mathcal{B}_K$ so that $\tilde{\mathcal{B}}$ represents the *disjoint* union of $\mathcal{B}_1, \dots, \mathcal{B}_K$.

This new problem instance has the same n variables as the original instance, plus $K|\mathcal{A}|$ new indicator variables, which we will index as $\langle Y, k \rangle$ for $k \in [K]$ and $Y \in \mathcal{A}$. We set $F_{\langle Y, k \rangle} = \{0\}$, i.e., each new variable can only take on a single value. Thus, the new set of elements $\tilde{\mathcal{X}}$ is given by

$$\tilde{\mathcal{X}} = \mathcal{X} \cup \{(\langle Y, k \rangle, 0) \mid k \in [K], Y \in \mathcal{A}\}.$$

To simplify notation in the construction, let us define $[Y, k]$ to denote the singleton set $[Y, k] = \{(\langle Y, k \rangle, 0)\}$. We construct the new set of bad events $\tilde{\mathcal{B}}$ as:

$$\tilde{\mathcal{B}} = \left\{ B \cup \bigcup_{Y \in \mathcal{A}} [Y, k] \mid B \in \mathcal{B}_k \right\}.$$

The following definitions will be used throughout our construction. For any $Y \in \mathcal{A}$ and $k \in [K]$, let us define

$$(Y, k) = Y \cup [Y, k] = Y \cup \{(\langle Y, k \rangle, 0)\}. \quad (7)$$

Definition 3.1 (Good Set). We say that a set $Z \subseteq \tilde{\mathcal{X}}$ is *good* if it has the form $Z = (Y, k)$, where $Y \in \mathcal{A}$ and $k \in [K]$. We say that a set $\mathcal{Z} \subseteq 2^{\tilde{\mathcal{X}}}$ is good if every member $Z \in \mathcal{Z}$ is good.

Using the notation (7), we often identify the collection of good sets with the space $\mathcal{A} \times [K]$. (Note that every good set has a *unique* representation as (Y, k)).

With these definitions, we construct the corresponding fractional hitting-set \tilde{Q} for $\tilde{\mathcal{B}}$ by

$$\tilde{Q}(Z) = \begin{cases} Q_k(Y) & \text{if } Z \text{ is a good set of the form } Z = (Y, k) \\ 0 & \text{otherwise} \end{cases}.$$

We next show that the original PRA on $\tilde{\mathcal{B}}$ is equivalent to the variant PRA on \mathcal{B} .

PROPOSITION 3.2. *Consider running PRA-multi on input $p, \mathcal{B}_1, \dots, \mathcal{B}_K, Q_1, \dots, Q_K, \mathcal{X}$, as well as running the original PRA on input $\tilde{p}, \tilde{\mathcal{B}}, \tilde{Q}, \tilde{\mathcal{X}}$, where we define \tilde{p} by*

$$\tilde{p}_{ij} = \begin{cases} p_{ij} & \text{if } i \in [n] \\ 1 & \text{if } i = \langle Y, k \rangle \end{cases}.$$

With appropriate choices for the resampling rule in the two algorithms, the probability distribution on the values of X_1, \dots, X_n after t time steps is the same for the two algorithms.

PROOF. We use a coupling construction where we run both the original and variant PRA in parallel, taking a common source of randomness for the two algorithms, such that the variables X_1, \dots, X_n have the same value at each time t . For PRA-multi, there are additional variables $X_{\langle Y, k \rangle}$; but these always have the value 0. At time $t = 0$, the two algorithms obviously agree on X , since the first step of each is to sample X_1, \dots, X_n from the probability distribution p . (PRA-multi also samples, trivially, the variables $X_{\langle Y, k \rangle}$.)

Consider now some time $t > 0$. Let us first suppose that in the original PRA, all the bad events \tilde{B} are false. Since $X_{\langle Y, k \rangle} = 0$ for every $Y \in \mathcal{A}$, it must be that every bad event $B \in \mathcal{B}_k$ is false as well. So, in this case, the two algorithms both terminate and return the common vector X .

Now suppose that in the original PRA, the violated set at time t is \tilde{B} , where $\tilde{B} = B \cup \bigcup_{Y \in \mathcal{A}} [Y, k]$ and $B \in \mathcal{B}_k$. This implies that B holds on X_1, \dots, X_n . PRA-multi may correspondingly use the resampling rule of selecting $k, B \in \mathcal{B}_k$ in step (3).

In step (4) of the original PRA, we select some $\tilde{Y} \subseteq \tilde{B}$ with probability proportional to $\tilde{Q}(\tilde{Y})$. Since \tilde{Q} is only supported on good sets, we must have $\tilde{Y} = (Y, k')$ for some $k' \in [K]$, $Y \in \mathcal{A}$. Since $Y, B \in \mathcal{A}$ we must have $Y \subseteq B$. Furthermore, since the only entries of \tilde{B} are B and some sets $[Y', k]$, we must have $k = k'$. Thus, $\tilde{Y} = (Y, k)$ for some $Y \subseteq B$. Then $\tilde{Q}(\tilde{Y}) = Q_k(Y)$. So step (4) of the original PRA can be viewed as selecting a set (Y, k) where $Y \subseteq B$ in which a given Y is chosen with probability proportional to $Q_k(Y)$.

Finally, in step (5) of the original PRA, we resample X_i for every $i \sim Y$ (as well as the variable $X_{\langle Y, k \rangle}$); this is precisely what is done in step (5) of PRA-multi. So, if we use the same random bits for the two resamplings, then the values of X agree at time $t + 1$ as well. \square

In light of Proposition 3.2, we need to satisfy Theorem 2.6 for the new problem instance. The following results translate the notations between the sets $\mathcal{B}_1, \dots, \mathcal{B}_K$ and their encoding into $\tilde{\mathcal{B}}$. These proofs are omitted.

Definition 3.3 (Symmetric Relation \bowtie_k). Given $Y, Y' \in \mathcal{A}$, we say that $Y \bowtie_k Y'$ if $Y \approx Y'$ and there is some $B \in \mathcal{B}_k$ with $Y, Y' \subseteq B$

PROPOSITION 3.4. For pairs $(Y, k), (Y', k')$ where $Y, Y' \in \mathcal{A}$ and $k, k' \in [K]$ the following hold:

- (1) $(Y, k) \sim (Y', k')$ iff $Y \sim Y'$.
- (2) $(Y, k) \bowtie (Y', k')$ iff $k = k'$ and $Y \bowtie_k Y'$.

We say that (Y, k) is *supported* if $Q_k(Y) > 0$. We say that \bowtie is *null* if $Y \not\bowtie_k Y'$ for all $k \in [K]$, and all sets Y, Y' such that $Q_k(Y) > 0, Q_k(Y') > 0$.

PROPOSITION 3.5. For any $Y \in \mathcal{A}$ and $k \in [K]$, the set $\mathcal{T} \subseteq \mathcal{A} \times [K]$ is a good neighbor-set of (Y, k) (and we write $\mathcal{T} \in \text{GNS}(Y, k)$) if the following conditions hold:

- (1) Every $(Z, \ell) \in \mathcal{T}$ has either (i) $Z \sim Y$ or (ii) $\ell = k$ and $Z \bowtie_k Y$.
- (2) There do not exist distinct $(Z, k), (Z', k') \in \mathcal{T}$ with $Z \sim Z'$.
- (3) There do not exist distinct $(Z, k), (Z', k) \in \mathcal{T}$ with $Z \bowtie_k Y, Z' \bowtie_k Y$.

We can now state our main theorem, translated into the new encoding:

THEOREM 3.6 (MAIN THEOREM FOR MULTIPLE EVENTS). Let Q_1, \dots, Q_K be fractional hitting-sets for $\mathcal{B}_1, \dots, \mathcal{B}_K$, respectively. In each of the following three cases, PRA-multi terminates in a configuration avoiding all bad events with probability 1.

- (a) Suppose that $\mu : \mathcal{A} \times [K] \rightarrow [0, \infty)$ satisfies, for all $Y \in \mathcal{A}$ and all $k \in [K]$,

$$\mu(Y, k) \geq p^Y Q_k(Y) \sum_{\mathcal{T} \in \text{GNS}(Y, k)} \prod_{(Y', k') \in \mathcal{T}} \mu(Y', k').$$

Then, the expected number of resamplings is at most $\sum_{(Y, k)} \mu(Y, k)$.

- (b) Suppose that $\mu : \mathcal{A} \times [K] \rightarrow [0, \infty)$ satisfies, for all $Y \in \mathcal{A}$ and all $k \in [K]$,

$$\mu(Y, k) \geq p^Y Q_k(Y) \left(\prod_{Y' \sim Y} \left(1 + \sum_{k' \in [K]} \mu(Y', k') \right) \right) \left(1 + \sum_{Y'' \bowtie_k Y} \mu(Y'', k) \right).$$

Then, the expected number of resamplings is at most $\sum_{(Y, k)} \mu(Y, k)$.

- (c) Suppose that $p^Y Q_k(Y) \leq P$ for all $Y \in \mathcal{A}, k \in [K]$; and suppose that for all supported pairs (Y, k) there are at most D supported pairs $(Y', k') \approx (Y, k)$. And suppose, finally, that $ePD \leq 1$. Then, the expected number of resamplings is at most $e \sum_{(Y, k)} p^Y Q_k(Y)$.

PROOF. As in Theorem 2.6, it suffices to prove case (a). We first claim that \tilde{Q} is a fractional hitting-set for $\tilde{\mathcal{B}}$. For, consider some $\tilde{B} \in \tilde{\mathcal{B}}$ of the form $\tilde{B} = B \cup \bigcup_{Y \in \mathcal{A}} [Y, k]$ for $B \in \mathcal{B}_k$. Then

$$\sum_{Y \subseteq \tilde{B}} \tilde{Q}(Y) \geq \sum_{Y \subseteq B} \tilde{Q}(Y \cup [Y, k]) = \sum_{Y \subseteq B} Q_k(Y) \geq 1.$$

Next, to apply Theorem 2.6(a), define the function $\tilde{\mu} : \tilde{\mathcal{A}} \rightarrow [0, \infty)$ by

$$\tilde{\mu}(Z) = \begin{cases} \mu(Z) & \text{if } Z \text{ is good} \\ 0 & \text{otherwise} \end{cases}.$$

We want to show that every $Z \in \tilde{\mathcal{A}}$ has

$$\tilde{\mu}(Z) \geq \tilde{p}^Z \tilde{Q}(Z) \sum_{\mathcal{T} \in \text{NS}(Z)} \prod_{Y' \in \mathcal{T}} \tilde{\mu}(Y'). \quad (8)$$

If $\mathcal{T} \in \text{NS}(Z)$ is not good, then $\prod_{Y' \in \mathcal{T}} \tilde{\mu}(Y') = 0$. If Z is not good, then Equation (8) holds easily, as $\tilde{Q}(Z) = 0$. So suppose $Z = (Y, k)$, and so Equation (8) reduces to

$$\mu(Y, k) \geq p^Y Q_k(Y) \sum_{\mathcal{T} \in \text{GNS}(Z)} \prod_{(Y', k') \in \mathcal{T}} \mu(Y', k'),$$

which holds by hypothesis.

So the expected number of resamplings is $\sum_{Z \in \tilde{\mathcal{A}}} \tilde{\mu}(Z) = \sum_{Y \in \mathcal{A}} \sum_{k \in [K]} \mu(Y, k)$. Since this is finite, the PRA terminates with probability 1. \square

We note that Theorem 3.6 is equivalent to Theorem 2.6 if $K = 1$. (Each time a set $Y \in \mathcal{A}$ is referenced, simply replace it with $(Y, 1)$.)

3.2 Parametrizing by λ

We now describe a criterion for PRA or PRA-multi in terms of a vector $\lambda \in [0, \infty)^X$, instead of the function $\mu : \mathcal{A} \times [K] \rightarrow \mathbf{R}$. This is a huge savings in terms of the number of parameters. In fact, the vector λ not only encodes μ , but also the probability vector p . The vector λ should be thought of as an “inflated” version of p ; roughly speaking, $\lambda_{i,j}$ is the probability that $X_i = j$ at some point during the execution of the PRA.

Given the parameter λ , we define a few related quantities that will be needed to state our theorem.

Definition 3.7 (Values λ_i, G_i, S that Depend on a Function Q and a Vector λ). Let $\mathcal{B} \subseteq \mathcal{A}$, let $\lambda \in [0, \infty)^X$ and let Q be a fractional hitting-set for \mathcal{B} .

Recalling the notation (5), define

$$S(\mathcal{B}, Q, \lambda) = \max_{Q(Y) > 0} \sum_{Z: Z \bowtie Y} Q(Z) \lambda^Z,$$

where the definition of \bowtie is with respect to \mathcal{B}, Q .

Also, for each $i \in [n]$, define

$$\lambda_i = \sum_j \lambda_{i,j}, \quad G_i(Q, \lambda) = \sum_{Y \sim i} Q(Y) \lambda^Y.$$

In the context of multiple bad events $\mathcal{B}_1, \dots, \mathcal{B}_K$, we often write $S_k = S(\mathcal{B}_k, Q_k, \lambda)$ for simplicity.

We next state our main theorem in terms of λ . The correspondence between the λ notation and the parameters p, μ will be given by the following formulas:

$$p_{ij} = \frac{\lambda_{ij}}{\lambda_i} \quad \mu(Y, k) = \frac{\lambda^Y Q_k(Y)}{1 - S(\mathcal{B}_k, Q_k, \lambda)}.$$

THEOREM 3.8 (MAIN THEOREM IN TERMS OF λ). *Let Q_1, \dots, Q_K be fractional hitting-sets for $\mathcal{B}_1, \dots, \mathcal{B}_K$, respectively, and let $\lambda \in [0, \infty)^X$. If $S(\mathcal{B}_k, Q_k, \lambda) < 1$ for all $k \in [K]$, and for all $i \in [n]$ we have*

$$\lambda_i \geq 1 + \sum_k \frac{G_i(Q_k, \lambda)}{1 - S(\mathcal{B}_k, Q_k, \lambda)},$$

then PRA terminates with probability 1; the expected number of resamplings is at most $\sum_i (\lambda_i - 1)$.

Before we prove Theorem 3.8, we record some preliminary calculations that occur in a number of places:

PROPOSITION 3.9. Assuming that the conditions of Theorem 3.8 are satisfied:

- (1) For any $i \in [n]$, we have $\sum_{Y \sim i} \sum_k \mu(Y, k) \leq \lambda_i - 1$.
- (2) For any supported (Y, k) , we have $\sum_{Z \bowtie_k Y} \mu(Z, k) \leq \frac{S_k}{1 - S_k}$.

PROOF. For the first result:

$$\sum_{Y \sim i} \sum_k \mu(Y, k) = \sum_{Y \sim i, k} \frac{\lambda^Y Q_k(Y)}{1 - S_k} = \sum_k \frac{G_i(Q_k, \lambda)}{1 - S_k} \leq \lambda_i - 1.$$

For the second result:

$$\sum_{Z \bowtie_k Y} \mu(Z, k) = \sum_{Z \bowtie_k Y} \frac{\lambda^Z Q_k(Z)}{1 - S_k} \leq \frac{S_k}{1 - S_k}. \quad \square$$

PROOF OF THEOREM 3.8. We want to show that $\mu(Y, k)$ satisfies Theorem 3.6(a) for a given $Y \in \mathcal{A}, k \in [K]$, that is,

$$\mu(Y, k) \geq p^Y Q_k(Y) \sum_{\mathcal{T} \in \text{GNS}(Y, k)} \prod_{(Y', k') \in \mathcal{T}} \mu(Y', k'). \quad (9)$$

We now bound the right-hand side of Equation (9). First, \mathcal{T} may contain at most one pair (Z, k) with $Z \bowtie_k Y$. By Proposition 3.9, the total contribution of such terms is at most $1 + \frac{S_k}{1 - S_k} = \frac{1}{1 - S_k}$. Next, for each $(i, j) \in Y$, the set \mathcal{T} may contain at most one pair (Y', k') with $i \sim Y'$. (If there two such pairs $(Y', k'), (Y'', k'')$, then $(Y', k) \sim (Y'', k'')$.) For any fixed (i, j) in Y' , this contributes at most $1 + \sum_{Y' \sim i} \sum_{k'} \mu(Y', k')$; by Proposition 3.9 this is at most $1 + (\lambda_i - 1) = \lambda_i$.

Putting these two estimates together, we estimate the right-hand side of Equation (9):

$$\begin{aligned} p^Y Q_k(Y) \sum_{\mathcal{T} \in \text{GNS}(Y, k)} \prod_{(Y', k') \in \mathcal{T}} \mu(Y', k') &\leq p^Y Q_k(Y) \times \frac{1}{1 - S_k} \times \prod_{(i, j) \in Y} \lambda_i \\ &= \frac{Q_k(Y) \prod_{(i, j) \in Y} p_{ij} \lambda_i}{1 - S_k} = \frac{Q_k(Y) \prod_{(i, j) \in Y} \frac{\lambda_{ij}}{\lambda_i} \lambda_i}{1 - S_k} = \frac{Q_k(Y) \lambda^Y}{1 - S_k} = \mu(Y, k). \end{aligned}$$

Thus, Theorem 3.6(a) holds. Note that since \mathcal{A} contains only non-empty sets, every $Y \in \mathcal{A}$ has $Y \sim i$ for at least one value $i \in [n]$. Therefore, using Proposition 3.9, we compute

$$\mathbf{E}[\# \text{ resamplings}] \leq \sum_{Y, k} \mu(Y, k) \leq \sum_{i \in [n]} \sum_{Y \sim i} \sum_k \mu(Y, k) \leq \sum_{i \in [n]} (\lambda_i - 1). \quad \square$$

When applying Theorem 3.8, we note that if \bowtie_k is null, then $S_k = 0$. Also, we frequently use the crude bound:

$$S(\mathcal{B}, Q, \lambda) \leq \sum_{Y \in \mathcal{A}} Q(Y) \lambda^Y. \quad (10)$$

The parametrization by λ can be useful for the standard MT algorithm (as the MT algorithm is a special case of the PRA). In particular, it gives clean formulas for the ‘‘MT distribution,’’ which is the distribution on the variables at the termination of the MT algorithm. See Appendix B for further discussion and some examples.

4 BAD-EVENTS DEFINED BY SUMS OF RANDOM VARIABLES

In this section, we explore a connection between symmetric polynomials and Chernoff tail bounds for sums of indicator variables of elements (i.e., terms of the form $[X_i = j]$). These will be central to assignment-packing problems. When the bad events are defined in terms of such sums, then these symmetric polynomials correspond in a natural way with fractional hitting-sets.

To state these results in their clearest form, we define the Chernoff upper-tail separation function:

Definition 4.1 (Chernoff Upper-Tail Separation Function $Ch(\mu, t)$). For real numbers $\mu, t > 0$, we define

$$Ch(\mu, t) = e^{t-\mu} (\mu/t)^t.$$

We recall three useful results of Referencee [33] on multivariate symmetric polynomials and Chernoff bounds.

PROPOSITION 4.2. For any real numbers $a_1, \dots, a_\ell \in [0, 1]$ and integer $k \leq a_1 + \dots + a_\ell$, we have

$$\sum_{\substack{X \subseteq [\ell] \\ |X|=k}} \prod_{x \in X} a_x \geq \binom{a_1 + \dots + a_\ell}{k}.$$

PROPOSITION 4.3. For any real numbers $a_1, \dots, a_\ell \in [0, \infty)$ and integer $k \geq 0$, we have

$$\sum_{\substack{X \subseteq [\ell] \\ |X|=k}} \prod_{x \in X} a_x \leq \frac{(a_1 + \dots + a_\ell)^k}{k!}.$$

PROPOSITION 4.4. For $0 \leq \mu \leq t$ and $d = \lceil t - \mu \rceil$, we have $\frac{\mu^d}{d! \binom{t}{d}} \leq Ch(\mu, t)$.

We can now state our main result, which transforms a bad event defined by sums of random variables into a fractional hitting set.

THEOREM 4.5. Let $\lambda \in [0, \infty)^X$ and $a \in [0, 1]^X$ be two vectors of real numbers, indexed by the elements $x = (i, j)$ of X . Define $\mu = \sum_x a_x \lambda_x$, and for each $i \in [n]$ define $\mu_i = \sum_j a_{i,j} \lambda_{i,j}$. For a real number $t \geq 0$, let \mathcal{B} be the complex bad event defined by

$$\mathcal{B} \equiv \sum_{i,j} a_{i,j} [X_i = j] \geq t.$$

Let d be any integer in the range $1 \leq d \leq t$. Then, recalling Definition 3.7, there is a fractional hitting-set Q with the property

$$S(\mathcal{B}, Q, \lambda) \leq \frac{\mu^d}{d! \binom{t}{d}}; \quad G_i(Q, \lambda) \leq \frac{d\mu_i}{\mu} \cdot \frac{\mu^d}{d! \binom{t}{d}}.$$

PROOF. We define Q by

$$Q(Y) = \begin{cases} \frac{\prod_{x \in Y} a_x}{\binom{t}{d}} & \text{if } |Y| = d \\ 0 & \text{otherwise} \end{cases},$$

To show that this is a valid fractional hitting-set, consider some atomic bad event B , where $\sum_{x \in B} a_x \geq t$. By Proposition 4.2, we have

$$\sum_{Y \subseteq B} Q(Y) = \frac{\sum_{Y \subseteq B, |Y|=d} a^Y}{\binom{t}{d}} \geq \frac{\binom{\sum_{x \in B} a_x}{d}}{\binom{t}{d}} \geq \frac{\binom{t}{d}}{\binom{t}{d}} = 1.$$

We use Proposition 4.3 and Equation (10) to compute $S(\mathcal{B}, Q, \lambda)$ as:

$$S(\mathcal{B}, Q, \lambda) \leq \sum_{Y \in \mathcal{A}} \lambda^Y Q(Y) = \frac{\sum_{Y \in \mathcal{A}, |Y|=d} \lambda^Y a^Y}{\binom{t}{d}} \leq \frac{(\sum_{x \in Y} a_x \lambda_x)^d}{d! \binom{t}{d}} = \frac{\mu^d}{d! \binom{t}{d}}.$$

Similarly, we use Proposition 4.3 to compute $G_i(Q, \lambda)$ as:

$$\begin{aligned} G_i(Q, \lambda) &= \sum_{Y \sim i} \lambda^Y Q(Y) = \frac{\sum_j a_{i,j} \lambda_{i,j}}{\binom{t}{d}} \sum_{\substack{Y \in \mathcal{A} \\ Y \sim i \\ |Y|=d-1}} \lambda^Y \leq \frac{\sum_j a_{i,j} \lambda_{i,j} \left(\sum_{\substack{(u,j) \in \mathcal{X} \\ u \neq i}} a_{(u,j)} \lambda_{(u,j)} \right)^{d-1}}{(d-1)! \binom{t}{d}} \\ &= \frac{\mu_i (\mu - \mu_i)^{d-1}}{(d-1)! \binom{t}{d}} \leq \frac{d \mu_i}{\mu} \frac{\mu^d}{d! \binom{t}{d}}. \quad \square \end{aligned}$$

We note that to use the fractional hitting-set Q of Theorem 4.5 in the PRA, we must be able to efficiently access Q . We discuss how to do this in Appendix C.

COROLLARY 4.6. *Let $\lambda \in [0, \infty)^X$ and $a \in [0, 1]^X$ be two vectors of real numbers, indexed by the elements $x = (i, j)$ of \mathcal{X} . Define $\mu = \sum_x a_x \lambda_x$, and for each $i \in [n]$ define $\mu_i = \sum_j a_{i,j} \lambda_{i,j}$. For a real number $t \geq 1$, let \mathcal{B} be the complex bad event defined by*

$$\mathcal{B} \equiv \sum_{i,j} a_{i,j} [X_i = j] \geq t.$$

Let r be any real number satisfying $1 \leq r < t$ and $r \geq \mu$. Then there is a fractional hitting set Q with the property

$$S(\mathcal{B}, Q, \lambda) \leq \text{Ch}(r, t); \quad G_i(Q, \lambda) \leq \mu_i \left(\frac{t+1}{r} - 1 \right) \text{Ch}(r, t);$$

PROOF. Apply Theorem 4.5 with parameter t and $d = \lceil t - r \rceil$. Note that since $r < t$ we have $d \geq 1$, and since $r \geq 1$ we have $d \leq \lceil t - 1 \rceil \leq t$. With this value of d , Proposition 4.4 gives:

$$S(\mathcal{B}, Q, \lambda) \leq \frac{\mu^d}{d! \binom{t}{d}} \leq \frac{r^d}{d! \binom{t}{d}} \leq \text{Ch}(r, t)$$

and similarly

$$G_i(Q, \lambda) \leq \frac{d \mu_i}{\mu} \cdot \frac{\mu^d}{d! \binom{t}{d}} \leq \frac{d \mu_i}{r} \cdot \frac{r^d}{d! \binom{t}{d}} \leq \frac{(t-r+1) \mu_i}{r} \cdot \frac{r^d}{d! \binom{t}{d}} \leq \mu \left(\frac{t+1}{r} - 1 \right) \text{Ch}(r, t). \quad \square$$

4.1 Column-sparse Assignment-packing Problems

We now consider a family of constraint satisfaction problems we refer to as *assignment-packing problems*. Here, we have m linear packing constraints of the form “ $\sum_{i,j} a_{k,i,j} [X_i = j] \leq b_k$,” where $a_{k,x} \in [0, 1]$, $b_k \geq 0$. We also assume that this CSP is “column-sparse,” in the sense that for any $x \in \mathcal{X}$ we have $\sum_k a_{k,x} \leq D$ for some parameter D . (As above, x will often refer to some element $(i, j) \in \mathcal{X}$.) When does such an integer linear program have a feasible solution?

One technique to solve this uses an LP relaxation, where each $(i, j) \in \mathcal{X}$ has a fractional variable $z_{i,j} \in [0, 1]$ to represent $[X_i = j]$. Each $i \in [n]$ has a linear constraint $\sum_{j \in F_i} z_{i,j} = 1$. In addition the packing constraints are tightened to $\sum_x a_{k,x} z_x \leq c_k$ for some $c_k \leq b_k$. We then convert the fractional solution z into an integral solution via some form of randomized rounding. The central problem becomes how close c_k can be to b_k , to ensure that a feasible integral solution can be converted into a feasible integral solution.

We describe here a randomized rounding algorithm based on a single application of the PRA. This involves a parameter $\epsilon > 0$, which determines a multiplicative increase in b_k as compared to c_k , in addition to a secondary discrepancy term. Our main result will be the following:

THEOREM 4.7. *Let us suppose that the linear program*

$$\sum_{j \in F_i} z_{i,j} = 1, \quad \sum_x a_{k,x} z_x \leq c_k, \quad z_x \in [0, 1]$$

is satisfiable, for some vector $c \in [1, \infty)^m$, and that we have a separation oracle for it (that is, given a variable assignment, we can either find a violated linear constraint, or determine that all constraints are satisfied.)

Let $D \geq 2$ be some parameter satisfying

$$D \geq \max_x \sum_k a_{k,x}$$

and let ϵ be a parameter satisfying $0 < \epsilon \leq 1/D$.

Then in expected time $\text{poly}(n)$, we can find a solution to the integer program

$$X_i \in F_i \quad \sum_{i,j} a_{k,i,j} [X_i = j] \leq b_k,$$

where we define the vector b by

$$b_k = \begin{cases} \frac{100 \ln(1/\epsilon)}{1 + \ln(\frac{\ln(1/\epsilon)}{c_k})} & \text{for } c_k \leq \ln(1/\epsilon) \\ c_k(1 + \epsilon) + 10 \sqrt{c_k \ln\left(D + \frac{1}{c_k \epsilon^2}\right)} & \text{for } c_k > \ln(1/\epsilon) \end{cases},$$

PROOF. The full proof requires technical analysis of the Chernoff tail-separation function. To separate this routine functional analysis from our analysis of the PRA, we defer some calculations to Appendix D, in which we show that the vector b satisfies the following conditions for all $k \in [m]$:

$$\begin{aligned} \text{(C1)} \quad & \left(\frac{b_k + 1}{c_k(1 + \epsilon)} - 1\right) \text{Ch}(c_k(1 + \epsilon), b_k) \leq \frac{\epsilon}{4D} \\ \text{(C2)} \quad & \text{Ch}(c_k(1 + \epsilon), b_k) \leq 1/2 \end{aligned}$$

To begin, the separation oracle allows us to solve the LP in polynomial time; let \hat{z} be the resulting fractional solution. We use the framework of Section 3, in which each packing constraint will correspond to a complex bad event (that is, $K = m$), and where $\lambda = (1 + \epsilon)\hat{z}$.

Let us analyze a constraint k , corresponding to the complex bad event $\mathcal{B}_k \equiv \sum_{i,j} a_{k,i,j} [X_i = j] \geq b_k$. The fractional hitting-set Q_k will be the one of Corollary 4.6, with $t = b_k$ and $r = (1 + \epsilon)c_k$. Note that $r \geq 1$, since $c_k \geq 1$, and $r < t$ by property (C2). To show $r \geq \mu$ using the terminology of Corollary 4.6, we compute

$$\mu = \sum_{i,j \in F_i} a_{k,i,j} \lambda_{i,j} = \sum_{i,j \in F_i} a_{k,i,j} (1 + \epsilon) \hat{z}_{i,j} \leq (1 + \epsilon) c_k = r.$$

We next calculate G_i and S_k . By Corollary 4.6 and using property (C2),

$$S_k = S(\mathcal{B}_k, Q_k, \lambda) \leq \text{Ch}(r, t) = \text{Ch}(c_k(1 + \epsilon), b_k) \leq 1/2.$$

Furthermore, for each $i \in [n]$, property (C1) gives

$$G_i(Q_k, \lambda) \leq \mu_i \left(\frac{t + 1}{r} + 1\right) \text{Ch}(r, t) = \mu_i \left(\frac{b_k + 1}{c_k(1 + \epsilon)} - 1\right) \text{Ch}(c_k(1 + \epsilon), b_k) \leq \mu_i \frac{\epsilon}{4D}.$$

We compute μ_i by:

$$\mu_i = \sum_{j \in F_i} a_{k,i,j} \lambda_{k,i,j} \leq (1 + \epsilon) \sum_j a_{k,i,j} \hat{z}_{i,j}.$$

To apply Theorem 3.8, we sum over all $j \in F_i$ obtaining

$$\begin{aligned}
\sum_{j \in F_i} \lambda_{i,j} - \sum_k \frac{G_i(Q_k, \lambda)}{1 - S_k} &\geq (1 + \epsilon) - \sum_k \frac{\epsilon/(4D) \times (1 + \epsilon) \sum_j a_{k,i,j} \hat{z}_{i,j}}{1/2} \\
&= (1 + \epsilon) - \frac{1}{2} \sum_{j \in F_i} \hat{z}_{i,j} (1 + \epsilon) \times (\epsilon/D) \times \sum_k a_{k,i,j} \\
&\geq (1 + \epsilon) - \frac{1}{2} \sum_{j \in F_i} \hat{z}_{i,j} (1 + \epsilon) \times (\epsilon/D) \times D \quad \text{definition of } D \\
&= (1 + \epsilon) - \frac{1}{2} (1 + \epsilon) \epsilon \geq 1 \quad \text{as } \epsilon \in [0, 1].
\end{aligned}$$

Furthermore, for any $i \in [n]$ we have $\sum_{j \in F_i} \lambda_{i,j} \leq 1 + \epsilon \leq 2$, so the expected number of iterations before the PRA terminates is at most $\sum_{i,j} \lambda_{i,j} \leq O(n)$. Each step of the PRA can be efficiently implemented using the separation oracle, so this gives a polynomial-time algorithm (even though the number of constraints m may be exponential in n). \square

We can simplify Theorem 4.7 when we have a uniform bound on the right-hand-side values c_k .

PROPOSITION 4.8. *Let us suppose that the linear program*

$$\sum_{j \in F_i} z_{i,j} = 1, \quad \sum_x a_{k,x} z_x \leq c_k, \quad z_x \in [0, 1].$$

is satisfiable, for some vector $c \in [1, R]^m$, and that we have a separation oracle for it.

Let $D \geq 2$ be some parameter satisfying $D \geq \max_x \sum_{k,x} a_{k,x}$.

Then in expected time $\text{poly}(n)$, we can find a solution to the integer program

$$X_i \in F_i \quad \sum_{i,j} a_{k,i,j} [X_i = j] \leq b_k,$$

where we define the vector b by

$$b_k = \begin{cases} \frac{C \ln D}{1 + \ln(\frac{\ln D}{c_k})} & \text{if } R \leq \ln D \\ c_k + C\sqrt{R \ln D} & \text{if } R > \ln D \end{cases},$$

for some universal constant C .

PROOF. Let $d = \ln D$. If $R \leq d$, then this follows immediately from Theorem 4.7 with $\epsilon = 1/D$. Otherwise, for $R > d$, we apply Theorem 4.7 with $\epsilon = D^{-10} \sqrt{d/R}$; note that $\epsilon \leq 1/D$ in this case. Let $\delta = \ln(1/\epsilon) = 10d + \frac{1}{2} \ln(R/d)$. If $c_k \leq \delta$, then

$$b_k = \frac{100\delta}{1 + \ln(\delta/R)} \leq \frac{O(d + \ln(R/d))}{1 + \ln(\delta/R)} \leq O(d + \ln(R/d)) \leq O(d + \log R) \leq O\left(\sqrt{R \log D}\right).$$

If $c_k \geq \delta$, then we have

$$b_k = c_k(1 + \epsilon) + 10\sqrt{R \ln\left(D + \frac{1}{c_k \epsilon^2}\right)} = c_k + \left(c_k D^{-10} \sqrt{d/R}\right) + 10\sqrt{c_k \ln\left(D + \frac{D^{20}R}{c_k d}\right)}.$$

As $c_k \leq R$, the term $c_k D^{-10} \sqrt{d/R}$ is at most $RD^{-10} \sqrt{\frac{\log D}{R}} \leq O(\sqrt{R})$. Likewise, simple analysis shows that $c_k \ln\left(D + \frac{D^{20}R}{c_k d}\right) \leq O(R \log D)$, giving the claimed result. \square

We note that there is a fundamental problem in how the standard LLL deals with fractional entries in the constraint matrix A . The issue is that variable X_i affects constraint k if $a_{k,i,j} > 0$, and it is possible that every variable affects every constraint. In the LLL setting, one cannot quantify

“how much” X_i affects a constraint. As shown in Reference [17], it is possible to sidestep this issue by quantization of A : if $a_{k,i,j}$ is close to zero, then it gets quantized to zero and thus X_i does not affect the given constraint at all. However, this is cumbersome and unnatural.

The LLL has other problems in dealing with CSP’s, even when the matrix entries are in the range $\{0, 1\}$. In Appendix A, we compare the PRA to the MT algorithm, with a rather extreme demonstration of how the PRA is able to make steady progress to the solution, whereas the MT algorithm on its own is completely unable to do so. Specifically, we construct a family of linear threshold constraint-satisfaction problems, in which every entry of the matrix A is in the range $\{0, 1\}$, but where still *every* variable effects *every* constraint. In this case, there is no “locality” in the sense of the LLL. The MT algorithm completely throws away its partial solution at every iteration, and starts from scratch. Not surprisingly, the MT algorithm cannot guarantee *any* scale-free approximation factors (independent of the number of constraints or variables) for this type of problem. By contrast, the PRA yields a very good approximation in expected polynomial time.

We also note that there is a related class of integer programming problems referred as *column-sparse covering integer problems*. A variant of the PRA can be used to solve these systems; see Reference [7] for more details.

5 APPLICATIONS OF COLUMN-SPARSE ASSIGNMENT PROBLEMS

In this section, we discuss two straightforward applications of Theorem 4.7, to a multi-dimensional scheduling approximation algorithm and to a problem in discrepancy. The results we obtain have not been stated before explicitly; however, it is possible to derive them using a combination of previous rounding algorithms including the LLL, polyhedral rounding such as Reference [23], and the algorithm of Reference [24]. We include them here to better explain the PRA, and to demonstrate how it gives a simpler and more unified approach to such discrepancy bounds.

5.1 Multi-dimensional Scheduling

Scheduling on unrelated parallel machines is a classic problem in operations research [26]. In this setting, we have n jobs and m machines, and each job i needs to be scheduled on some machine.² If job i is scheduled on machine j , then j incurs a load of $p_{i,j}$. The goal is to minimize the *makespan*, the maximum total load on any machine. The standard way to approach this is to introduce an auxiliary parameter T , and ask if we can schedule with makespan T [26, 34]. This leads to the integer program formulation:

$$\begin{aligned} \forall i \in [n], \sum_{j \in F_i} z_{i,j} &= 1; \\ \forall j \in [m], \sum_i p_{i,j} z_{i,j} &\leq T; \\ \forall (i, j), p_{i,j} > T &\Rightarrow z_{i,j} = 0. \end{aligned}$$

Azar and Epstein [4] considered a generalization where there are d dimensions to the load (say runtime, energy, heat consumption, etc.) when job i gets assigned to machine j . In dimension ℓ , this assignment leads to a load of $p_{i,j,\ell}$ on j . We ask here: *Given a vector (T_1, T_2, \dots, T_d) , is there an assignment that has a makespan of at most T_ℓ in each dimension ℓ ?* Azar and Epstein described a $(d + 1)$ -approximation algorithm for this. We show how the PRA can improve this to $O(\frac{\log d}{\log \log d})$:

²We interchange the standard use of the indices i and j here to conform to the rest of our notation.

THEOREM 5.1. *Given a feasible makespan vector (T_1, \dots, T_d) , there is a randomized polynomial-time algorithm to find a schedule with makespan vector (T'_1, \dots, T'_d) , where*

$$T'_i \leq O\left(T_i \times \frac{\log d}{\log \log d}\right).$$

PROOF. This is an easy application of Proposition 4.8. First, we set $z_{i,j} := 0$ if there exists some ℓ with $p_{i,j,\ell} > T_\ell$. After this filtering, we solve the LP relaxation with the constraints:

$$\forall j \in [m], \forall \ell \in [d], \sum_i \frac{p_{i,j,\ell}}{T_\ell} z_{i,j} \leq 1.$$

This LP has its right-hand-side values (i.e. entries of c_k) equal to 1, and our filtering ensures that the coefficient matrix has entries $a_{i,(j,\ell)} = \frac{p_{i,j,\ell}}{T_\ell} \in [0, 1]$. Each element $(i, j) \in X$ has d constraints, so the maximum column sum is $D = d$. By Proposition 4.8, we get $b_k = O(\frac{\log D}{\log \log D})$. \square

5.2 Discrepancy

As another application, we consider a discrepancy problem introduced in Reference [17] and analyzed there via the LLL.

THEOREM 5.2. *Let Y be an $m \times n$ matrix whose entries are real numbers in the range $[-1, 1]$, which satisfies the following bounds on the ℓ_1 norms of the rows and columns:*

$$\forall i \sum_k |Y_{k,i}| \leq D \quad \forall k \sum_i |Y_{k,i}| \leq R,$$

for parameters $R \geq 1, D \geq 2$. Then there is a randomized polynomial-time algorithm to find a vector $v \in \{-1, +1\}^n$ such that, for all $k \in [m]$

$$|Y_k \cdot v| \leq O\left(\sqrt{R \log D}\right).$$

PROOF. Let $d = \ln D$ and for each $k \in [m]$ let $y_k = \sum_i |Y_{k,i}| \leq R$. If $R \leq d$, then the stated result holds trivially, as $|Y_k \cdot v| \leq y_k \leq R \leq \sqrt{Rd}$ for any such vector v . So, let us assume $R > d$.

For each $i \in [n]$, we introduce a variable X_i that takes two possible values that we name $+1, -1$. For each $k \in [m]$, we introduce a packing constraint

$$\sum_{i: Y_{k,i} > 0} Y_{k,i} z_{i,+1} + \sum_{i: Y_{k,i} < 0} (-Y_{k,i}) z_{i,-1} \leq c_k,$$

where $c_k = y_k/2$. This LP has a fractional solution defined by $z_{i,+1} = z_{i,-1} = 1/2$ for all i , and has maximum ℓ_1 -column-norm of D . We now apply Theorem 4.8 with parameter R to obtain an integral solution $v_1, \dots, v_n \in \{-1, 1\}^n$, such that

$$\sum_{i: Y_{k,i} > 0} Y_{k,i} [v_i = +1] + \sum_{i: Y_{k,i} < 0} (-Y_{k,i}) [v_i = -1] \leq b_k;$$

here, we observe that our assumption $R > d$ ensures that $b_k \leq c_k + O(\sqrt{Rd}) = y_k/2 + O(\sqrt{Rd})$.

This vector v achieves the desired result, as for each constraint $k \in [m]$, we have

$$\begin{aligned} Y_k \cdot v &= \sum_{i:Y_{k,i}>0} (-Y_{k,i} + 2Y_{k,i}[v_i = +1]) + \sum_{i:Y_{k,i}<0} (Y_{k,i} - 2Y_{k,i}[v_i = -1]) \\ &= -y_k + 2 \left(\sum_{i:Y_{k,i}>0} Y_{k,i}[v_i = +1] + \sum_{i:Y_{k,i}<0} -Y_{k,i}[v_i = -1] \right) \\ &\leq -y_k + 2 \left(y_k/2 + O(\sqrt{Rd}) \right) \leq O(\sqrt{Rd}). \quad \square \end{aligned}$$

By contrast, Reference [17] shows the weaker bound $|Y_k \cdot v| \leq O(\sqrt{R \log(RD)})$.

6 TRANSVERSALS WITH OMITTED SUBGRAPHS

Consider a graph $G = (V, E)$ with a partition of its vertices into sets $V = V_1 \sqcup V_2 \sqcup \dots \sqcup V_\ell$, each of size b . We refer to these sets as *blocks* or *classes*. We wish to select exactly one vertex from each block; such a set of vertices $A \subseteq V$ is known as a *transversal*. There is a large literature on selecting transversals such that the graph induced on A omits certain subgraphs. (This problem was introduced in a slightly different form in Reference [6]; more recently it has been analyzed in References [18, 21, 22, 35, 36].) For example, when A is an independent set of G (omits the 2-clique K_2), this is referred to as an *independent transversal*.

Alon gives a short LLL-based proof that a sufficient condition for such an independent transversal to exist is $b \geq 2e\Delta$ [2], where Δ is the maximum degree of G . The cluster-expansion version of the LLL [5] easily shows that $b \geq 4\Delta$ suffices. Haxell shows non-constructively that a sufficient condition is $b \geq 2\Delta$ [19]; this condition is existentially optimal, in the sense that $b \geq 2\Delta - 1$ is not always admissible [22, 35, 36]. A similar criterion $b \geq \Delta + \lfloor \Delta/r \rfloor$ is given in Reference [21] for the existence of a transversal that induces no connected component of size greater than r . Finally, Reference [27] gives a criterion of $b \geq \Delta$ for the existence of a transversal omitting K_3 ; this is the optimal constant but the result is non-constructive.

6.1 Avoiding Large Cliques

Let us consider the problem of finding an independent transversal omitting K_s . We will be interested in the case when both s and Δ are large. More specifically, for any value of s we seek a bound of the form $b \geq \gamma_s \Delta$ (where b, Δ may go to infinity). We then seek to understand the behavior of the value of γ_s as $s \rightarrow \infty$.

We must have $\gamma_s \geq 1/(s-1)$. To see this, note that for any value of $b \geq 1$, we may take a graph consisting of s blocks each containing b vertices, where every vertex is connected to all vertices outside its block. This graph has $\Delta = b(s-1)$, and clearly any transversal contains a copy of K_s . An argument of Reference [35] shows the slightly stronger lower bound $\gamma_s \geq \frac{s}{(s-1)^2}$; intriguingly, Reference [35] also conjecture this to be exactly tight. However, a non-constructive argument of Reference [27] shows that $\gamma_s \leq 2/(s-1)$; this is the best general upper-bound on γ_s previously known.

The following result shows that the lower-bound of Reference [35] gives the correct *asymptotic* rate of growth, up to lower-order terms.

THEOREM 6.1. *For $s \geq 1$, we have $\gamma_s \leq \frac{1}{s}(1 + O(\frac{1}{\sqrt{s}}))$.*

PROOF. We define a variable X_i for each block i , wherein X_i is the choice of which vertex in block i goes into the transversal, and we use the probability distribution setting $p_{iv} = 1/b$ for each block i and vertex v in that block.

We have a separate bad event for each s -clique. We define Q by setting $Q(Y) = 1/\binom{s}{r}$ whenever Y corresponds to an r -clique in the graph, where $r < s$ is some parameter to be chosen. This satisfies the definition of a fractional hitting set, since an s -clique contains exactly $\binom{s}{r}$ r -cliques.

We will apply Theorem 2.6(c) to show that the PRA terminates with a good configuration as long as $b \geq (\Delta/s)(1 + \frac{c}{\sqrt{s}})$ and c is some sufficiently large universal constant.

For any r -clique Y , we have $p^Y Q(Y) = \frac{(1/b)^r}{\binom{s}{r}}$. We need to count how many other r -cliques Z have $Z \approx Y$. First, to enumerate all $Z \sim Y$, we may select any vertex $v \in Y$, select another vertex u from the block of v , and any choices of $r - 1$ neighbors of u . Thus, there are at most $br \binom{\Delta}{r-1}$ choices of Z with $Z \sim Y$. To count the number of r -cliques Z with $Z \bowtie Y$, note that if we fix some vertex $v \in Y$, then every vertex in Z is a neighbor of v (since Y, Z are subsets of a common s -clique). So there are at most $\binom{\Delta}{r}$ choices for Z .

So, we apply Theorem 2.6(c) with $P = \frac{1}{b^r \binom{s}{r}}$ and $D = br \binom{\Delta}{r-1} + \binom{\Delta}{r}$. We calculate

$$ePD = \frac{e \left(br \binom{\Delta}{r-1} + \binom{\Delta}{r} \right)}{b^r \binom{s}{r}} \leq \frac{e(r^2(b/\Delta) + 1)}{(b/\Delta)^r \binom{s}{r} r!}.$$

Simple calculus shows that $ePD \leq 1$ is satisfied for s sufficiently large with $r = \lceil \sqrt{s} \rceil$ and $b/\Delta = s^{-1} + 3s^{-3/2}$. This implies that for s sufficiently large and $b \geq \Delta(s^{-1} + s^{-3/2})$, the PRA will find a transversal omitting K_s . In particular, $\gamma_s \leq s^{-1} + 3s^{-3/2}$ for s sufficiently large. \square

By way of comparison, let us consider bounding γ_s via the standard LLL. Each s -clique H of the graph corresponds to a bad event. Each such bad event has probability $(1/b)^s$, and affects at most $sb\Delta^{s-1}/(s-1)!$ other s -cliques. The symmetric LLL criterion is therefore satisfied when

$$e \times (1/b)^s \times sb\Delta^{s-1}/(s-1)! \leq 1,$$

which leads to the condition $b/\Delta \geq \left(\frac{es}{(s-1)!}\right)^{\frac{1}{s-1}} = e/s + o(1/s)$, which is worse by a constant factor.

Although Theorem 6.1 shows that the PRA terminates with a configuration avoiding s -cliques, this does not lead to an efficient algorithm. The reason is that to implement the PRA, we must detect if there is some true bad event; this would require finding a clique in the graph, which is intractable. To obtain a fully constructive algorithm, we must enforce a *stronger* (but easy-to-check) condition on the graph: Not only do we avoid copies of K_s , but we in fact avoid $(s-1)$ -stars. This leads to a bound on b with slightly weaker second-order terms.

THEOREM 6.2. *Let G be a graph of maximum degree Δ whose vertex set is partitioned into blocks of size b . If*

$$b \geq \frac{\Delta}{s} \left(1 + c \sqrt{\frac{\log s}{s}} \right),$$

for some constant c , then G has a transversal that does not contain any s -stars, which can be found in randomized polynomial time.

PROOF. We use a fractional hitting set that assigns weight $1/\binom{s}{r}$ to each r -star, and zero to every other subgraph, for some integer $r \leq s$. We will use Theorem 3.8 with $K = 1$, assigning the constant vector

$$\vec{\lambda} = \alpha = \frac{s-r}{\Delta(1+r)^{2/r}}.$$

As any two r -stars H, H' that both correspond to the same s -star, will overlap in their central vertex, we see that \bowtie is null so $S_1 = 0$.

For any vertex v , there are at most $\binom{\Delta}{r}$ r -stars in which v is the central vertex and at most $\Delta \binom{\Delta-1}{r-1}$ r -stars where it is a peripheral vertex. So the condition of Theorem 3.8 becomes

$$b\alpha - b \frac{\left(\binom{\Delta}{r} + \Delta \binom{\Delta-1}{r-1}\right)}{\binom{s}{r}} \alpha^{r+1} \geq 1. \quad (11)$$

We estimate this as:

$$b\alpha - b \frac{\left(\binom{\Delta}{r} + \Delta \binom{\Delta-1}{r-1}\right)}{\binom{s}{r}} \alpha^{r+1} \geq b\alpha - b \left((r+1) \Delta^r \frac{(s-r)!}{s!} \right) \alpha^{r+1} \geq b\alpha - \frac{b(r+1)\Delta^r \alpha^{r+1}}{(s-r)^r}.$$

Thus, substituting the value of α , a sufficient condition to satisfy (11) is given by

$$b \geq (\Delta/s) \times \frac{(r+1)^{1+2/r}}{r(1-r/s)}.$$

At this point we set $r = \lceil \sqrt{s \ln s} \rceil$. As $(r+1)^{1+2/r}/r$ is a decreasing function of r , we have:

$$\frac{(r+1)^{1+2/r}}{r(1-r/s)} \leq \frac{(1 + \sqrt{s \ln s})^{1 + \frac{2}{\sqrt{s \ln s}}}}{\sqrt{s \ln s} (1 - \frac{1 + \sqrt{s \ln s}}{s})} \leq 1 + O\left(\sqrt{\frac{\log s}{s}}\right).$$

This implies that the PRA converges under the criterion

$$b \geq (\Delta/s) \left(1 + c \sqrt{\frac{\log s}{s}} \right)$$

for some sufficiently large constant $c > 0$.

To implement a step of the PRA, one must search the graph for any s -star in the current candidate transversal; this can be done easily in polynomial time. \square

Theorem 6.2 improves on Reference [27] in three distinct ways: it gives a better asymptotic estimate for γ_s ; it is fully constructive; it finds a transversal omitting not only s -cliques but also s -stars.

6.2 Bounds in Terms of Average Block Degree

The *maximum degree* Δ is a relatively crude statistic. Let us define d to be the maximum *average* degree of any class V_i and get bounds in d instead. (Formally, we take the average of the degree (in G) of all vertices in V_i , and then maximize this over all i .) For some graphs H , the PRA gives a simple method of finding independent transversals avoiding H , where b is a simple function of d .

We say that a graph H is *intersecting* if for all edges f, f' of H we have $f \cap f' \neq \emptyset$. Note that an intersecting graph H with r edges is an r -star, unless $r = 3$ in which case H can also be a triangle.

THEOREM 6.3. *Let H be an intersecting graph with r edges. Let G be a graph whose vertex set is partitioned into blocks of size at least b , and let d be the maximum average degree of any block. If $b \geq 4d/r$, then G has a transversal avoiding H that can be found in randomized polynomial time.*

PROOF. For each block i , we define a variable X_i , which is the choice of which vertex in that block to place into the transversal. We give X_i the uniform distribution over vertices in that block.

We will apply Theorem 3.8 with $K = 1$. Each copy of H in G corresponds to an atomic bad event of \mathcal{B}_1 . We define a fractional hitting-set Q_1 by setting $Q_1(\{u, v\}) = 1/r$ for each edge $f = (u, v) \in E$, and Q_1 is zero everywhere else.

Since the atomic bad events all involve exactly r edges, Q_1 satisfies the conditions of fractional hitting-set. Furthermore, any pair of edges f_1, f_2 that are both part of a copy of H overlap in at least one vertex, so \bowtie is null and $S_1 = 0$.

The vector λ used for Theorem 3.8 has all its entries equal to a scalar value $\alpha \geq 0$. Let d_v denote the degree of vertex v . Then, to prove $\lambda_i \geq 1 + \sum_k G_i(Q_k, \lambda)$, we need to show

$$b\alpha - \sum_{v \in V_i} d_v \alpha^2 / r \geq 1.$$

By definition of d , we have $\sum_{v \in V_i} d_v \leq bd$, so we need to show

$$b\alpha - bd\alpha^2 / r \geq 1.$$

When $b \geq 4d/r$, this has a solution α with $0 < \alpha \leq \frac{r}{2d}$. This shows that the PRA converges in time $O(n\alpha) \leq O(nr/2d)$. Note that H must either be a triangle, or an r -star; either of these can be found in polynomial time in G . \square

As shown in References [22, 35, 36], when $H = K_2$ this cannot be improved to $b \geq 2d - 1$. As shown in Reference [27], when H is a triangle this cannot be improved to $b \geq (1 - \epsilon)d$ for any constant $\epsilon > 0$.

7 PACKET ROUTING

Consider a graph G with N packets, each of which has a specified simple path of length at most D to reach its endpoint vertex (we refer to D as the *dilation*). In any timestep, a packet may wait at its current position, or move along the next edge on its path. Our goal is to find a schedule of smallest makespan in which, in any given timestep, an edge carries at most a single packet.

We begin by reviewing the basic strategy in Reference [25], and its improvements in References [32] and [29]. We recommend consulting [32], which has a very detailed explanation of this problem as well as many more variants than we cover here. We note that Reference [29] studied a more general version of the packet-routing problem, so their choice of parameters was not (and could not be) optimized.

We define the *congestion* C to be the maximum, over all edges, of the number of packets scheduled to traverse that edge. It is clear that D and C are both lower bounds for the makespan, and Reference [25] has shown that in fact a schedule of makespan $O(C + D)$ is possible. The work in Reference [32] provided an explicit constant bound of $39(C + D)$, as well as describing an algorithm to find such a schedule. This was improved to $23.4(C + D)$ in Reference [29] as will be described below.

While the final schedule only allows one packet to cross an edge at a time, we will relax this constraint during our construction. We consider “infeasible” schedules, in which arbitrarily many packets pass through each edge at each timestep. We define an *interval* to be a consecutive set of times in our schedule, and the *congestion* of an edge in a given interval to be the number of packets crossing that edge. If we are referring to intervals of length i , then we define a *frame* to be an interval that starts at an integer multiple of i .

From our original graph, one can easily form an (infeasible) schedule with delay D and overall congestion C . Initially, this congestion may “bunch up” in time, that is, certain edges may have very high congestion in some timesteps and very low congestion in others. Our construction will “even out” the schedule, bounding the congestion on successively smaller intervals.

Ideally, this process would eventually finish with each individual timestep (i.e., interval of length 1) having congestion roughly C/D . In this case, the infeasible schedule could be turned into a feasible schedule, by simply expanding each timestep into C/D separate timesteps.

Peis and Wiese [29] improved the bound on the makespan to $23(C + D)$ by controlling the congestion on intervals of length 2 (instead of length 1). Given our infeasible schedule, we can view each interval of length 2 as defining a new subproblem with dilation 2 and congestion C . We quote their result:

PROPOSITION 7.1 ([29]). *If $D = 2$, then there is a schedule of makespan $C + 1$ that can be found in polynomial time.*

7.1 Using the LLL to Find a Schedule

As a starting point, our construction is based on Reference [32] with some optimized parameters. We add random delays to each packet, and then allow the packet to move through each of its edges in turn without hesitation. The LLL is used to ensure that the congestion does not get too large on any interval.

LEMMA 7.2. *Suppose there is a schedule S of length L such that every interval of length i has congestion at most C . For positive integers m, C', i' with $i' \leq i/2$, suppose that*

$$e \times \Pr \left(\text{Binomial} \left(C, \frac{i'}{i - i'} \right) > C' \right) \times (Cmi^2 + 1) < 1.$$

Then there is a schedule S' of length $L' = L(1 + 1/m) + i$, in which every interval of length i' has congestion at most C' . Furthermore, S' can be constructed in expected polynomial time.

PROOF. We break the schedule S into frames of length $F = mi$, and refine each separately. Within each frame, we add a random delay to each packet separately. The delays are uniformly distributed in the range $\{0, \dots, i - i' - 1\}$ and are independent. (We refer to this as *adding a random delay in the range $i - i'$ to each packet*)

Let us fix an F -frame for the moment. Each edge f and i' -interval I has a bad event $\mathcal{B}_{f,I}$ that the congestion exceeds C' . Each f, I has at most C possible packets that could traverse it, and each does so with probability at most $p = \frac{i'}{i - i'}$. Hence the probability of $\mathcal{B}_{f,I}$ is at most the probability that a binomial random variable with C trials and probability p exceeds C' .

If a packet x was originally scheduled to cross some edge f' at time $s \leq mi$ in the schedule S , then in the schedule S' it potentially affects f' within intervals $\{s - i' + 1, \dots, s\}, \dots, \{s + (i - i'), \dots, s + (i - i') + i' - 1\}$, a total of i intervals. Thus, overall x can affect at most mi^2 other events $\mathcal{B}_{f',I'}$. Since there are at most C packets that could affect f, I , this implies that each $\mathcal{B}_{f,I}$ affects at most $d = Cmi^2$ other bad events.

By the LLL, the condition in the hypothesis guarantees that there is a positive probability that the delays avoid all bad events. In this case, we refine each frame of S to obtain a new schedule S' as desired. We can use the MT algorithm to actually find schedule S' in polynomial time.

So far, this ensures that *within each frame*, the congestion within any interval of length i' is at most C' . In the refined schedule S' there may be intervals that cross frames. To ensure that these do not pose any problems, we insert a delay of length i' between successive frames, during which no packets move at all. This step means that the schedule S' may have length up to $L(1 + 1/m) + i$. \square

Lemma 7.2 allows us to transform the original problem instance into one where C, D are small finite values, with a negligible cost to the approximation ratio. For simplicity here, we focus on the case in which C, D are very large and so certain rounding effects can be disregarded.

LEMMA 7.3. *Assume $C + D \geq 2^{896}$. There is a schedule of length at most $1.004(C + D)$ and in which the congestion on any interval of length 2^{24} is at most 17040600. Furthermore, this schedule can be produced in randomized polynomial time.*

PROOF. We provide a sketch here; see Reference [32] for a much more thorough explanation of this process.

Define the sequence a_k recursively as $a_0 = 256$ and $a_{k+1} = 2^{a_k}$. There is a unique k with $a_k^{3.5} \leq C + D < a_{k+1}^{3.5}$. By a slight variant on Lemma 7.2, one can add delays to obtain a schedule of length $C + D$, in which the congestion on any interval of length $i' = a_k^3$ is at most $C' = i'(1 + 4/a_k)$.

At this point, we repeatedly apply Lemma 7.2 with $i = a_j$, $i' = a_{j+1}$, for $j = k - 1, \dots, 0$. At each step, this increases the length of the resulting schedule from L_j to $L_j(1 + 1/a_{j+1}) + a_j$, and increases the congestion on the relevant interval from $i(1 + 4/a_k)$ to

$$i(1 + 4/a_k) \prod_{j=0}^{k-1} (1 + 4/a_j) \left(\frac{1}{1 - (a_j/a_{j+1})^3} \right).$$

(We use the Chernoff bound to estimate the binomial tail in Lemma 7.2.)

For $C + D \geq a_k^{3.5}$, a simple calculation shows that the schedule length increases from $C + D$ (after the original refinement) to at most $1.004(C + D)$. In the final step of this analysis, we are bounding the congestion of intervals of length $a_0^3 = 2^{24}$, and the congestion on such an interval is at most 17040600. \square

LEMMA 7.4. *If $C + D \geq 2^{896}$, then there is a feasible schedule of length at most $10.92(C + D)$, which can be constructed in randomized polynomial time.*

PROOF. By Lemma 7.3, we form a schedule S_1 , of length $L_1 \leq 1.004(C + D)$, in which each interval of length 2^{24} has congestion at most 17040600.

Now apply Lemma 7.2 to S_1 , with $m = 64$, $i' = 1024$, $C' = 1385$ to obtain a schedule S_2 , of length $L_2 \leq 1.0157L_1 + 2^{24}$, in which each interval of length 1024 has congestion at most 1385.

Now apply Lemma 7.2 to S_2 with $m = 64$, $i' = 2$, $C' = 20$, to obtain a schedule S_3 of length $L_3 \leq 1.0157L_2 + 1024$, in which each frame of length 2 has congestion at most 20.

Now apply Proposition 7.1 to S_3 , expanding each 2-frame to a *feasible* schedule of length 21. The total length of the resulting schedule is at most $\frac{21}{2}L_3 \leq 10.92(C + D)$. \square

7.2 The PRA Applied to Packet Routing

The schedule modification in Lemma 7.2 essentially comes down to an assignment-packing problem: within each frame, we assign a delay to each packet, and a bad event corresponds to an edge receiving an excessive congestion in some time interval. We thus modify Lemma 7.2 to use the PRA instead of the LLL.

PROPOSITION 7.5. *Suppose there is a schedule S of length L such that every interval of length i has congestion at most C . Let m, C', d, i' be positive integers with $i' < i$ and $d \leq C'$, and let $\alpha \in [0, 1]$ be a real number.*

Define

$$p = \frac{(Ci'\alpha)^d}{d!(C'+1)^d}.$$

Suppose that $p < 1$ and

$$(i - i')\alpha - \frac{mi^2 dp}{C(1 - p)} \geq 1.$$

Then there is a schedule S' of length $L' = L(1 + 1/m) + i$, in which every interval of length i' has congestion at most C' . Furthermore, such a schedule can be found in polynomial time.

PROOF. Suppose we add delays in the range $\{0, \dots, i - i' - 1\}$ uniformly to each packet within each frame of length $F = mi$. In this case, we have a variable corresponding to each packet x , and

for each delay δ we assign $\lambda_{x,\delta} = \alpha$. For each edge f and i' -interval I , we have a complex bad event $\mathcal{B}_{f,I}$ that the congestion in the interval exceeds C' . Each such bad event uses the fractional hitting-set $Q_{f,I}$ with parameter d as described in Theorem 4.5.

For a given f, I , we must compute μ , which is the total contribution of λ summed over all packets/delays, that could contribute to the congestion of that edge-interval. There are at most C packets that could be scheduled to pass through the given edge, and there are i' possible delays that affect f, I . So, in all, the total contribution is $\mu \leq Ci'\alpha$. The bad event is that this exceeds C' , so $t = C' + 1$. By Theorem 4.5, this gives $S_{f,I} = S(\mathcal{B}_{f,I}, Q_{f,I}, \lambda) \leq \frac{\mu^d}{d!(C'+1)} \leq p$.

Next, consider some packet x ; we wish to compute μ_x , which is the total contribution to the bad event $\mathcal{B}_{f,I}$ summed over all possible delays to packet x . There are at most i' delays that can cause x to transit f within I , hence $\mu_x \leq i'\alpha$ if packet x could cross edge f in interval I . So $G_x(Q_{f,I}, \lambda) \leq \frac{i'\alpha}{\mu} d \frac{\mu^d}{d!(C'+1)} \leq \frac{dp}{C}$.

Each packet x affects up to mi edges within the frame; if a packet x was originally scheduled to cross an edge f at time $s \leq mi$ in the schedule S , then in the schedule S' it potentially affects f within the i intervals $\{s - i' + 1, \dots, s\}, \dots, \{s + (i - i'), \dots, s + (i - i') + i' - 1\}$. So summing $G_x(Q_{f,I}, \lambda)$ over all f, I affected by packet x yields

$$\sum_{f,I} G_x(Q_{f,I}, \lambda) \leq \frac{mi^2 dp}{C}.$$

To apply Theorem 3.8 each packet x must satisfy the constraint

$$\lambda_x \geq 1 + \sum_{f,I} \frac{G_x(Q_{f,I}, \lambda)}{1 - S_{f,I}}. \quad (12)$$

Each packet x has $\lambda_x = (i - i')\alpha$ and $S_{f,I} \leq p$ and $\sum_{f,I} G_x(Q_{f,I}, \lambda) \leq mi^2 dp/C$, so Reference (12) becomes

$$(i - i')\alpha \geq 1 + \frac{mi^2 dp}{(1 - p)C}.$$

This is precisely the constraint specified in the hypothesis. The expected number of resamplings is $\sum_{x,\delta} \lambda_{x,\delta} \leq ND\alpha$, which is polynomially bounded. \square

We can use this to improve various steps in the construction.

PROPOSITION 7.6. *Suppose $C + D \geq 2^{896}$. Then there is a schedule of length $\leq 1.00652(C + D)$, in which every interval of length 1024 has congestion at most 1320, which can be constructed in randomized polynomial time.*

PROOF. By Lemma 7.3, form a schedule S_1 , of length $L_1 \leq 1.004(C + D)$, in which each interval of length 2^{24} has congestion at most 17040600. Apply Proposition 7.5 with $\alpha = 5.98328 \times 10^{-8}$, $C' = 1320$, $d = 270$, $m = 400$ to obtain a schedule S_2 of length $L_2 \leq 1.0025L_1 + 2^{24} \leq 1.00652(C + D)$, in which each interval of length 1024 has congestion at most 1320. \square

THEOREM 7.7. *Suppose $C + D \geq 2^{896}$. Then there is a schedule of length at most $8.61(C + D)$, which can be constructed in randomized polynomial time.*

PROOF. By Proposition 7.6, there is a schedule S_1 of length at most $L_1 = 1.00652(C + D)$ in which each interval of length 1024 has congestion at most 1320.

Now apply Lemma 7.3 with $i = 1024$, $C = 1320$, $i' = 2$, $m = 100$, $C' = 16$, $d = 12$, $\alpha = 0.00107911$ to obtain a schedule S_2 of length $L_2 \leq 1.01L_1 + 1024$, in which each interval of length 2 has congestion at most 16.

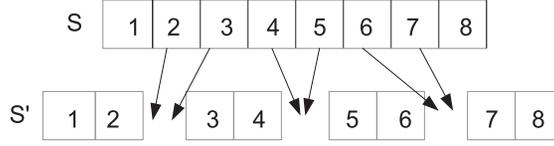


Fig. 1. The packets in the original schedule S are shifted into overflow times in the schedule S' .

Now apply Proposition 7.1 to S_2 , expanding each 2-frame to a *feasible* schedule of length 17. The total length of the resulting schedule is at most $\frac{17}{2}L_2 \leq 8.61(C + D)$. \square

7.3 Better Scheduling of the Final 2-frame

Let us examine more closely the penultimate stage in the proof of Theorem 7.7, in which the schedule gets divided into 2-frames where the congestion of each edge is bounded by some parameter C' . For a given edge f and time t , we define $c_t(f)$ to be the number of packets scheduled to cross f at time t ; for a given packet x and time t , we define $e_t(x)$ to be the identity of the packet edge at time t (possibly there is no edge, in which case $e_t(x) = \emptyset$).

For a given value of t , it is relatively likely that $c_t(f) + c_{t+1}(f)$ or $c_{t+2}(f) + c_{t+3}(f)$ are much larger than their mean. However, it is unlikely that *both* these bad events happen simultaneously. To take advantage of this, we construct a schedule in which we insert an “overflow” time between the 2-frames to handle the situation in which *either* $c_t(f) + c_{t+1}(f)$ or $c_{t+2}(f) + c_{t+3}(f)$ is too large. Our goal will be to modify either of the intervals $\{t, t + 1\}$ or $\{t + 2, t + 3\}$ to ensure that both have congestion at most some parameter T .

For a given 2-frame I , we add two overflow time slots, before and after I , to schedule the excess packets. If an edge f has more than T transits scheduled during the interval I , then we can fix this by either finding some packet x with $e_1(x) = f$ and shifting it into the earlier overflow time, or by finding some packet x with $e_2(x) = f$, and shifting it into the later overflow time. See Figure 1.

We need to be careful to account for how often a given edge f appears as $e_t(x)$ or $e_{t+1}(x)$. For example, if there are no (remaining) packets with $e_{t+1}(x) = f$, then we are only allowed to shift f into the earlier overflow, and similarly if there are no remaining packets with $e_t(x) = f$. Keeping this constraint in mind, we seek to equalize as far as possible the distribution of edges into earlier and later overflows. We do so as follows:

- 1: **for** each edge f and each odd integer t **do**
- 2: **while** $c_t(f) + c_{t+1}(f) > T$ **do**
- 3: if $c_t(f) = 0$, then shift one packet into the later overflow time.
- 4: else if $c_{t+1}(f) = 0$, then shift one packet into the earlier overflow time.
- 5: else if $c_t(f) + c_{t+1}(f) = \text{odd}$, then shift one packet into the earlier overflow time.
- 6: else if $c_t(f) + c_{t+1}(f) = \text{even}$, then shift one packet into the later overflow time.

For any odd integer t and edge f , let c' denote the congestions at the end of this overflow-shifting process, so that $c'_t(f) + c'_{t+1}(f) \leq T$. The number of packets shifted into the earlier (respectively, later) overflow time can be viewed as a function of the original values $c_t(f), c_{t+1}(f)$. We denote these “overflow” functions by $\text{OF}^-(c_t, c_{t+1}; T)$ and $\text{OF}^+(c_t, c_{t+1}; T)$, respectively. Specifically we get the following condition:

PROPOSITION 7.8. *Let S be a schedule of length L , and let $c_t(f)$ for $t = 1, \dots, L$ denote the number of times f is scheduled as the t^{th} edge of a packet. Suppose that $T \geq T' \geq 1$, and suppose that for all edges f and all odd integers t the schedule S satisfies the constraint*

$$\text{OF}^+(c_t(f), c_{t+1}(f); T) + \text{OF}^-(c_{t+2}(f), c_{t+3}(f); T) \leq T'.$$

Then there is a feasible schedule S' of makespan $(L + 1)/2 \times (T + T' + 2) - 2$, which can be constructed in polynomial time. (Note that for $t < 1$ and $t > L$, we define $c_t(f) = 0$.)

PROOF. After the modification, each 2-frame has congestion at most T , while each overflow time has congestion at most T' . Each overflow time has delay at most 2, since for any packet x , there may be at most two edges scheduled into that overflow time, namely the edge that had been originally marked as the second edge of the earlier 2-frame, and the edge that had been originally marked as the first edge of the latter 2-frame. Hence each 2-frame can be scheduled in time $T + 1$ and each overflow can be scheduled in time $T' + 1$. Also note that the final and initial overflow times have delay 1, so they can be scheduled in time T' .

Let us first suppose that L is even. There are $L/2$ 2-frames in the original schedule and so there are $L/2 + 1$ overflow periods. Hence the total cost is at most $\frac{L}{2}(T + 1) + (\frac{L}{2} - 1)(T' + 1) + 2T' = \frac{L}{2}(2 + T + T') + T' - 1$. The condition $T \geq T'$ implies that this is at most $(L + 1)/2 \times (T + T' + 2) - 2$.

If L is odd, then we can merge the final overflow time into the final ordinary time; in this final 2-frame, each edge has congestion at most $T + T'$. Thus, overall the cost is at most $\frac{L-1}{2}(T + 1) + T' + (\frac{L-1}{2} - 1)(T' + 1) + (T + T' + 1) = (L + 1)/2 \times (T + T' + 2) - 2$. \square

The conditions required by Proposition 7.8 are local, in the sense that any violation is any event that affects an individual edge and a 4-interval that starts at an odd time t . We refer to the conditions required by this Proposition as the *4-interval-conditions* (abbreviated *4IC*); these conditions can be viewed as either pertaining to an entire schedule or to individual 4-interval.

We will use the PRA to find a schedule satisfying the conditions of Proposition 7.8.

PROPOSITION 7.9. Let $m = 36, C = 1320, i = 1024, T = 6, T' = 5$.

Suppose there is a schedule S of length L such that every interval of length i has congestion at most C . There is a schedule of length $L' \leq (1 + 1/m)L + i$, which satisfies the 4IC with respect to T, T' . This schedule can be produced in polynomial time.

PROOF. Our plan is to break the schedule into frames of size $F = mi$; within each packet and frame we add a random delay in the range $i - 4$. Let us fix a frame for the moment.

For each edge f , and 4-interval I starting at time s where s is an odd integer, we have a complex bad event $\mathcal{B}_{f,I}$ that

$$\text{OF}^+(c_s(f), c_{s+1}(f); T) + \text{OF}^-(c_{s+2}(f), c_{s+3}(f); T) > T'.$$

For this edge/interval f, I , and any packet x with delay t , we say that $\langle x, t \rangle$ has *type* j , if that packet-delay assignment would cause the given packet x to land at position $s + j$ within the bad event, for $j = 0, \dots, 3$. If that assignment x, s does not contribute to $\mathcal{B}_{f,I}$, then $\langle x, s \rangle$ has no type. Note that each f, I has at most C packet-delay combinations of each type.

For a bad event $\mathcal{B}_{f,I}$ and a fractional hitting-set $Q_{f,I}$, we define the quantities Φ and Φ_j , for $j = 0, 1, 2, 3$, as

$$\Phi_j = \max_{f,I} \max_{\substack{\langle x, t \rangle \text{ has type } j \\ \text{for } f, I}} \sum_{Y \ni \langle x, t \rangle} Q_{f,I}(Y) \lambda^Y, \quad \Phi = \max_{f,I} \sum_Y Q_{f,I}(Y) \lambda^Y.$$

We will apply Theorem 3.8 using a separate complex-bad event for each f, I , and a separate variable for each packet (the value of a variable is the chosen delay), and the vector $\vec{\lambda}_{x,t} = \alpha = 0.001051$. For each such label f, I we have $S_{f,I} = S(\mathcal{B}_{f,I}, Q_{f,I}, \lambda) \leq \sum_Y Q_{f,I}(Y) \lambda^Y \leq \Phi$.

For any packet x and delay t and $j = 0, \dots, 3$, there are at most $mi/2$ pairs f, I for which packet x, t has type j . Each such f, I has $G_{x,t}(Q_{f,I}, \lambda) \leq \Phi_j$. Summing over f, I and the i choices for the

delay t gives

$$\sum_{f,I} G_x(Q_{f,I}, \lambda) \leq \frac{mi^2}{2} (\Phi_0 + \Phi_1 + \Phi_2 + \Phi_3).$$

By Theorem 3.8, each packet x must satisfy the constraint

$$\lambda_x \geq 1 + \sum_{f,I} \frac{G_x(Q_{f,I}, \lambda)}{1 - S_{f,I}}.$$

We have $\lambda_x = (i - 4)\alpha$, and so it suffices to satisfy the condition

$$(i - 4)\alpha - \frac{mi^2}{2} \frac{\Phi_0 + \Phi_1 + \Phi_2 + \Phi_3}{1 - \Phi} \geq 1 \quad (13)$$

to find acceptable delays. Such delays lead to a schedule of length $L' \leq (1 + 1/m)L + i$, which satisfies the 4IC with T, T' .

Thus, we have reduced our problem to constructing fractional hitting-sets $Q_{f,I}$ which have a sufficiently small value for $\frac{\Phi_0 + \Phi_1 + \Phi_2 + \Phi_3}{1 - \Phi}$. Although we have stated the proposition for a particular choice of parameters, we will walk through the algorithm we use to construct it next.

The bad event depends $\mathcal{B}_{f,I}$ is determined by the number of variables of each type assigned to edge f on interval I . There are at most C such variables of each type; to simplify the notation, we suppose there are exactly C . The fractional hitting-set $Q_{f,I}$ assigns weights to any subset of the $4C$ variables involved in the bad event; we can write such a subset as $Y = Y_1 \cup Y_2 \cup Y_3 \cup Y_4$, where the packet/delays in Y_j all have type j .

We will make $Q_{f,I}$ symmetric, in the sense that for any such $Y = Y_0 \cup Y_1 \cup Y_2 \cup Y_3$, the value of $Q_{f,I}(Y)$ depends solely on the cardinalities $|Y_0|, |Y_1|, |Y_2|, |Y_3|$. Thus, we define

$$Q_{f,I}(Y_0 \cup Y_1 \cup Y_2 \cup Y_3) = b(|Y_0|, |Y_1|, |Y_2|, |Y_3|),$$

where $b : [C]^4 \rightarrow [0, 1]$ is a function that we will determine. Let us define $\hat{\Phi}_0$, which serve as an upper bound on Φ_0 , by

$$\hat{\Phi}_0 = \sum_{y_0, y_1, y_2, y_3} \binom{C-1}{y_0-1} \binom{C}{y_1} \binom{C}{y_2} \binom{C}{y_3} b(y_0, y_1, y_2, y_3) \alpha^{y_0+y_1+y_2+y_3}$$

and similarly for $\hat{\Phi}_1, \hat{\Phi}_2, \hat{\Phi}_3, \hat{\Phi}$. Here y_0, y_1, y_2, y_3 denote the possible cardinalities of Y_0, Y_1, Y_2, Y_3 , respectively. (The reason for the term $\binom{C-1}{y_0-1}$ here, as opposed to $\binom{C}{y_0}$, is that in computing Φ_0 , we have fixed the presence of a single packet/delay $\langle x, t \rangle$ with type 0 for the given f, I . Thus, there are only $\binom{C-1}{y_0-1}$ choices for the *additional* type-0 packets involved in f, I .)

We say a tuple (k_0, k_1, k_2, k_3) is *bad* if it satisfies

$$\text{OF}^+(k_0, k_1; T) + \text{OF}^-(k_2, k_3; T) > T',$$

and we say it is *minimal bad* if it is bad, but no other (k'_0, k'_1, k'_2, k'_3) strictly smaller than it is bad.

The fractional hitting-set must satisfy $\sum_{Y \subseteq A} Q(Y) \geq 1$ for any $A \in \mathcal{B}_{f,I}$. By symmetry, this means that if k_0, k_1, k_2, k_3 are minimal bad, then we require

$$\sum_{y_0, y_1, y_2, y_3} \binom{k_0}{y_0} \binom{k_1}{y_1} \binom{k_2}{y_2} \binom{k_3}{y_3} b(y_0, y_1, y_2, y_3) \geq 1. \quad (14)$$

We set $b(y_0, y_1, y_2, y_3) = 0$ unless there is some minimal bad tuple (k_0, k_1, k_2, k_3) with $k_0 \geq y_0, \dots, k_3 \geq y_3$.

We are trying to satisfy $(\hat{\Phi}_0 + \hat{\Phi}_1 + \hat{\Phi}_2 + \hat{\Phi}_3)/(1 - \hat{\Phi}) \leq t$ for some target value t . When t is fixed, this is equivalently to minimizing $\hat{\Phi}_0 + \hat{\Phi}_1 + \hat{\Phi}_2 + \hat{\Phi}_3 + t\hat{\Phi}$. If we view the collection of all values

$b(y_0, y_1, y_2, y_3)$ as linear unknowns, then we can view both the objective function and the constraints as linear. Hence this defines an LP, which we can solve using standard algorithms. We can then optimize t by binary search.

For $T = 6, T' = 5$, the resulting linear program has 12,000 variables and 259 constraints. This is too large to write explicitly, but we wrote computer code to generate and solves it. The resulting hitting-set achieves a bound of

$$\frac{\hat{\Phi}_0 + \hat{\Phi}_1 + \hat{\Phi}_2 + \hat{\Phi}_3}{1 - \hat{\Phi}} \leq 3.495 \times 10^{-9}$$

and this satisfies Equation (13). (We recommend that any reader who wishes to recover it should construct the linear program of Equation (14) and solve it for themselves.)

To show that the number of resamplings is polynomially bounded, note that we are treating each F -frame separately. Thus, all of the quantities $\sum_x \lambda_x$, and so on, for the PRA are functions of parameters m, C, i, T, T' : In particular, they do not depend on the overall problem size. So, the expected number of resamplings within each frame is some constant, and so the overall number of resamplings is poly(N). \square

We now apply this construction to replace the two final steps in the construction of Section 7.3.

THEOREM 7.10. *There is a feasible schedule of makespan at most $6.73(C + D)$, which can be constructed in randomized polynomial time.*

PROOF. We give the full proof in Appendix E (which has numerous cases and calculations). Here, we focus on the most interesting case, where $C + D \geq 2^{896}$.

By Proposition 7.6, we obtain a schedule S_1 of length $L_1 \leq 1.00652(C + D)$, in which each interval of length 1024 has congestion at most 1320.

Now apply Proposition 7.9. This gives a schedule S_2 of length $L_2 \leq 1.02779L_1 + 1024$ satisfying the 4IC with $T = 6, T' = 5$. By Proposition 7.8, this yields a schedule whose makespan is $6.5L_2 + 4.5 \leq 6.73(C + D)$. \square

APPENDICES

A COMPARISON WITH THE MT ALGORITHM

To compare our results with the MT algorithm, let us consider the class of assignment-packing problems where $c_k = R$ for all k . We have seen (Proposition 4.8) that the PRA converges in polynomial time if the bad events are defined with right-hand-side vector $b_k = R'$; here R' is a function of R and of the maximum ℓ_1 -norm D of the constraint matrix. Crucially, R' is scale-free: it does *not* depend on the number of variables n or number of constraints m . By contrast, we show that MT cannot achieve in sub-exponential time *any* value b_k which depends solely on R, D or the number of non-zeroes in each column D' . This holds even if the entries of A are in the set $\{0, 1\}$.

There are many possible parametrizations of the LLL and the MT algorithm for this problem, including strategies based on iterated applications. We consider only the simplest of these, which has a separate bad event for each row of the constraint matrix.

PROPOSITION A.1. *Let R, R' be fixed real numbers with $1 \leq R \leq R'$. For m sufficiently large there is an assignment-packing problem with the following properties:*

- (1) *The system has entries in $\{0, 1\}$, has m constraints, and has $n = \Theta(Rm)$ variables.*
- (2) *There is a fractional solution z achieving right-hand-side value $c_k = R$.*
- (3) *For each $i \in [n]$ and each possible assignment $j \in F_i$, there is exactly one row k with $a_{k,i,j} > 0$. (So the matrix has maximum ℓ_0 norm of $D' = 1$.)*

- (4) Suppose we run the PRA, using resampling probabilities given by $p_{ij} = z_{ij}$, and with right-hand-side values given by $b_k = R + C\sqrt{R}$ for a universal constant C . Then the PRA (with an appropriate fractional hitting-set) terminates in expected polynomial time.
- (5) Suppose we run the MT algorithm, using resampling probabilities given by $p_{ij} = z_{ij}$ and with right-hand-side values b_k given by $b_k = R'$. Then the probability that this MT algorithm terminates after $2^{\phi m}$ steps is at most $2^{-\phi' m}$, where $\phi, \phi' > 0$ are parameters depending solely on R, R' .

PROOF. For $i \in [n]$, the variable X_i has domain $F_i = \{1, \dots, m\}$. To construct the constraint matrix A , we select for each $i \in [n]$ a permutation $\pi_i \in S_m$ independently and uniformly at random. For $k \in [m]$, we then set $a_{k,i,j} = [\pi_i(k) = j]$. Thus, all the entries of the constraint matrix are either 0 or 1. Also, observe that for any i, j , the only value of k with $a_{k,i,j} > 0$ is given by $k = \pi_i^{-1}(j)$.

This system has a fractional solution of $z_{i,j} = 1/m$ for all i, j . For any value of k this gives

$$\sum_{i,j} a_{k,i,j} z_{i,j} = \sum_{i,j} [\pi_i(k) = j]/m = n/m$$

For $n = \lfloor Rm \rfloor$, this satisfies the constraints fractionally with right-hand-side vector $c_k = R$.

The convergence of the PRA with this fractional vector z follows from Proposition 4.8.

Finally, we will show that the MT algorithm requires a long time to terminate. To begin, we will show that for any fixed vector $x_1, \dots, x_n \in [m]^n$, the probability (over the random choice of A) that $A_k x \leq R'$ for all $k \in [m]$, is at most $e^{-\Omega(m)}$. To show this, we view the vector of counts $A_1 x, \dots, A_m x$ as what is known as a *competing ball-and-urns problem*; there is an urn corresponding to each $k \in [m]$, there is a ball corresponding to each $i \in [n]$, and the value of $A_k x$ is the number of balls placed into urn k . We place ball i into urn k iff $\pi_i(k) = x_i$ — in other words, the placement of each ball is independently chosen among the k urns.

Consider some fixed value of k . The value of $A_k x$ is a binomial random variable, with number of trials n and success probability $1/m$, and so

$$\begin{aligned} \Pr(A_k x > R') &\geq \binom{n}{R'+1} (1/m)^{R'+1} (1-1/m)^{n-R'-1} \geq \frac{(n-R')^{R'+1} (1-1/m)^{n-R'-1}}{(R'+1)! m^{R'+1}} \\ &\geq \frac{(Rm-1-R')^{R'+1} (1-1/m)^{Rm-R'-1}}{(R'+1)! m^{R'+1}} \quad \text{as } Rm-1 \leq n \leq Rm \\ &\geq \Omega(1) \quad \text{for fixed } R, R'. \end{aligned}$$

As shown in Reference [8], the events $A_1 x \leq R', \dots, A_m x \leq R'$ are negatively correlated. Thus,

$$\Pr\left(\bigwedge_{k=1}^m A_k \leq R'\right) \leq \prod_{k=1}^m \Pr(A_k \leq R') \leq \prod_{k=1}^m (1 - (1 - \Omega(1))) \leq e^{-\Omega(m)}.$$

Every constraint in this CSP depends on every variable. Thus, whenever the MT resamples a bad event, it resamples all variables. So after the MT performs T resamplings, the current value of the variables X_1, \dots, X_n is simply the T th row of the resampling table. Consequently, a necessary condition for the MT algorithm to terminate after T steps is that one of the first T rows of the resampling table satisfies all the constraints. Since any individual row satisfies all the constraints with probability $e^{-\Omega(m)}$, the probability that MT terminates after T rounds is at most $T e^{-\Omega(m)}$. This is $e^{-\Omega(m)}$ unless $T \geq e^{\Omega(m)}$. \square

B SOME RESULTS ON THE MT DISTRIBUTION

If Theorem 2.6 holds, then we know that there exists a configuration that avoids all bad events. We may wish to learn more about such configurations, other than that they exist. One useful tool

is the *MT-distribution*, which is the distribution on the variables X_1, \dots, X_n at the termination of the PRA. We write $\Pr_{MT}(E)$ to mean the probability of event E in this space. (Note that, in order for this to be well-defined, we must choose a fixed rule for which bad event to resample if there are multiple candidates; the bounds we derive hold for any such rule.) This probability space has been analyzed for the MT algorithm in References [11, 12, 15].

In this section we show bounds on the MT distribution analogous to those shown in Reference [11] for the original MT algorithm. We also examine how the parametrization in terms of λ gives particularly simple formulas, which can be useful even for analyzing the MT algorithm. We illustrate by using the MT distribution (for the original MT algorithm) for bounds on weighted independent transversals.

We need two preliminary definitions. For $E \in \mathcal{A}$ and $\mathcal{B} \subseteq \mathcal{A}$, define

$$\mathcal{B}[E] = \{B \in \mathcal{B} \mid E \not\subseteq B\}.$$

We also define the *strict neighbor-set* of E as follows:

Definition B.1 (Strict Neighbor-Set). For a set $E \in \mathcal{A}$ and a set $\mathcal{T} \subseteq \mathcal{A}$, we say that \mathcal{T} is a *strict neighbor-set* for E (or $\mathcal{T} \in \text{SNS}(E)$) if the following conditions hold:

- (1) Every $Z \in \mathcal{T}$ has $Z \sim E$.
- (2) There do not exist $Z, Z' \in \mathcal{T}$ with $Z \sim Z'$.

THEOREM B.2. *Let $E \in \mathcal{A}$ and suppose that μ satisfies Theorem 2.6 for the set of bad events $\mathcal{B}[E]$. Then, when we run the PRA on events \mathcal{B} , we have*

$$\Pr_{MT}(E) \leq p^E \sum_{\mathcal{T} \in \text{SNS}(E)} \prod_{Y \in \mathcal{T}} \mu(Y) \leq p^E \prod_{Y \sim E} (1 + \mu(Y)).$$

PROOF. We assume that there is no bad event $B \subseteq E$; for, if so, the the PRA output can never satisfy E and so the result holds trivially.

We use a coupling construction following that of Reference [11]. Consider running the PRA with the set of bad events $\mathcal{B}' = \mathcal{B}[E] \cup \{E\}$ and fractional hitting-set Q' defined by

$$Q'(Y) = \begin{cases} 1 & \text{if } Y = E \\ Q(Y) & \text{otherwise} \end{cases}.$$

When we run the PRA on \mathcal{B}' , we make a small change: whenever there is a choice of bad event to resample, we will always choose to resample E before $B \in \mathcal{B}$ if possible. We take advantage here of our freedom to select an arbitrary bad event to resample, if there are multiple choices. Observe that the PRA on \mathcal{B} has identical behavior to the PRA on \mathcal{B}' , up to the first time t when E is true. Since $Q(E) = 1$ and there are no bad events $B \subseteq E$, the PRA on \mathcal{B}' will have E as its resampled set at time t . The witness tree $\hat{\tau}^t$ thus has its root node labeled E .

We make a number of other observations about this tree $\hat{\tau}^t$. First, every subtree of $\hat{\tau}^t$ is a proper tree-structure with respect to the set of bad events $\mathcal{B}[E]$. The reason for this is E is never the resampled set before time t , and so does not affect the generation of $\hat{\tau}^t$. Second, consider the set of children w_1, \dots, w_s of the root node of $\hat{\tau}^t$. By Proposition 2.13, these have distinct labels and $\{L(w_1), \dots, L(w_s)\} \in \text{NS}(E)$. Suppose that $L(w_i) \bowtie E$. Then $E \subseteq B, L(w_i) \subseteq B$ for some bad event $B \in \mathcal{B}'$. By definition of \mathcal{B}' this is only possible if $E = B$ in which case $L(w_i) = E \not\bowtie E$. So in fact $\{L(w_1), \dots, L(w_s)\} \in \text{SNS}(E)$.

Define Γ^* to be the set of all tree-structures satisfying these properties. By Lemma 2.9, we have:

$$\Pr_{MT}(E) \leq [\text{some } \tau \in \Gamma^* \text{ appears during execution of PRA on } \mathcal{B}'] \leq \sum_{\tau \in \Gamma^*} w(\tau).$$

Now, any $\tau \in \Gamma^*$ has a root node labeled E , and its children w_1, \dots, w_s have distinct labels $\{L(w_1), \dots, L(w_s)\} \in \text{SNS}(E)$. Furthermore, the subtrees of w_1, \dots, w_s are proper tree-structures with respect to $\mathcal{B}[E]$. Thus, by Proposition 2.14, we have

$$\sum_{\tau \in \Gamma^*} w(\tau) \leq p^E Q(E) \sum_{\mathcal{T} \in \text{SNS}(E)} \prod_{Y \in \mathcal{T}} \sum_{\tau_Y \in \Gamma(Y)} w(\tau_Y) \leq p^E Q(E) \sum_{\mathcal{T} \in \text{SNS}(E)} \prod_{Y \in \mathcal{T}} \mu(Y).$$

Finally, note that $Q(E) = 1$, and we have shown the first bound on the probability of E . \square

We can extend this result to the setting of Theorem 3.6 and Theorem 3.8:

COROLLARY B.3. *Let $E \in \mathcal{A}$ and suppose that μ satisfies Theorem 3.6 for events $\mathcal{B}[E]$. Then*

$$\Pr_{MT}(E) \leq p^E \sum_{\mathcal{T} \in \text{SNS}(E)} \prod_{Y \in \mathcal{T}} \sum_k \mu(Y, k) \leq p^E \prod_{Y \sim E} \left(1 + \sum_k \mu(Y, k) \right).$$

PROOF. Consider forming some strict neighbor-set $\mathcal{T} \in \text{SNS}(E)$, with respect to the expanded set of bad events $\tilde{\mathcal{B}}$ over the set of elements $\tilde{\mathcal{X}}$. If \mathcal{T} is not good, then its contribution $\prod_{Y \in \mathcal{T}} \tilde{\mu}(Y)$ is zero. If \mathcal{T} is good, then it corresponds to $\{(Y_1, k_1), \dots, (Y_r, k_r)\}$, where $\{Y_1, \dots, Y_r\}$ is a strict neighbor-set of E (with respect to the original set of elements \mathcal{X}), and, furthermore, $\prod_{Y \in \mathcal{T}} \tilde{\mu}(Y) = \prod_{i=1}^r \mu(Y_i, k_i)$. Thus,

$$\sum_{\substack{\mathcal{T} \subseteq \tilde{\mathcal{A}} \\ \mathcal{T} \in \text{SNS}(E)}} \prod_{Y \in \mathcal{T}} \tilde{\mu}(Y) \leq \sum_{\substack{\mathcal{T} \subseteq \tilde{\mathcal{A}} \\ \mathcal{T} \in \text{SNS}(E)}} \prod_{Y \in \mathcal{T}} \sum_{k \in [K]} \mu(Y, k). \quad \square$$

THEOREM B.4. *If λ satisfies Theorem 3.8, then any atomic set E has $\Pr_{MT}(E) \leq \lambda^E$.*

PROOF. Let us enumerate $\mathcal{T} \in \text{SNS}(E)$ to apply Theorem B.3. For each $(i, j) \in E$, the set \mathcal{T} may contain one or zero sets $Y \sim i$. Therefore, Theorem B.3 gives:

$$\begin{aligned} \Pr_{MT}(E) &\leq p^E \sum_{\mathcal{T} \in \text{SNS}(E)} \prod_{Y \in \mathcal{T}} \sum_k \mu(Y, k) \leq p^E \prod_{(i,j) \in E} \left(1 + \sum_{Y \sim i} \sum_k \mu(Y, k) \right) \\ &\leq p^E \prod_{(i,j) \in E} \lambda_i = \lambda^E \quad (\text{by Proposition 3.9}) \quad \square \end{aligned}$$

We can obtain a stronger bound than Theorem B.4 when E is defined by a *single variable*. To state this result, it is useful to define

$$H_{i,j} = \sum_k \sum_{Y \ni (i,j)} \frac{Q_k(Y) \lambda^Y}{1 - S_k}$$

and similarly the ‘‘summation notation’’ $H_i = \sum_j H_{i,j}$. We also assume throughout that $\lambda_{i,j} \geq H_{i,j}$ for each (i, j) (if not, simply set $\lambda_{i,j} = 0$).

THEOREM B.5. *Let $u \in [n]$, and suppose that λ satisfies Theorem 3.8. Let $J \subseteq F_u$, where recall F_u is the set of possible assigned value to variable X_u . Then:*

$$\Pr_{MT}(X_u \in J) \leq \frac{\sum_{j \in J} \lambda_{u,j}}{\lambda_u - H_u + \sum_{j \in J} H_{u,j}}.$$

PROOF. We define a function μ for the set of bad events $\mathcal{B}[E]$; note that $\mathcal{B}[E]$ is derived from \mathcal{B} by removing every bad event $B \in \mathcal{B}$ such that $(i, j) \in B$ for some $j \in J$. We therefore define

$$\mu(Y, k) = \begin{cases} \frac{\lambda^Y Q_k(Y)}{1-S_k} & \text{if } u \approx Y \\ \alpha \frac{\lambda^Y Q_k(Y)}{1-S_k} & \text{if } (u, j) \in Y \text{ for } j \notin J, \\ 0 & \text{if } (u, j) \in Y \text{ for } j \in J \end{cases}$$

where $\alpha \in [0, 1]$ is some parameter to be determined. We need to check that μ satisfies Theorem 3.6 with respect to $\mathcal{B}[E]$. This is nearly identical to the proof of Theorem 3.8; the only difficult case is to check the condition on $\mu(Y, k)$ where $(u, j) \in Y$ and $j \notin J$. For this, we have

$$\begin{aligned} p^Y Q_k(Y) & \sum_{\mathcal{T} \in \text{GNS}(Y, k)} \prod_{(Y', k') \in \mathcal{T}} \mu(Y', k') \\ & \leq p^Y Q_k(Y) \left(1 + \sum_{Z \bowtie_k Y} \mu(Z, k) \right) \left(1 + \sum_{Z \sim u} \sum_k \mu(Z, \ell) \right) \prod_{i \sim Y, i \neq u} \left(1 + \sum_{Z \sim i} \sum_l \mu(Z, \ell) \right) \\ & \leq p^Y Q_k(Y) \frac{1}{1-S_k} \left(1 + \sum_{Z \sim u} \sum_k \mu(Z, \ell) \right) \prod_{i \sim Y, i \neq u} \lambda_i \quad (\text{by Proposition 3.9}) \\ & \leq \frac{p^Y Q_k(Y)}{1-S_k} \left(1 + \sum_{\ell} \sum_{j \notin J} \sum_{(u, j) \in Z} \frac{\alpha \lambda^Z Q_\ell(Z)}{1-S_\ell} \right) \prod_{i \sim Y, i \neq u} \lambda_i \\ & = \frac{p^Y Q_k(Y)}{1-S_k} \left(1 + \alpha \sum_{j \notin J} H_{u, j} \right) \prod_{i \sim Y, i \neq u} \lambda_i = \frac{\lambda^Y Q_k(Y)}{1-S_k} \times \frac{1 + \alpha \sum_{j \notin J} H_{u, j}}{\lambda_u}. \end{aligned}$$

Since $\mu(Y, k) = \alpha \frac{\lambda^Y Q_k(Y)}{1-S_k}$, it suffices to satisfy

$$\alpha \frac{\lambda^Y Q_k(Y)}{1-S_k} \geq \frac{\lambda^Y Q_k(Y)}{1-S_k} \times \frac{1 + \alpha \sum_{j \notin J} H_{u, j}}{\lambda_u}.$$

So, after a little algebra, it suffices to take:

$$\alpha = \frac{1}{\lambda_u - \sum_{j \notin J} H_{u, j}} = \frac{1}{\lambda_u - H_u + \sum_{j \in H} H_{u, j}}.$$

Note that $\lambda_u \geq 1 + \sum_j H_{u, j}$, so α is indeed in the range $[0, 1]$ as desired.

Corollary B.3 thus gives:

$$\begin{aligned} \Pr_{\text{MT}}(X_u \in J) & = \sum_{j \in J} p_{u, j} \sum_{\mathcal{T} \in \text{SNS}(E)} \prod_{Y \in \mathcal{T}} \sum_k \mu(Y, k) \\ & = \left(\sum_{j \in J} p_{u, j} \right) \left(1 + \sum_{\ell \notin J} \sum_{Y \ni (u, \ell)} \sum_k \alpha \frac{\lambda^Y Q_k(Y)}{1-S_k} \right) \\ & = \left(\sum_{j \in J} p_{u, j} \right) \left(1 + \alpha \sum_{\ell \notin J} H_{u, \ell} \right) = \left(\sum_{j \in J} p_{u, j} \right) \left(1 + \frac{\sum_{\ell \notin J} H_{u, \ell}}{\lambda_u - \sum_{\ell \notin J} H_{u, \ell}} \right) \\ & = \left(\sum_{j \in J} p_{u, j} \right) \left(\frac{\lambda_u}{\lambda_u - \sum_{\ell \notin J} H_{u, \ell}} \right) = \frac{\sum_{j \in J} \lambda_{u, j}}{\lambda_u - H_u + \sum_{j \in J} H_{u, j}}. \quad \square \end{aligned}$$

COROLLARY B.6. *Suppose that λ satisfies Theorem 3.8 and let $J \subseteq F_u$. Then:*

$$\Pr_{MT}(X_u \in J) \geq \frac{\sum_{j \in J} \lambda_{u,j} - \sum_{j \in J} H_{u,j}}{\lambda_u - \sum_{j \in J} H_{u,j}}.$$

PROOF. Apply Theorem B.5 to bound from above the probability of $X_u \in F_u - J$. \square

B.1 The MT Distribution and Independent Transversals

To illustrate our MT distribution results, let us consider *weighted* transversals, as discussed in Reference [1]. Suppose G has maximum degree Δ and each block of G has size exactly b . Given a weighting function $w : V \rightarrow [0, \infty)$, we may wish to find an independent transversal of minimum or maximum weight. Clearly, G has a transversal (not necessarily independent) of weight at most (respectively at least) $w(V)/b$; we would like to find an independent transversal whose weight is comparable to this.

One effective method to find weighted independent transversals is to find a *strong coloring* of G , which is a decomposition $V = I_1 \sqcup \dots \sqcup I_b$, wherein each I_i is an independent transversal of G . Clearly, given such a strong coloring of G , we can find in polynomial time an independent transversal I such that $w(I) \geq w(V)/b$ (respectively, $w(I) \leq w(V)/b$). When b is large compared to Δ , then such strong colorings exist and can even be found efficiently.

PROPOSITION B.7 ([10, 16, 20]). *When $b \geq \frac{11}{4}\Delta$ and $\Delta \geq \Delta_0$ for some constant Δ_0 , then a strong coloring of G exists. When $b \geq 5\Delta$, or when $b \geq 3\Delta + 1$ and Δ is constant, then a strong coloring of G can be found in randomized polynomial time.*

A more general method to find such weighted independent transversals is given in Reference [1] via *fractional strong colorings*. A fractional strong coloring of G is a probability distribution Ω over independent transversals I , with the property that any vertex v has $P_\Omega(v \in I) = 1/b$.

PROPOSITION B.8 ([1]). *When $b \geq 2\Delta$, there exists a fractional strong coloring of G . In particular, there exists an independent transversal I with $w(I) \geq w(V)/b$ (respectively, $w(I) \leq w(V)/b$).*

When $b \geq 4\Delta$, we can use the MT algorithm to construct independent transversals whose weight can be upper-bounded or lower-bounded in terms of $w(V)$. (When $b \geq 5\Delta$, then these follow immediately from Proposition B.7 already.) The two constructions are very similar, so we summarize them here together. We apply the PRA, in which there is a variable X_i corresponding to each block V_i ; we set $X_i = v$ to mean that $v \in V_i \cap I$. We will apply Theorem 3.8 by selecting a subset $B_i \subseteq V_i$ of size $|B_i| = r \geq 4\Delta$, and defining

$$\lambda_{i,v} = \alpha[v \in B_i] \quad \text{where } \alpha = \frac{r - \sqrt{r}\sqrt{r - 4\Delta}}{2r\Delta}.$$

We take as our set of bad events a single family \mathcal{B}_1 , which contains a separate atomic bad event corresponding to each edge. We also use the trivial hitting-set. With this choice, \varkappa_1 is null. So $\lambda_i = r\alpha$ and we have

$$G_i(Q, \lambda) = \sum_{v \in B_i} \sum_{\substack{\text{edges } f \\ \text{involving } v}} \alpha^2 \leq r\Delta\alpha^2.$$

Thus, Theorem 3.8 is satisfied. Furthermore, for any block i and $v \in B_i$ we have $H_{i,v} \leq \alpha^2\Delta$. We will show that, by selecting the sets B_i appropriately, the expected weight of the resulting independent transversal satisfies certain bounds. We can easily achieve an actual independent transversal whose weight is close to the expected weight by a polynomial number of repetitions.

THEOREM B.9. *Let G be a graph of maximum degree Δ whose vertex set is partitioned into blocks of size exactly b . If $4\Delta \leq b \leq 4.5\Delta$, then there is an efficient procedure to randomly sample an independent transversal I of G such that*

$$\mathbf{E}[w(I)] \geq w(V) \left(\frac{\sqrt{b} + \sqrt{b - 4\Delta}}{\sqrt{b}(2b - 1) + \sqrt{b - 4\Delta}} \right) \geq \frac{w(V)}{8\Delta - 1}.$$

If $4.5 \leq b \leq 5\Delta$, then there is an efficient procedure to randomly sample an independent transversal I of G such that

$$\mathbf{E}[w(I)] \geq \frac{4w(V)}{27\Delta}$$

PROOF. In the first result, we set $B_i = V_i$, of size $r = b$. Corollary B.6 gives

$$\Pr_{\text{MT}}(v \in I) \geq \frac{\lambda_{i,j} - H_{i,v}}{\lambda_i - H_{i,v}} \geq \frac{\alpha - \alpha^2\Delta}{b\alpha - \alpha^2\Delta} = \frac{\sqrt{b} + \sqrt{b - 4\Delta}}{\sqrt{b}(2b - 1) + \sqrt{b - 4\Delta}}$$

For the second result, sort the vertices in decreasing order of weight within block i as $v_{i,1}, v_{i,2}, \dots, v_{i,b}$ where $w(v_{i,1}) \geq w(v_{i,2}) \geq \dots \geq w(v_{i,b})$. We take $B_i = \{v_{i,1}, \dots, v_{i,r}\}$, where $r = \lceil 9\Delta/2 \rceil$. By Corollary B.6, for any block i we have

$$\Pr_{\text{MT}}(X_i = v_{i,j}) \geq \frac{\alpha - \alpha^2\Delta}{r\alpha - \alpha^2\Delta} \geq \frac{\alpha - \alpha^2\Delta}{(9\Delta/2 + 1/2)\alpha - \alpha^2\Delta}.$$

Routine algebraic calculations show that this expression is lower-bounded by $q = \frac{4}{27\Delta}$. Now consider some block V_i , and write $w_j = w(v_{i,j})$ for $j \leq b$. As $X_i \in B_i$ with probability 1, we have

$$\mathbf{E}[w(V_i \cap I)] = w_r + \sum_{j=1}^r \Pr(X_i = v_{i,j})(w_j - w_r) \geq w_r + \sum_{j=1}^r q(w_j - w_r).$$

Define $t = \sum_{j=1}^r w_j$. Since the vertices are in sorted order, $w_r \geq \frac{w(V_i) - t}{b - r}$ and so

$$\begin{aligned} \mathbf{E}[w(V_i \cap I)] &\geq w_r + q(t - rw_r) = w_r(1 - rq) + qt \\ &\geq \frac{w(V_i) - t}{b - r}(1 - rq) + qt = -t \frac{(1 - bq)}{b - r} + \frac{w(V_i)(1 - rq)}{b - r} \\ &\geq -w(V_i) \frac{(1 - bq)}{b - r} + \frac{w(V_i)(1 - rq)}{b - r} = qw(V_i) \quad \text{since } t \leq w(V_i) \text{ and } bq \leq 1. \end{aligned}$$

The result follows by linearity of expectation, summing over all blocks i . \square

THEOREM B.10. *Let G be a graph of maximum degree Δ whose vertex set is partitioned into blocks of size at least b . If $b \geq 4\Delta$, then there is an efficient procedure to randomly sample an independent transversal I of G such that*

$$\mathbf{E}[w(I)] \leq w(V) \frac{2}{b + 4\sqrt{(b - 4\Delta)\Delta}}.$$

PROOF. We may assume $b \leq 5\Delta$, as otherwise this follows from Proposition B.7.

Sort the vertices in increasing order of weight, so that in each block i we have $w(v_{i,1}) \leq w(v_{i,2}) \leq \dots \leq w(v_{i,b})$. As before, let us write $X_i = j$ as shorthand for $X_i = v_{i,j}$. In this case, we apply our construction with $r = 4\Delta$ and $B_i = \{v_{i,1}, \dots, v_{i,r}\}$

Let us fix a block i , and write $w_j = w(v_{i,j})$ for $j \leq b$. Then

$$w(V_i \cap I) = w_1 + (w_2 - w_1)[X_i \geq 2] + (w_3 - w_2)[X_i \geq 3] + \dots + (w_r - w_{r-1})[X_i = r]. \quad (15)$$

We now have $H_{i,v} \leq \alpha^2 \Delta = \frac{1}{4\Delta}$. So for any $j \geq 1$, Corollary B.5 gives $\Pr(X_i \geq j) \leq \frac{(r-j+1)\alpha}{r\alpha - (j-1)\frac{1}{4\Delta}} = \frac{2(r-j+1)}{2r-j+1} =: q_j$. With this notation, taking the expectation of Equation (15) gives

$$\mathbf{E}[w(V_i \cap I)] \leq w_1 q_1 + (w_2 - w_1) q_2 + (w_3 - w_2) q_3 + \cdots + (w_r - w_{r-1}) q_r.$$

Let us define $u_j = w_j - w_{j-1} \geq 0$ for $j \geq 2$, and $u_1 = w_1$. Noting that $\sum_j w_j = \sum_j u_j (b - j + 1)$, we can write this as:

$$\begin{aligned} \mathbf{E}[w(V_i \cap I)] &\leq \sum_{j=1}^r q_j u_j = \sum_{j=1}^r u_j (b - j + 1) \times \frac{2(r - j + 1)}{(2r - j + 1)(b - j + 1)} \\ &\leq w(V_i) \max_{j \in [r]} \frac{2(r - j + 1)}{(2r - j + 1)(b - j + 1)} \leq w(V_i) \max_{x \in [0, r-1]} \frac{2(r - x)}{(b - x)(2r - x)}. \end{aligned}$$

We can view this expression $f(x) = \frac{2(r-x)}{(b-x)(2r-x)}$ as a differentiable function of x , which has critical points at $x = r \pm \sqrt{br - r^2}$. As $b \leq 8\Delta$, the function $f(x)$ achieves its maximum value at $x = r - \sqrt{br - r^2}$, and we have there

$$f(x) = \frac{2}{b + 2\sqrt{(b-r)r}} = \frac{2}{b + 4\sqrt{(b-4\Delta)\Delta}}.$$

So $\mathbf{E}[w(V_i \cap I)] \leq \frac{2w(V_i)}{b + 4\sqrt{(b-4\Delta)\Delta}}$. The result follows by linearity of expectation, summing over all blocks i . \square

C ALGORITHMICALLY IMPLEMENTING THE HITTING SET FOR THEOREM 4.5

To implement the PRA using the fractional hitting-set Q of Theorem 4.5, we must efficiently sample from Q , in the following sense: given an atomic bad event B on elements x_1, \dots, x_k with weights a_1, \dots, a_k , we must select a subset $Y \subseteq B$ with probability proportional to $Q(Y)$. Note that implemented naively this step might take $\binom{n}{d}$ time, which is potentially exponential.

For any set $W \subseteq B$, define

$$R(W) = \sum_{\substack{Y: W \subseteq Y \subseteq B \\ |Y|=d}} Q(Y).$$

This can be evaluated in time $O(dk)$ using a dynamic program. To efficiently sample $Y \subseteq B$ with probability proportional to $Q(Y)$, use the following procedure:

- 1: Let $Y_0 = \emptyset$
- 2: **for** $i = 1, \dots, d$ **do**
- 3: **for each** $j \in B - Y_{i-1}$ **do** compute $q_j = R(Y_{i-1} \cup \{j\})$.
- 4: Select some $j \in B - Y_{i-1}$ with probability proportional to q_j .
- 5: Set $Y_i = Y_{i-1} \cup \{j\}$.
- 6: **return** Y_d

PROPOSITION C.1. *For any $Z \subseteq B$ with $|Z| = d$, we have $\Pr(Y_d = Z) = \frac{Q(Z)}{\sum_{Y \subseteq B} Q(Y)}$.*

PROOF. We show by induction on i the following: for any sets $W \subseteq Z \subseteq B$ with $|W| = i, |Z| = d$, and $0 \leq i \leq d$, we have

$$\Pr(Y_d = Z \mid Y_i = W) = \frac{Q(Z)}{R(W)}.$$

Applying this with $i = 0, W = \emptyset$ will give us the desired result. Also, the induction case with $i = d$ is trivially true. For the induction step:

$$\begin{aligned} \Pr(Y_d = Z \mid Y_i = W) &= \frac{\sum_{z \in Z-W} R(W \cup \{z\}) \Pr(Y_d = Z \mid Y_{i+1} = W + z)}{\sum_{x \in B-W} R(W \cup \{x\})} \\ &= \frac{\sum_{z \in Z-W} R(W \cup \{z\}) Q(Z)/R(W + z)}{\sum_{x \in B-W} R(W \cup \{x\})} \quad \text{induction hypothesis} \\ &= \frac{Q(Z)(d-i)}{\sum_{x \in B-W} \sum_{Y: W \cup \{x\} \subseteq Y \subseteq Z} Q(Y)} \\ &= \frac{Q(Z)(d-i)}{\sum_{Y: W \subseteq Y \subseteq Z} \sum_{x \in Y-W} Q(Y)} = \frac{Q(Z)(d-i)}{R(W)(d-i)} = \frac{Q(Z)}{R(W)}, \end{aligned}$$

thus completing the induction. \square

D FUNCTIONAL ANALYSIS FOR THEOREM 4.7

We prove that the vector b_k given in Theorem 4.7 satisfies the stated properties (C1) and (C2).

PROPOSITION D.1. *Let $c \geq 1$, let $D \geq 2$, and let ϵ satisfy $0 < \epsilon \leq 1/D$. Then the quantity b given below satisfies properties (C1) and (C2):*

$$b = \begin{cases} \frac{100 \ln(1/\epsilon)}{1 + \ln(\frac{\ln(1/\epsilon)}{c})} & \text{for } c \leq \ln(1/\epsilon) \\ c(1 + \epsilon) + 10\sqrt{c \ln(D + \frac{1}{c\epsilon^2})} & \text{for } c > \ln(1/\epsilon) \end{cases}.$$

PROOF. Let $\delta = \ln(1/\epsilon)$ and let $d = \ln D$. As $D \geq 2$, we have $\epsilon \leq 1/2$ and hence $\delta \geq \ln 2$. We also write $\mu = c(1 + \epsilon)$. It is immediately clear that $b \geq \mu$.

Case I: $c \leq \delta$. Let $x = \delta/c \geq 1$, so $b = \frac{100\delta}{1 + \ln x}$. We estimate $\text{Ch}(\mu, b)$ as:

$$\text{Ch}(\mu, b) = e^{b-c(1+\epsilon)} \left(\frac{c(1+\epsilon)}{b} \right)^b \leq e^b \left(\frac{2c}{b} \right)^b = \left(\frac{2e(\delta/x)(1 + \ln x)}{100\delta} \right)^{100\delta/(1+\ln x)} = \left(\frac{e(1 + \ln x)}{50x} \right)^{\frac{100\delta}{1+\ln x}}.$$

Simple calculus shows that, for $x \geq 1$, we have $(\frac{e(1+\ln x)}{50x})^{1/(1+\ln x)} \leq e^{e/50-1} \leq 0.389$. So

$$\text{Ch}(\mu, b) \leq (0.389)^{100\delta} \leq (0.389)^{100 \ln 2} \leq 3.78 \times 10^{-29} \leq 1/2.$$

So (C2) is satisfied. To show (C1):

$$\left(\frac{b+1}{c(1+\epsilon)} - 1 \right) \text{Ch}(\mu, b) \times 4D/\epsilon \leq \left(\frac{2b}{1} \right) \text{Ch}(\mu, b) \times 4\epsilon^{-2} \leq 800\delta \text{Ch}(\mu, b) \times \epsilon^{-2} \leq \delta(0.389)^{100\delta} \times e^{2\delta}.$$

Simple calculus shows that, for $\delta \geq \ln 2$, this attains a maximum value of 1.048×10^{-28} , which is attained at $\delta = \ln 2$. In particular, it is smaller than 1.

Case II: $c > \delta$. Let $v = D + \epsilon^{-2}/c$; clearly $D \leq v$, and as $\epsilon \leq 1/D$ we also have the crude bound $v \leq \epsilon^{-1} + \epsilon^{-2}/\delta \leq \epsilon^{-3}$. With this notation, we have $b = \mu + 10\sqrt{c \ln v}$. The relative deviation between μ and b here is given by $\lambda = b/\mu - 1 = 10\frac{\sqrt{c \ln v}}{c(1+\epsilon)}$. We observe the following bound on the size of λ :

$$\lambda = \frac{10\sqrt{\ln v}}{\sqrt{c}(1+\epsilon)} \leq \frac{10\sqrt{\ln(\epsilon^{-3})}}{\sqrt{\delta}} = 10\sqrt{3} \leq 17.4.$$

Since $\lambda \leq 17.4$, a simple calculation shows that $\text{Ch}(\mu, \mu(1 + \lambda)) \leq e^{-\mu\lambda^2/10}$; thus, $\text{Ch}(\mu, b) \leq v^{-\frac{10}{1+\epsilon}} \leq v^{-6.6}$. As $v \geq D \geq 2$, this is at most 0.0104, and thus (C1) is satisfied. To show (C2):

$$\begin{aligned} \left(\frac{b+1}{c(1+\epsilon)} - 1 \right) \text{Ch}(c(1+\epsilon), b) &\leq \left(\frac{c(1+\epsilon) + 10\sqrt{c \ln v} + 1 - c(1+\epsilon)}{c(1+\epsilon)} \right) \times v^{-6.6} \\ &= \left(\frac{10\sqrt{\ln v} + 1/\sqrt{c}}{\sqrt{c}(1+\epsilon)} \right) \times v^{-6.6} \\ &\leq 5v^{-5.6}c^{-1/2} \quad \text{as } 10\sqrt{\ln v} + 1 \leq 5v \text{ for } v \geq D \geq 2 \\ &= 5(D + \epsilon^{-2}/c)^{-5.6}c^{-1/2}. \end{aligned}$$

Simple analysis shows that this quantity $5(D + \epsilon^{-2}/c)^{-5.6}c^{-1/2}$, as a function of c , has a maximum value at $c = 10.2/(D\epsilon^2)$, at which point we have

$$5(D + \epsilon^{-2}/c)^{-5.6}c^{-1/2} \leq \frac{0.927277\epsilon}{D^{5.1}}.$$

Therefore, as $D \geq 2$, we have shown that

$$\left(\frac{b+1}{c(1+\epsilon)} - 1 \right) \text{Ch}(c(1+\epsilon), b) \leq \frac{0.927277\epsilon}{D^{5.1}} \leq \frac{\epsilon}{4D},$$

satisfying (C2). □

E FULL PROOF OF THEOREM 7.10

In this section, we extend the proof of Theorem 7.10 to cover the case when $C + D \leq 2^{896}$.

PROPOSITION E.1. *Suppose the original problem instance has congestion C and dilation D . Let i', C', d be positive integers with $d \leq C'$, and let $\beta > 1$. Define*

$$p = \frac{(i'\beta)^d}{d! \binom{C'+1}{d}}.$$

If $p < 1$ and $\beta - \frac{Ddp}{1-p} \geq 1$, then there is a schedule S of length $L \leq C + D$, in which every interval of length i' has congestion at most C' . Furthermore, S can be found in polynomial time.

PROOF. We add delays in the range $b = \{0, \dots, C-1\}$ uniformly to each packet. In this case, we have a variable corresponding to each packet x , and for each delay t we assign $\lambda_{x,t} = \beta/C$. For each edge f and i' -interval I , we have a complex bad event $\mathcal{B}_{f,I}$ that the congestion in the interval exceeds C' . We use the fractional hitting-set $Q_{f,I}$ with parameter d as described in Theorem 4.5.

For a given f, I , we first must compute $\mu_{f,I}$, which is the sum of $\lambda_{x,t}$ over all packets x and delays t contributing to congestion for f, I . There are at most C packets which pass through f , and at most i' choices for t cause a transit of f within I . So $\mu_{f,I} \leq Ci'\beta/C = i'\beta$. The bad event is that this is at least $C' + 1$. By Theorem 4.5, this gives $S_{f,I} = S(\mathcal{B}_{f,I}, Q_{f,I}, \lambda) \leq \frac{\mu_{f,I}^d}{d! \binom{C'+1}{d}} \leq p$.

Let $\mu_{x,f,I}$ be the sum of $\lambda_{x,t}$ over all delays t contributing to congestion on edge f in interval I . By Theorem 4.5 we have

$$\sum_{f,I} G_x(Q_{f,I}, \lambda) \leq \sum_{f,I} \frac{\mu_{x,f,I}}{\mu_{f,I}} \times \frac{d\mu_{f,I}^d}{d! \binom{C'+1}{d}} \leq \sum_{f,I} \frac{dp\mu_{x,f,I}}{i'\beta}.$$

For each of the D edges traversed by x , a choice of delay t contributes to congestion within i' separate intervals. So $\sum_{f,I} \mu_{x,f,I} \leq C \times \beta/C \times D \times i' = Di'\beta$. So

$$\sum_{f,I} \frac{G_x(Q_{f,I}, \lambda)}{1 - S_{f,I}} \leq \frac{p \times Di'\beta}{i'\beta(1-p)} = \frac{Ddp}{1-p}.$$

To apply Theorem 3.8, each packet x must satisfy the constraint

$$\lambda_x \geq 1 + \sum_{f,I} \frac{G_x(Q_{f,I}, \lambda)}{1 - S_{f,I}}. \quad (16)$$

We have $\lambda_x = C\beta/C = \beta$ and so Equation (16) becomes $\beta \geq 1 + \frac{Ddp}{1-p}$, which is precisely the constraint specified in the hypothesis. \square

PROPOSITION E.2. *Suppose the original problem instance has congestion C and dilation D with $C + D \leq X = 100,000$. Then there is a schedule of length $L \leq C + D$, which satisfies the 4IC with respect to $T = 6, T' = 5$, which can be produced in polynomial time.*

PROOF. For each packet x we add a random delay in the interval $\{0, \dots, C - 1\}$. For each edge f , and each 4-interval I starting at time s for odd integer s , we have a complex bad event $\mathcal{B}_{f,I}$ that

$$\text{OF}^+(c_s(f), c_{s+1}(f); T) + \text{OF}^-(c_{s+2}(f), c_{s+3}(f); T) > T'.$$

The overall proof here is very similar to Proposition 7.9: We will apply the PRA using separate variable for each packet (the value of a variable is the chosen delay), and the vector $\vec{\lambda}_{x,t} = \alpha = \beta/C$, where $\beta = 1.430599$. We define $\Phi_0, \Phi_1, \Phi_2, \Phi_3, \Phi$ as in Proposition 7.9.

Thus again $S_{f,I} \leq \sum_Y Q_{f,I}(Y)\lambda^Y \leq \Phi \leq \hat{\Phi}$. For any packet x and delay t and $j = 0, \dots, 3$, there are at most $D/2$ pairs f, I for which packet x, t has type j . Each such f, I has $G_{x,t}(Q_{f,I}, \lambda) \leq \Phi_j$. Summing over all C values of t and all such f, I gives

$$\sum_{f,I} G_x(Q_{f,I}, \lambda) \leq \frac{CD}{2} (\Phi_0 + \Phi_1 + \Phi_2 + \Phi_3) \leq \frac{CX}{2} (\hat{\Phi}_0 + \hat{\Phi}_1 + \hat{\Phi}_2 + \hat{\Phi}_3).$$

To apply Theorem 3.8, each packet x must satisfy the constraint

$$\lambda_x \geq 1 + \sum_{f,I} \frac{G_x(Q_{f,I}, \lambda)}{1 - S_{f,I}}.$$

We have $\lambda_x = \beta$, and so it suffices to satisfy the condition

$$\beta - X/2 \times \frac{C(\hat{\Phi}_0 + \hat{\Phi}_1 + \hat{\Phi}_2 + \hat{\Phi}_3)}{1 - \hat{\Phi}} \geq 1. \quad (17)$$

We now observe that the left-hand side of Equation (17) is a decreasing function of C . To see this, note that for instance we have

$$C\hat{\Phi}_0 = C \sum_{y_0, y_1, y_2, y_3} \binom{C-1}{y_0-1} \binom{C}{y_1} \binom{C}{y_2} \binom{C}{y_3} b(y_0, y_1, y_2, y_3) (\beta/C)^{y_0+y_1+y_2+y_3}, \quad (18)$$

which is an increasing function of C . A similar argument holds for $C\hat{\Phi}_1, C\hat{\Phi}_2, C\hat{\Phi}_3, \Phi$.

Because of this fact, it suffices to satisfy Equation (17) at $C = X$. We use a computer code similar to Proposition 7.9 to select b in this case. Overall, we get a bound

$$\frac{\hat{\Phi}_0 + \hat{\Phi}_1 + \hat{\Phi}_2 + \hat{\Phi}_3}{1 - \hat{\Phi}} \leq 8.16 \times 10^{-11},$$

which satisfies Equation (17). \square

THEOREM E.3. *Suppose $2 \leq C + D < 2^{896}$. Then there is a feasible schedule of makespan at most $6.73(C + D)$.*

PROOF. Case I: $2^{32} \leq C + D < 2^{896}$.

Apply Proposition E.1 with $\beta = 1.00563$, $C' = 17040600$, $i' = 2^{24}$, $d = 250000$ to produce a schedule S_1 of length $L_1 \leq C + D$, in which each interval of length 2^{24} has congestion at most 17040600.

Apply Proposition 7.5 to S_1 with $\alpha = 5.98328 \times 10^{-8}$, $C' = 1320$, $d = 270$, $m = 400$ to get a schedule S_2 of length $L_2 \leq 1.0025L_1 + 2^{24}$, in which each interval of length 1024 has congestion at most 1320.

Apply Proposition 7.9 to S_2 to get a schedule S_3 of length $L_3 \leq 1.02779L_2 + 1024$ satisfying the 4IC with $T = 6$, $T' = 5$. By Proposition 7.8, this yields a feasible schedule S_4 whose makespan is $L_4 \leq 6.5L_3 + 4.5$. As $C + D \geq 2^{32}$, we have $L_4 \leq 6.73(C + D)$.

Case II: $150,000 \leq C + D < 2^{32}$. Apply Proposition E.1 with $\beta = 1.01831$, $C' = 1320$, $i' = 1024$, $d = 280$ to produce a schedule S_1 of length $L_1 = C + D$, in which each interval of length 1024 has congestion at most 1320.

Apply Proposition 7.9 to S_1 to get a schedule S_2 of length $L_2 \leq 1.02779L_1 + 1024$ satisfying the 4IC with $T = 6$, $T' = 5$. By Proposition 7.8, this yields a feasible schedule S_3 whose makespan is $L_3 \leq 6.5L_2 + 4.5$. As $C + D \geq 2^{17}$, we have $L_3 \leq 6.73(C + D)$.

Case III: $2^{15} \leq C + D < 150,000$. Apply Proposition E.1 with $\beta = 1.00202$, $C' = 675$, $i' = 512$, $d = 140$ to produce a schedule S_1 of length $L_1 = C + D$, in which each interval of length 512 has congestion at most 675.

We now apply a slight variant of Proposition 7.9 to S_1 , using parameters $m = 51$, $C = 675$, $i = 512$, $T = 6$, $T' = 5$, $\alpha = 0.002113$ to produce a schedule of length $L_2 = (1 + 1/51)L_1 + 512$, which satisfies the 4IC for $T = 6$, $T' = 5$. By Proposition 7.8, this yields a feasible schedule S_3 whose makespan is $L_3 \leq 6.5L_2 + 4.5$. As $C + D \geq 2^{15}$, we have $L_3 \leq 6.73(C + D)$.

Case IV: $20 \leq C + D \leq 2^{15}$. Apply Proposition E.2 to produce a schedule S_1 of length $L_1 \leq C + D$ satisfying the 4IC for $T = 6$, $T' = 5$. By Proposition 7.8, this yields a feasible schedule S_2 whose makespan is $L_2 \leq 6.5L_1 + 4.5$. As $C + D \geq 20$, this is at most $6.73(C + D)$.

Case V: $2 \leq C + D \leq 19$. Apply Proposition E.1 with $\beta = 1.27877$, $C' = 9$, $i' = 2$, $d = 8$ to produce a schedule S_1 of length $L_1 \leq C + D$, in which each interval of length 2 has congestion at most 9. By Proposition 7.1, this yields a feasible schedule of length $\lceil L_1/2 \rceil(9 + 1) \leq 5(C + D) + 5/2 \leq 6.25(C + D)$. \square

ACKNOWLEDGMENTS

We thank Tom Leighton and Satish Rao for valuable discussions that served as the foundation for this work. We are thankful to Noga Alon, Venkatesan Guruswami, Bernhard Haeupler, Penny Haxell, and Jeff Kahn for helpful discussions, as well as to the STOC 2013, FOCS 2013, and journal referees for their valuable comments.

REFERENCES

- [1] R. Aharoni, E. Berger, and R. Ziv. 2007. Independent systems of representatives in weighted graphs. *Combinatorica* 27, 3 (2007), 253–267.
- [2] N. Alon. 1988. The linear arboricity of graphs. *Isr. J. Math.* 62, 3 (1988), 311–325.
- [3] N. Alon. 1992. The strong chromatic number of a graph. *Rand. Struct. Algor.* 3, 1 (1992), 1–7.
- [4] Y. Azar and A. Epstein. 2005. Convex programming for scheduling unrelated parallel machines. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC'05)* 331–337.
- [5] R. Bissacot, R. Fernandez, A. Procacci, and B. Scoppola. 2011. An improvement of the Lovász Local Lemma via cluster expansion. *Combin. Probab. Comput.* 20–5 (2011), 709–719.
- [6] B. Bollobás, P. Erdős, and E. Szemerédi. 1975. On complete subgraphs of r -chromatic graphs. *Discr. Math.* 13, 2 (1975), 97–107.

- [7] A. Chen, D. Harris, and A. Srinivasan. 2016. Partial resampling to approximate covering integer programs. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'16)*. 1984–2003.
- [8] D. Dubhashi and D. Ranjan. 1996. Balls and bins: A study in negative dependence. *Random Structures & Algorithms* 13, 2 (1998), 99–124.
- [9] P. Erdős and L. Lovász. 1975. Problems and results on 3-chromatic hypergraphs and some related questions. In *Infinite and Finite Sets*, Vol. 11 of Colloq. Math. Soc. J. Bolyai. 609–627. North-Holland.
- [10] Alessandra Graf and Penny Haxell. 2018. Finding independent transversals efficiently. *arXiv preprint arXiv:1811.02687*.
- [11] B. Haeupler, B. Saha, and A. Srinivasan. 2011. New constructive aspects of the Lovász Local Lemma. *J. ACM* 58, 6 (2011).
- [12] D. Harris. 2016. New bounds for the Moser-Tardos distribution. *arXiv preprint arXiv:1610.09653*.
- [13] D. Harris and A. Srinivasan. 2013. Constraint satisfaction, packet routing, and the Lovász Local Lemma. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC'13)*. 685–694.
- [14] D. Harris and A. Srinivasan. 2013. The Moser-Tardos framework with partial resampling. In *Proceedings of the IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS'13)*. 469–478.
- [15] D. Harris and A. Srinivasan. 2017. Algorithmic and enumerative aspects of the Moser-Tardos distribution. *ACM Trans. Algor.* 13, 3 (2017), 33.
- [16] D. Harris and A. Srinivasan. 2017. A constructive Lovász Local Lemma for permutations. *Theory Comput.* 13, 17 (2017), 1–41.
- [17] N. J. A. Harvey. 2015. A note on the discrepancy of matrices with bounded row and column sums. *Discr. Math.* 338, 4 (2015), 517–521.
- [18] P. Haxell and T. Szabó. 2006. Odd independent transversals are odd. *Combin. Probab. Comput.* 15, 1–2 (2006), 193–211.
- [19] P. E. Haxell. 2001. A note on vertex list colouring. *Combin. Probab. Comput.* 10, 4 (2001), 345–348.
- [20] P. E. Haxell. 2008. An improved bound for the strong chromatic number. *J. Graph Theory* 58, 2 (2008), 148–158.
- [21] P. E. Haxell, T. Szabó, and G. Tardos. 2003. Bounded size components—Partitions and transversals. *J. Combin. Theory B* 88, 2 (2003), 281–297.
- [22] G. Jin. 1992. Complete subgraphs of r -partite graphs. *Combin. Probab. Comput.* 1, 3 (1992), 241–250.
- [23] R. M. Karp, F. T. Leighton, R. L. Rivest, C. D. Thompson, U. V. Vazirani, and V. V. Vazirani. 1987. Global wire routing in two-dimensional arrays. *Algorithmica* 1, 1–4 (1987), 113–129.
- [24] F. T. Leighton, C.-J. Lu, S. B. Rao, and A. Srinivasan. 2001. New algorithmic aspects of the local lemma with applications to routing and partitioning. *SIAM J. Comput.* 31, 2 (2001), 626–641.
- [25] F. T. Leighton, B. M. Maggs, and S. B. Rao. 1994. Packet routing and jobshop scheduling in $O(\text{congestion} + \text{dilation})$ steps. *Combinatorica* 14, 2 (1994), 167–186.
- [26] J. K. Lenstra, D. B. Shmoys, and É. Tardos. 1990. Approximation algorithms for scheduling unrelated parallel machines. *Math. Program.* 46, 1–3 (1990), 259–271.
- [27] P.-S. Loh and B. Sudakov. 2007. Independent transversals in locally sparse graphs. *J. Combin. Theory B* 97, 6 (2007), 904–918.
- [28] R. Moser and G. Tardos. 2010. A constructive proof of the general Lovász Local Lemma. *J. ACM* 57, 2 (2010), 1–15.
- [29] B. Peis and A. Wiese. 2011. Universal packet routing with arbitrary bandwidths and transit times. In *Proceedings of the 15th International Conference on Integer Programming and Combinatorial Optimization (IPCO'11)*. 362–375.
- [30] P. Raghavan and C. D. Thompson. 1987. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica* 7, 4 (1987), 365–374.
- [31] T. Rothvoß. 2013. A simpler proof for $O(\text{congestion} + \text{dilation})$ packet routing. In *Proceedings of the 16th International Conference on Integer Programming and Combinatorial Optimization (IPCO'13)*. 336–348.
- [32] C. Scheideler. 1998. Universal routing strategies for interconnection networks. In *Lecture Notes in Computer Science*, Vol. 1390. Springer.
- [33] J. P. Schmidt, A. Siegel, and A. Srinivasan. 1995. Chernoff-Hoeffding bounds for applications with limited independence. *SIAM J. Discr. Math.* 8, 2 (1995), 223–250.
- [34] M. Singh. 2008. *Iterative Methods in Combinatorial Optimization*. Ph.D. Dissertation, Tepper School of Business, Carnegie-Mellon University.
- [35] T. Szabó and G. Tardos. 2006. Extremal problems for transversals in graphs with bounded degree. *Combinatorica* 26, 3 (2006), 333–351.
- [36] R. Yuster. 1997. Independent transversals in r -partite graphs. *Discr. Math.* 176, 1–3 (1997), 255–261.

Received June 2014; revised October 2018; accepted June 2019