# AN ALGORITHMIC PROOF OF THE LOVÁSZ LOCAL LEMMA VIA RESAMPLING ORACLES*

NICHOLAS J. A. HARVEY† AND JAN VONDRÁK‡

**Abstract.** The Lovász Local Lemma is a seminal result in probabilistic combinatorics. It gives a sufficient condition on a probability space and a collection of events for the existence of an outcome that simultaneously avoids all of those events. Finding such an outcome by an efficient algorithm has been an active research topic for decades. Breakthrough work of Moser and Tardos (2009) presented an efficient algorithm for a general setting primarily characterized by a product structure on the probability space.

In this work we present an efficient algorithm for a much more general setting. Our main assumption is that there exist certain functions, called *resampling oracles*, that can be invoked to address the undesired occurrence of the events. We show that, in *all* scenarios to which the original Lovász Local Lemma applies, there exist resampling oracles, although they are not necessarily efficient. Nevertheless, for essentially all known applications of the Lovász Local Lemma and its generalizations, we have designed efficient resampling oracles. As an application of these techniques, we present a new result on packings of rainbow spanning trees.

**Key words.** Lovász Local Lemma, randomized algorithms, general probability spaces.

**AMS subject classifications.** 68Q87, 68R05, 68W20

## 1. Introduction.

The Lovász Local Lemma (LLL) is a powerful tool with numerous uses in combinatorics and theoretical computer science. If a given probability space and collection of events satisfy a certain condition, then the LLL asserts the existence of an outcome that simultaneously avoids those events. The classical formulation of the LLL [15, 46] is as follows.

Let $\Omega$ be a probability space with probability measure $\mu$. Let $E_1, \ldots, E_n$ be certain "undesired" events in that space. Let $G$ be a graph with vertex set $[n] = \{1, \ldots, n\}$. The edges of $G$ are denoted $E(G)$. For the purposes of this paper we will assume that $\Omega$ is finite and that $G$ is undirected, but these assumptions are not actually necessary in the classical formulation. The notation $i \backsim j$ denotes that $\{i, j\} \in E(G)$ and $i \neq j$. The neighbors of vertex $i$ are $\Gamma(i) = \{\, j : i \backsim j \,\}$. Let $\Gamma^+(i) = \Gamma(i) \cup \{i\}$ and let $\Gamma^+(I) = \bigcup_{i \in I} \Gamma^+(i)$ for $I \subseteq [n]$.

THEOREM 1 (General Lovász Local Lemma [15, 46]). *Suppose that the events satisfy the following condition that controls their dependences*

$$\text{(Dep)} \qquad \Pr_\mu[E_i \mid \cap_{j \in J} \overline{E_j}] \;=\; \Pr_\mu[E_i] \qquad \forall i \in [n],\, J \subseteq [n] \backslash \Gamma^+(i)$$

*and the following criterion that controls their probabilities*

$$\text{(GLL)} \qquad \exists x_1, \ldots, x_n \in (0,1) \quad \textit{such that} \quad \Pr_\mu[E_i] \;\leqslant\; x_i \prod_{j \in \Gamma(i)} (1 - x_j) \qquad \forall i \in [n].$$

*Then* $\Pr_\mu[\bigcap_{i=1}^n \overline{E_i}] > 0$.

An equivalent statement of (Dep) is that the event $E_i$ must be independent of the sigma-algebra generated by the events $\{\, E_j : j \notin \Gamma^+(i) \,\}$. When (Dep) holds, $G$

*Submitted January 2018.

**Funding:** N. Harvey was supported by an NSERC Discovery Grant.

†University of British Columbia, Vancouver, Canada. (nickhar@cs.ubc.ca).

‡Stanford University, CA, USA. (jvondrak@stanford.edu).

is called a *dependency graph*. The literature contains several dependency conditions generalizing (Dep) and criteria generalizing (GLL) under which the conclusion of the theorem remains true. We will discuss several such generalizations below.

**Algorithms.** Algorithms to efficiently find an outcome in $\bigcap_{i=1}^{n} \overline{E_i}$ have been the subject of research for several decades. In 2008, a nearly optimal result was obtained by Moser [34] for a canonical application of the LLL, the bounded-degree $k$-SAT problem. Shortly thereafter, Moser and Tardos [35] extended that result to a general scenario called the "variable model" in which $\Omega$ consists of independent variables, each $E_i$ depends on a subset of the variables, and events $E_i$ and $E_j$ are adjacent in $G$ if there is a variable on which they both depend. Clearly the resulting graph is a dependency graph. The Moser-Tardos algorithm is extremely simple: after drawing an initial sample of the variables, it repeatedly checks if any undesired event occurs, then *resamples* any such event. Resampling an event means that the variables on which it depends receive fresh samples according to $\mu$; here, the independence of the variables is crucial. Moser and Tardos prove that, if the (GLL) condition is satisfied, this algorithm will produce the desired outcome after at most $\sum_{i=1}^{n} \frac{x_i}{1-x_i}$ resampling operations, in expectation.

Numerous extensions of the Moser-Tardos algorithm have been proposed. These extensions can handle more general criteria [28, 38, 1, 29], derandomization [12], exponentially many events [22], distributed scenarios [13], etc. However, these results are restricted to the Moser-Tardos variable model and hence cannot be viewed as algorithmic proofs of the LLL in full generality. There are many known scenarios for the LLL and its generalizations that fall outside the scope of the variable model [31, 32]. One example scenario, random spanning trees in complete graphs, is discussed in Section 3.

Recently, some efficient algorithms have been developed that go beyond the variable model. This related work is discussed in Section 1.5.

**1.1. Our contributions.** The primary motivating question for this work is whether there is an "algorithmic proof" of the Lovász Local Lemma in general probability spaces. We answer this question in the following sense: We propose an algorithmic framework for the general Lovász Local Lemma, based on a new notion of *resampling oracles*. In this framework, we present an algorithm that finds a point in $\bigcap_{i=1}^{n} \overline{E_i}$ (avoiding all undesired events) efficiently, if given access to three types of subroutines outlined below, the most crucial one being resampling oracles. The *existence* of such subroutines, ignoring efficiency considerations, turns out to be a consequence of the Lovász Local Lemma's assumptions. Whether these subroutines can be implemented *efficiently* is an instance-dependent issue, as discussed further below.

Thus, using the existence of these (potentially inefficient) subroutines, our algorithm provides a new proof of the existential LLL as formulated in Theorem 1, and even of several more general formulations. Algorithmically, we reduce the problem of finding a point in $\bigcap_{i=1}^{n} \overline{E_i}$ to the problem of implementing the three subroutines that we discuss next.

**1.1.1. Algorithmic assumptions.** In order to discuss algorithms for the LLL in full generality, one must assume some form of access to the probability space at hand. It is natural to assume that one can efficiently sample from $\mu$, and efficiently check whether a given event $E_i$ occurs. However, even under these assumptions, finding the desired output can be computationally hard; this result is established in

Section 2.2. Therefore, our framework assumes the existence of one more subroutine that can be used by our algorithm. This leads us to the notion of resampling oracles.

Let us introduce some notation. An atomic event $\omega$ in the probability space $\Omega$ will be called a *state*. We write $\omega \sim \mu$ to denote that a random state $\omega$ is distributed according to $\mu$, and $\omega \sim \mu|_{E_i}$ to denote that the distribution is $\mu$ conditioned on $E_i$. The resampling oracles are defined with respect to a graph $G$ on $[n]$ with neighborhood structure $\Gamma$, not necessarily satisfying the (Dep) condition.

The three subroutines required by our algorithm are as follows.

- *Sampling from $\mu$:* There is a subroutine that provides an independent random state $\omega \sim \mu$.
- *Checking events:* For each $i \in [n]$, there is a subroutine that determines whether $\omega \in E_i$.
- *Resampling oracles:* For each $i \in [n]$, there is a randomized subroutine $r_i : \Omega \to \Omega$ with the following properties.
  - (R1) If $E_i$ is an event and $\omega \sim \mu|_{E_i}$, then $r_i(\omega) \sim \mu$. (The oracle $r_i$ removes conditioning on $E_i$.)
  - (R2) For any $j \notin \Gamma^+(i)$, if $\omega \notin E_j$ then also $r_i(\omega) \notin E_j$. (Resampling an event cannot cause new non-neighbor events to occur.)

  When these conditions hold, we say that $r_i$ is a resampling oracle for events $E_1, \dots, E_n$ and graph $G$.

If efficiency concerns are ignored, the first two subroutines trivially exist. It turns out that (possibly inefficient) resampling oracles exist if and only if a certain relaxation of (Dep) holds; this result is established in Section 1.3.

**Main Result.** Our main result is an algorithm that uses the three subroutines to find a point in $\bigcap_{i=1}^n \overline{E_i}$. This algorithm is efficient whenever the three subroutines have efficient implementations.

THEOREM 2 (Informal). *Consider any probability space, any events $E_1, \dots, E_n$, and any undirected graph $G$ on vertex set $[n]$. If (GLL) is satisfied and if the three subroutines described above are available, then our algorithm finds a state in $\bigcap_{i=1}^n \overline{E_i}$ efficiently in terms of the number of calls to these subroutines.*

We make a more precise statement in the following section. We note that this theorem does not assume that (Dep) holds, and the existence of resampling oracles is actually a strictly weaker condition. Thus, our algorithm provides a new proof of the existential LLL under its original assumptions, as they are stated in Theorem 1.

**1.2. Our algorithm: MaximalSetResample.** A striking aspect of the work of Moser and Tardos [35] is the simplicity and flexibility of their algorithm — in each iteration, *any* event $E_i$ that occurs can be resampled. We propose a different algorithm that is somewhat less flexible, but whose analysis seems to be simpler in our scenario. Roughly speaking, our algorithm proceeds in iterations where in each iteration we resample events that form an independent set in $G$. The independent set is generated by a greedy algorithm that adds a vertex $i$ and resamples $E_i$, if $i$ is not adjacent to the previously selected vertices and $E_i$ occurs in the current state. This is repeated until no events occur. Pseudocode for this procedure is shown in Algorithm 1. Nearly identical algorithms have been proposed before, particularly parallel algorithms [35, 28], although our interest lies not in the parallel aspects but rather in making the LLL (and its stronger variants) algorithmic in our general setting.

**Algorithm 1** MaximalSetResample uses resampling oracles to output a state $\omega \in \bigcap_{i=1}^{n} \overline{E_i}$. It requires the three subroutines described in Section 1.1.1: sampling $\omega \sim \mu$, checking if an event $E_i$ occurs, and the resampling oracles $r_i$.

Initialize $\omega$ with a random state sampled from $\mu$;
$t := 0$;
**repeat**
  $t := t + 1$;
  $J_t := \varnothing$;
  **while** there is $i \notin \Gamma^+(J_t)$ such that $\omega \in E_i$ **do**
    Let $i$ be the minimum index satisfying that condition;
    $J_t := J_t \cup \{i\}$;
    $\omega := r_i(\omega)$;  ▷ *Resample* $E_i$
  **end while**
**until** $J_t = \varnothing$;
**return** $\omega$.

Our algorithmic proof of the LLL amounts to showing that MaximalSetResample terminates, at which point $\omega \in \bigcap_{i=1}^{n} \overline{E_i}$ clearly holds. Our bound on the running time of MaximalSetResample is shown by the following theorem, which is proven in Section 4. We note that our bound is at most quadratic in the quantity $\sum_{i=1}^{n} \frac{x_i}{1-x_i}$ which was the bound proved by Moser and Tardos [35].

THEOREM 3. *Suppose that the events $E_1, \ldots, E_n$ satisfy (GLL) and that the three subroutines described above in Section 1.1.1 are available. Then the expected number of calls to the resampling oracles before MaximalSetResample terminates is*

$$O\left(\sum_{i=1}^{n} \frac{x_i}{1-x_i} \sum_{j=1}^{n} \log \frac{1}{1-x_j}\right).$$

**1.3. Generalizing the dependency condition.** A result due to Erdős and Spencer [16] shows that Theorem 1 still holds when (Dep) is generalized to[1]

$$\text{(Lop)} \qquad \Pr_{\mu}[E_i \mid \cap_{j \in J} \overline{E_j}] \leqslant \Pr_{\mu}[E_i] \qquad \forall i \in [n],\ J \subseteq [n] \backslash \Gamma^+(i).$$

They playfully called this the "lopsidependency" condition, and called $G$ a "lopsidependency graph". This more general condition enables several interesting uses of the LLL in combinatorics and theoretical computer science, e.g., existence of Latin transversals [16] and optimal thresholds for satisfiability [19].

Recall that Theorem 3 did not assume (Dep) and instead assumed the existence of resampling oracles. It is natural to wonder how the latter assumption relates to lopsidependency. We show that the existence of resampling oracles is equivalent to a condition that we call *lopsided association*, and whose strength lies strictly between (Dep) and (Lop). The lopsided association condition is

$$\text{(LopA)} \qquad \Pr_{\mu}[E_i \cap F] \geqslant \Pr_{\mu}[E_i] \cdot \Pr_{\mu}[F] \qquad \forall i \in [n], \forall F \in \mathcal{F}_i$$

[1] More precisely, (Lop) should be restricted to $J$ for which $\Pr_{\mu}[\cap_{j \in J} \overline{E_j}] > 0$. However that restriction is ultimately unnecessary because, in the context of the LLL, the theorem of Erdős and Spencer implies that $\Pr_{\mu}[\cap_{j \in [n]} \overline{E_j}] > 0$.

where $\mathcal{F}_i$ contains all events $F$ whose indicator variable is a monotone non-decreasing function of the indicator variables of $(E_j : j \notin \Gamma^+(i))$. We call a graph satisfying (LopA) a *lopsided association graph* for events $E_1, \ldots, E_n$.

THEOREM 4 (Informal). *Resampling oracles exist for events $E_1, \ldots, E_n$ and a graph $G$ if and only if $G$ is a lopsided association graph for events $E_1, \ldots, E_n$.*

This equivalence follows essentially from LP duality: the existence of a resampling oracle can be formulated as a *transportation problem* for which the lopsided association condition is exactly the necessary and sufficient condition for a feasible transportation to exist. Section 2.1 proves this result in detail.

The fact that a graph is a lopsidependency graph can be a useful property to study in its own right. For example, there are applications in which the LLL with lopsidependency is used to estimate the number of objects satisfying certain conditions, e.g., [39]. Designing resampling oracles could be a useful approach to proving that a certain graph indeed satisfies lopsidependency.

As remarked above, the dependency conditions are related by the implications

$$\text{(Dep)} \quad \Rightarrow \quad \text{(LopA)} \quad \Rightarrow \quad \text{(Lop)}.$$

The first implication is obvious since (Dep) implies that $E_i$ is independent of $F$ in (LopA). To see the second implication, simply take $F = \bigcup_{j \in J} E_j$ for any $J \subseteq [n] \backslash \Gamma^+(i)$ to obtain that $\Pr_\mu[E_i \mid \cup_{j \in J} E_j] \geqslant \Pr_\mu[E_i]$. Although lopsided association is formally a stronger assumption than lopsidependency, every use of the LLL with lopsidependency that we have studied actually satisfies the stronger lopsided association condition.

Our technical report [25] presents efficient resampling oracles for all of those scenarios. Consequently, Theorem 3 makes the LLL efficient in all of those scenarios. Due to our intention that the present paper be illustrative but not exhaustive, Section 3 discusses resampling oracles only in the scenario of random spanning trees.

As remarked above, Section 2.2 describes a scenario in which (Dep) and (GLL) are satisfied for a dependency graph $G$ but finding a state $\omega \in \bigcap_{i=1}^n \overline{E_i}$ is computationally hard, assuming standard complexity theoretic beliefs. In that scenario resampling oracles must necessarily exist since (Dep) is satisfied, but they cannot be efficiently implemented due to the computational hardness. Therefore the equivalence between (LopA) and resampling oracles comes with no efficiency guarantees. Nevertheless in all lopsidependency scenarios that we have encountered in applications of the LLL, efficient implementations of the resampling oracles arise naturally from existing work, or can be devised with modest effort. In particular this is the case for random permutations, perfect matchings in complete graphs, and spanning trees in complete graphs, as discussed in our technical report [25].

**1.4. Generalizing the LLL criterion.** In the early papers on the LLL [15, 46], the (GLL) criterion relating the dependency graph $G$ and the probabilities $\Pr_\mu[E_i]$ was shown to be a sufficient condition to ensure that $\Pr_\mu[\bigcap_{i=1}^n \overline{E_i}] > 0$. Shearer [45] discovered a more general criterion that ensures the same conclusion. In fact Shearer's criterion is the best possible, under the assumption that $G$ is undirected: whenever Shearer's criterion is violated, there exist a corresponding measure $\mu$ and events $E_1, \ldots, E_n$ for which $\Pr_\mu[\bigcap_{i=1}^n \overline{E_i}] = 0$. There are a few uses of the LLL in which $G$ is directed, but not many.

Section 4 formally defines Shearer's criterion and uses it in a fundamental way to prove Theorem 3. Moreover, we give an algorithmic proof of the LLL under Shearer's

criterion instead of the (GLL) criterion. This algorithm is efficient in typical situations, although the efficiency depends on Shearer's parameters. The following simplified result is stated formally and proven in Section 4.4.

THEOREM 5 (Informal). *Let $G$ be a graph. Let $\Pr_\mu[E_1], \ldots, \Pr_\mu[E_n]$ be probabilities that satisfy Shearer's criterion for $G$ with $\epsilon$ slack. Suppose that the three subroutines described in Section 1.1.1 are available. Then the expected number of calls to the resampling oracles by MaximalSetResample is $O(\frac{n}{\epsilon} \log \frac{1}{\epsilon})$.*

We also prove a more refined bound valid for any probabilities satisfying Shearer's criterion. This bound is similar to the bound obtained by Kolipaka and Szegedy [28]; see Section 4.4 for details.

Unfortunately Shearer's criterion is unwieldy and has not seen much use in applications of the LLL. Recently several researchers have proposed criteria of intermediate strength between (GLL) and Shearer's criterion [7, 29]. The first of these, called the *cluster expansion* criterion, was originally devised by Bissacot et al. [7], and is based on insights from statistical physics. This criterion has given improved results in several applications of the local lemma [8, 24, 36]. Previous algorithmic work has also used the cluster expansion criterion in the variable model [1, 38] and for permutations [24].

We give a new, elementary proof that the cluster expansion criterion implies Shearer's criterion. In contrast, the previous proof is analytic and requires several ideas from statistical physics [7]. As a consequence, we obtain the first purely combinatorial proof that the existential LLL holds under the cluster expansion criterion. Another consequence (Theorem 6) is an algorithm for the LLL under the cluster expansion criterion, obtained using our algorithmic results under Shearer's criterion. This generalizes Theorem 3 by replacing (GLL) with the cluster expansion criterion, stated below as (CLL). To state the result, we require additional notation: let $\mathsf{Ind}$ denote the family of independent sets in the graph $G$.

THEOREM 6. *Suppose that the events $E_1, \ldots, E_n$ satisfy the following criterion*

$$\text{(CLL)} \qquad \exists y_1, \ldots, y_n > 0 \qquad \text{such that} \qquad \Pr_\mu[E_i] \leqslant \frac{y_i}{\sum_{J \subseteq \Gamma^+(i), J \in \mathsf{Ind}} \prod_{j \in J} y_j}.$$

*and that the three subroutines described in Section 1.1.1 are available. Then the expected number of calls to the resampling oracles before MaximalSetResample terminates is $O\big(\sum_{i=1}^n y_i \sum_{j=1}^n \ln(1 + y_j)\big)$.*

**1.5. Techniques and related work.** The breakthrough work of Moser and Tardos [34, 35] stimulated a string of results on algorithms for the LLL. This section reviews the results that are most relevant to our work. Several interesting techniques play a role in the analyses of these previous algorithms. These can be roughly categorized as the *entropy compression method* [33, 3], *witness trees* or *witness sequences* [35, 24, 28] and *forward-looking combinatorial analysis* [20].

Moser [34, 33] developed the entropy compression method to analyze a very simple algorithm for the "symmetric" LLL [15], which incorporates the maximum degree of $G$ and a uniform bound on $\Pr_\mu[E_i]$. The entropy compression method roughly shows that, if the algorithm runs for a long time, a transcript of the algorithm's actions provides a compressed representation of the algorithm's random bits, which is unlikely due to entropy considerations. Following this, Moser and Tardos [35] showed that a similar algorithm will produce a state in $\bigcap_{i=1}^n \overline{E_i}$, assuming the independent variable model and the (GLL) criterion. This paper is primarily responsible for the

development of witness trees, and proved the "witness tree lemma", which yields an extremely elegant analysis in the variable model. The witness tree lemma has further implications. For example, it allows one to analyze separately for each event its expected number of resamplings. Moser and Tardos also extended the variable model to incorporate a limited form of lopsidependency, and showed that their analysis still holds in that setting.

While the work of Moser and Tardos [35] does not refer explicitly to the entropy compression method of [33], the conceptual connection is strong. The entropy compression argument inspired a significant line of work where this method is tailored to various combinatorial problems, most often dealing with graph colorings [17, 21, 41, 42, 14], and typically yielding results stronger than those obtained by a direct application of the LLL. The entropy compression method also inspired a general existential result strengthening the LLL [6].

Our work is more closely related to the original LLL: We aim to develop an algorithmic framework for applications of the LLL that overcomes the main limitation of Moser and Tardos [35], the assumption of underlying independent random variables. We achieve this by addressing the occurrence of an event through the abstract notion of resampling oracles rather than directly resampling the variables of the variable model. Furthermore we give efficient implementations of resampling oracles for essentially all known probability spaces to which the LLL has been applied. A significant difference with our work is that we do not have an analogue of the witness tree lemma; our approach provides a simpler analysis when the LLL criterion has slack but requires a more complicated analysis to remove the slack assumption. As a consequence, our bound on the number of resampling oracle calls is larger than the Moser-Tardos bound. The lack of a witness tree lemma is inherent: our technical report [25] proves that the witness tree lemma is false in the abstract scenario of resampling oracles.

The Moser-Tardos algorithm is known to terminate under criteria more general than (GLL), while still assuming the variable model. Pegden [38] showed that the cluster expansion criterion suffices, whereas Kolipaka and Szegedy [28] showed more generally that Shearer's criterion suffices. Furthermore, Harris [23] gives a new criterion under which the Moser-Tardos algorithm terminates, even though Shearer's criterion may be violated; this result also assumes the variable model. Our analysis of MaximalSetResample also holds under the cluster expansion criterion or Shearer's criterion, in the more general context of resampling oracles. Our bounds on the number of resampling operations are somewhat larger than those of [38, 28], but the increase is at most quadratic.

Kolipaka and Szegedy [28] present another algorithm, called GeneralizedResample, whose analysis proves the LLL under Shearer's condition for arbitrary probability spaces. GeneralizedResample is similar to MaximalSetResample in that they both work with abstract distributions and that they repeatedly choose a maximal independent set $J$ of undesired events to resample. However, the way that the bad events are resampled is different: GeneralizedResample needs to sample from $\mu\big|_{\cap_{j\notin\Gamma^+(J)}\overline{E_j}}$, which is a complicated operation that seems difficult to implement efficiently. Thus MaximalSetResample can be viewed as a variant of GeneralizedResample that can be made efficient in all known scenarios.

Harris and Srinivasan [24] show that the Moser-Tardos algorithm can be adapted to handle certain events in a probability space involving random permutations. Their method for resampling an event is based on the Fisher-Yates shuffle. Since their

resampling method perfectly satisfies the criteria of a resampling oracle, this permutation scenario can also be handled by our framework, as discussed in our technical report [25]. The result of Harris and Srinivasan is stronger than ours in that they do prove an analog of the witness tree lemma. Consequently their algorithm requires fewer resamplings than ours, and they are able to derive parallel variants of their algorithm. The work of Harris and Srinivasan is technically challenging, and generalizing it to a more abstract setting seems daunting.

Achlioptas and Iliopoulos [3] proposed a general framework for finding "flawless objects", based on actions for addressing flaws. We call this the A-I framework. They show that, under certain conditions, a random walk over such actions rapidly converges to a flawless object. This naturally relates to the LLL by viewing each event $E_i$ as a flaw. At the same time, the A-I framework is not tied to the probabilistic formulation of the LLL, and can derive results, such as the greedy algorithm for vertex coloring, that seem to be outside the scope of typical LLL formulations, such as Theorem 1. The A-I framework [3] has other restrictions and does not claim to recover any particular form of the LLL. Nevertheless, the framework can accommodate applications of the LLL where lopsidependency plays a role, such as rainbow matchings and rainbow Hamilton cycles. In contrast, our framework embraces the probabilistic formulation and can recover the original existential LLL (Theorem 1) in full generality, even incorporating Shearer's generalization. The A-I analysis [3] is inspired by Moser's entropy method. Technically, it entails an encoding of random walks by "witness forests" and combinatorial counting thereof to estimate the length of the random walk. The terminology of witness forests is reminiscent of the witness trees of Moser and Tardos, but conceptually they are different in that the witness forests grow "forward in time" rather than backward. This is conceptually similar to "forward-looking combinatorial analysis", which we discuss next.

Giotis et al. [20] show that a variant of Moser's algorithm gives an algorithmic proof in the variable model of the symmetric LLL. While this result is relatively limited when compared to the results above, their analysis is a clear example of forward-looking combinatorial analysis. Whereas Moser and Tardos use a *backward-looking* argument to find witness trees in the algorithm's "log", Giotis et al. analyze a *forward-looking* structure: the tree of resampled events and their dependencies, looking forward in time. This viewpoint seems more natural and suitable for extensions.

Our approach can be roughly described as *forward-looking analysis* with a careful modification of the Moser-Tardos algorithm, formulated in the framework of resampling oracles. Our main conceptual contribution is the simple definition of the resampling oracles, which allows the resamplings to be readily incorporated into the forward-looking analysis. Our modification of the Moser-Tardos algorithm is designed to combine this analysis with the technology of "stable set sequences" [28], defined in Section 4.1, which allows us to accommodate various LLL criteria, including Shearer's criterion. This plays a fundamental role in the full proof of Theorem 3.

Our second contribution is a technical idea concerning slack in the LLL criteria. This idea is a perfectly valid statement regarding the existential LLL as well, although we will exploit it algorithmically. One drawback of the forward-looking analysis is that it naturally leads to an exponential bound on the number of resamplings, unless there is some slack in the LLL criterion; this same issue arises in [3, 20]. Our idea eliminates the need for slack in the (GLL) and (CLL) criteria, at least in the setting where $G$ is undirected. We prove that, even if (GLL) or (CLL) are tight, we can instead perform our analysis using Shearer's criterion, which is never tight because it defines an open set. For example, consider the familiar case of Theorem 1, and suppose that

(GLL) holds with equality, i.e., $\Pr_\mu[E_i] = x_i \prod_{j\in\Gamma(i)}(1-x_j)$ for all $i$. We show that the conclusion of the LLL remains true even if each event $E_i$ actually had the larger probability $\Pr_\mu[E_i]\cdot\left(1+(2\sum_i \frac{x_i}{1-x_i})^{-1}\right)$. The proof of this fact crucially uses Shearer's criterion and it does not seem to follow from more elementary tools [15, 46].

**Follow-up work.** Achlioptas and Iliopoulos generalized their framework further to incorporate our notion of resampling oracles [2]. This subsequent work can be viewed as a unification of their framework and ours; it has the benefit of both capturing the framework of resampling oracles and allowing some additional flexibility (in particular, the possibility of regenerating the measure $\mu$ approximately rather than exactly). We remark that this work is still incomparable with ours, primarily due to the facts that our analysis is performed in Shearer's more general setting, and that our algorithm is efficient even when the LLL criteria are tight.

Kolmogorov [30] studies why MaximalSetResample needs to examine the events in a fixed order. He formulates a notion of "commutativity" in the resampling operations, and shows that this suffices to allow events to be resampled in an arbitrary order. He shows that the resampling oracles for permutations [24, 25] and perfect matchings [25] are commutative. However, the resampling oracles for spanning trees discussed in Section 3.2 appear not to be commutative. So it seems possible that there are scenarios for the LLL with lopsidependency for which resampling oracles exist but commutative ones do not.

Although we have shown that the witness tree lemma does not necessarily hold in the setting of resampling oracles [25], Iliopoulos [27] has shown that the witness tree lemma does hold with commutative resampling oracles. This has several uses, for example, to approximate the distribution $\mu|_{\cap_{j=1}^n \overline{E_j}}$ in the setting of commutative resampling oracles.

**1.6. Organization.** The rest of the paper is organized as follows. In Section 2, we discuss the connection between resampling oracles and the assumptions of the Lovász Local Lemma. We also show here that resampling oracles as well as the LLL itself can be computationally hard in general. In Section 3, we show concrete examples of efficient implementations of resampling oracles and an illustrative application. Finally, in Section 4 we present the full analysis of our algorithm.

**2. Resampling oracles: existence and efficiency.** The algorithms in this paper do not explicitly assume that the lopsidependency condition (Lop) holds, but instead assume the existence of resampling oracles. To understand the relationship between those two assumptions, first recall the condition (LopA), which was defined on page 4, and is a strengthening of (Lop). In Section 2.1 we show that the existence of a resampling oracle for each event is equivalent to the condition (LopA).

We should emphasize that the *efficiency of an implementation* of a resampling oracle is a separate issue. There is no general guarantee that resampling oracles can be implemented efficiently. Indeed, as we show in Section 2.2, there are applications of the LLL such that the resampling oracles are hard to implement efficiently, and finding a state avoiding all events is computationally hard, under standard computational complexity assumptions.

Nevertheless, this is not an issue in common applications of the LLL: resampling oracles exist and can be implemented efficiently in all uses of the LLL of which we are aware, even those involving lopsidependency. Section 3 discusses the scenario of random spanning trees, and other scenarios are discussed in our technical report [25].

**2.1. Existence of resampling oracles.** This section proves an equivalence lemma connecting resampling oracles with the notion of lopsided association. First, let us define formally what we call a resampling oracle.

DEFINITION 7. *Let $E_1, \dots, E_n$ be events on a space $\Omega$ with a probability measure $\mu$, and let $G = ([n], E)$ be a graph with neighbors of $i \in [n]$ denoted by $\Gamma(i)$. Let $r_i$ be a randomized procedure that takes a state $\omega \in \Omega$ and outputs a state $r_i(\omega) \in \Omega$. We say that $r_i$ is a resampling oracle for $E_i$ with respect to $G$, if*

(R1) *For $\omega \sim \mu|_{E_i}$, we obtain $r_i(\omega) \sim \mu$. (The oracle $r_i$ removes conditioning on $E_i$.)*

(R2) *For any $j \notin \Gamma^+(i) = \Gamma(i) \cup \{i\}$, if $\omega \notin E_j$ then also $r_i(\omega) \notin E_j$. (Resampling an event cannot cause new non-neighbor events to occur.)*

Next, let us define the notion of a lopsided association graph.

DEFINITION 8. *A graph $G$ with neighborhood function $\Gamma$ is a lopsided association graph for events $E_1, \dots, E_n$ if*

$$\text{(LopA)} \qquad \Pr_\mu[E_i \cap F] \;\geqslant\; \Pr_\mu[E_i] \cdot \Pr_\mu[F] \qquad \forall i \in [n], \forall F \in \mathcal{F}_i$$

*where $\mathcal{F}_i$ contains all events $F$ whose indicator variable is a monotone non-decreasing function of the indicator variables of $(E_j : j \notin \Gamma^+(i))$.*

LEMMA 9. *Consider a fixed $i \in [n]$ and assume $\Pr_\mu[E_i] > 0$. The following statements are equivalent.*

(a) *There exists a resampling oracle $r_i$ satisfying the conditions* (R1) *and* (R2) *with respect to a neighborhood $\Gamma^+(i)$.*

(b) *$\Pr_\mu[E_i \cap F] \geqslant \Pr_\mu[E_i] \cdot \Pr_\mu[F]$ for any event $F \in \mathcal{F}_i$.*

We remark that Lemma 9 ignores all issues of computational efficiency.

COROLLARY 10. *Resampling oracles $r_1, \dots, r_n$ exist for events $E_1, \dots, E_n$ with respect to a graph $G$ if and only if $G$ is a lopsided association graph for $E_1, \dots, E_n$. Both statements imply that the lopsidependency condition* (Lop) *holds.*

*Proof* (of Lemma 9). In this proof it will be convenient to emphasize the relationship between events and states by letting $E_i[\omega]$ denote the $\{0,1\}$-valued function that indicates whether event $E_i$ occurs at the state $\omega \in \Omega$.

(a) $\Rightarrow$ (b): Consider the coupled states $(\omega, \omega')$ where $\omega \sim \mu|_{E_i}$ and $\omega' = r_i(\omega)$. By (R1), $\omega' \sim \mu$. For any event $F \in \mathcal{F}_i$, if $F$ does not occur at $\omega$ then it does not occur at $\omega'$ either, due to (R2). This establishes that

$$\Pr_\mu[F] \;=\; \mathbf{E}_{\omega' \sim \mu}[F[\omega']] \;\leqslant\; \mathbf{E}_{\omega \sim \mu|E_i}[F[\omega]] \;=\; \Pr_\mu[F \mid E_i],$$

which implies $\Pr_\mu[F \cap E_i] \geqslant \Pr_\mu[F] \cdot \Pr_\mu[E_i]$. In particular this implies (Lop), by taking $F = \bigcup_{j \in J} E_j$.

(b) $\Rightarrow$ (a): We begin by formulating the existence of a resampling oracle as the following *transportation problem*. Consider a bipartite graph $(U \cup W, E)$, where $U$ and $W$ are disjoint, $U$ represents all the states $\omega \in \Omega$ satisfying $E_i$, and $W$ represents all the states $\omega \in \Omega$. Edges represent the possible actions of the resampling oracle: $(u, w) \in E$ if $u$ satisfies every event among $(E_j : j \notin \Gamma^+(i))$ that $w$ satisfies. Each vertex has an associated weight: For $w \in W$, we define $p_w = \Pr_\mu[w]$, and for $u \in U$, $p_u = \Pr_\mu[u] / \Pr_\mu[E_i]$, i.e, $p_u$ is the probability of $u$ conditioned on $E_i$. We claim that the resampling oracle $r_i$ exists if and only if there is an assignment $f_{uw}$ of values to

the edges such that

$$(1)\qquad \begin{array}{ll} \sum_{w:(u,w)\in E} f_{uw} = p_u & \forall u \in U \\ \sum_{u:(u,w)\in E} f_{uw} = p_w & \forall w \in W \\ f_{uw} \geqslant 0 & \forall u \in U,\, w \in W. \end{array}$$

Such an assignment is called a feasible transportation. Given such a transportation, the resampling oracle is defined naturally by following each edge from $u \in U$ with probability $f_{uw}/p_u$, and the resulting distribution on $W$ is $p_w$. Conversely, for a resampling oracle which, for a given state $u \in U$, generates $w \in W$ with probability $q_{uw}$, we define $f_{uw} = p_u q_{uw}$. This assignment satisfies (1).

Our goal at this point is to show that (b) implies feasibility of (1). A condition that is equivalent to (1), but more convenient for our purposes, can be determined from LP duality [43, Theorem 21.11]. A feasible transportation exists if and only if

$$(2)\qquad \begin{array}{lll} (2.1) & \sum_{u\in U} p_u & = \sum_{w\in W} p_w \\ (2.2) & \sum_{u\in A} p_u & \leqslant \sum_{w\in \Gamma(A)} p_w \qquad \forall A \subseteq U, \end{array}$$

where $\Gamma(A) = \{\, w \in W : \exists u \in A \text{ s.t. } (u,w) \in E \,\}$. This is an extension of Hall's condition for the existence of a perfect matching.

Our goal at this point is to show that (b) implies feasibility of (2). Let us now simplify (2). Fix any $A \subseteq U$. The neighborhood $\Gamma(A)$ consists of states satisfying at most those events among $\{\, E_j : j \notin \Gamma^+(i) \,\}$ satisfied by some state in $A$. Thus $\Gamma(A)$ corresponds to an event $F'$ such that $F'[\omega]$ is a *non-increasing* function of $(\, E_j[\omega] : j \notin \Gamma^+(i) \,)$. Next observe that, if the set of events among $\{\, E_j : j \notin \Gamma^+(i) \,\}$ satisfied by $u' \in U$ is a subset of those satisfied by $u \in U$, then $\Gamma(u') \subseteq \Gamma(u)$. Suppose that, for each $u \in A$, we add to $A$ all such vertices $u'$. Doing so can only increase the left-hand side of (2.2), but does not increase the right-hand side as $\Gamma(A)$ remains unchanged (since $\Gamma(u') \subseteq \Gamma(u)$). Furthermore, the resulting set $A$ corresponds to the same event $F'$, but restricted to the states in $U$. Let us call such a set $A$ non-increasing. Let (2*) denote the simplification of (2) in which we restrict to non-increasing $A$. We have argued that (2) and (2*) are equivalent.

Our goal at this point is to show that (b) implies feasibility of (2*). It is easy to see that (b) is equivalent to

$$\Pr_\mu[\overline{F} \cap E_i] \leqslant \Pr_\mu[\overline{F}] \cdot \Pr_\mu[E_i] \qquad \forall F \in \mathcal{F}_i.$$

Assuming $\Pr[E_i] > 0$, we can rewrite this as $\Pr_\mu[\overline{F} \mid E_i] \leqslant \Pr_\mu[\overline{F}]$ $\forall F \in \mathcal{F}_i$. Now consider using this inequality with $F = \overline{F'}$ for each $F'$ corresponding to some non-increasing set $A \subseteq U$. Since $F'$ is a non-increasing function of $(\, E_j[\omega] : j \notin \Gamma^+(i) \,)$, then $F = \overline{F'}$ is a non-decreasing function of those events, and therefore $F \in \mathcal{F}_i$ as required. We then have

$$\sum_{u\in A} p_u = \Pr_\mu[F' \mid E_i] \leqslant \Pr_\mu[F'] = \sum_{w\in\Gamma(A)} p_w.$$

This verifies the feasibility of (2*), and hence the desired resampling oracle exists. □

**2.2. Computational hardness of the LLL.** This section considers whether the LLL can always be made algorithmic. We show that, even in fairly simple scenarios where the LLL applies, finding the desired output can be computationally hard. This

fact seems not to have been observed in the literature to date. We first observe that the question of algorithmic efficiency must be stated carefully otherwise hardness is trivial.

*A trivial example.* Given a Boolean formula $\phi$, let the probability space be $\Omega = \{0,1\}$, and let $\mu$ be the uniform measure on $\Omega$. There is a single event $E_1$ defined to be $E_1 = \{1\}$ if $\phi$ is satisfiable, and $E_1 = \{0\}$ if $\phi$ is not satisfiable. Since $\Pr[E_1] = 1/2$, the (GLL) criterion holds trivially with $x_1 = 1/2$. The LLL gives the obvious conclusion that there is a state $\omega \notin E$. Yet, finding this state requires deciding satisfiability of $\phi$, which is NP-complete.

The reason that this example is trivial is that even deciding whether the undesired event has occurred is computationally hard. A more meaningful discussion of LLL efficiency ought to rule out this trivial example by considering only scenarios that satisfy some reasonable assumptions. With that in mind, we will assume that

- there is a probability space $\Omega$, whose states can be described by $m$ bits;
- a graph $G$ satisfying (Dep) for events $E_1, \dots, E_n$ is explicitly provided;
- $x_1, \dots, x_n \in (0,1)$ satisfying the (GLL) conditions are provided, and the value $\sum_{i=1}^{n} \frac{x_i}{1-x_i}$ is at most $\mathrm{poly}(n)$;
- there is a subroutine that provides an independent random state $\omega \sim \mu$ in $\mathrm{poly}(m)$ time;
- for each $i \in [n]$, there is a subroutine which determines for any given $\omega \in \Omega$ whether $\omega \in E_i$, in $\mathrm{poly}(m)$ time.

As far as we know, no prior work refutes the possibility that there is an algorithmic form of the LLL, with running time $\mathrm{poly}(m,n)$, in this general scenario.

Our results imply that resampling oracles do *exist* in this general scenario, so it is only the question of whether these resampling oracles are *efficient* that prevents Theorem 3 from providing an efficient algorithm. Nevertheless, we show that there is an instance of the LLL that satisfies the reasonable assumptions stated above, but for which finding a state in $\bigcap_i \overline{E_i}$ requires solving a problem that is computationally hard (under standard computational complexity assumptions). As a consequence, we conclude that the resampling oracles cannot always be implemented efficiently, even under the reasonable assumptions of this general scenario.

We remark that NP-completeness is not the right notion of hardness here [37]. Problems in NP involve deciding whether a solution exists, whereas the LLL *guarantees that a solution exists*, and the goal is to explicitly find a solution. Our result is instead based on hardness of the *discrete logarithm* problem, a standard belief in computational complexity theory. This problem is defined as follows. For a prime $p$ and positive integer $n$, let $\mathrm{GF}(p^n)$ denote the finite field of order $p^n$, and $\mathrm{GF}^*(p^n)$ its multiplicative group of nonzero elements. Given a generator $g$ of $\mathrm{GF}^*(p^n)$ and an element $h \in \mathrm{GF}^*(p^n)$, the goal is to find an integer $1 \leqslant k \leqslant p^n - 1$ such that $g^k = h$.

THEOREM 11. *There are instances of events $E_1, \dots, E_n$ on a probability space $\Omega = \{0,1\}^n$ under the uniform probability measure, such that*

- *the events $E_i$ are mutually independent;*
- *for each $i \in [n]$, the condition $\omega \in E_i$ can be checked in poly$(n)$ time for given $\omega \in \Omega$;*
- *the (GLL) conditions are satisfied with $x_i = 1/2$ for each $i \in [n]$;*

*but finding a state in $\bigcap_{i=1}^{n} \overline{E_i}$ is as hard as solving the discrete logarithm problem in $\mathrm{GF}^*(2^n)$.*

*Remark.* At first glance, this result may seem to contradict the fact that the LLL can

be made algorithmic in the variable model [35], where events are defined on underlying independent random variables. The key point is that the variable model also relies on a particular type of dependency graph (defined by shared variables) which might have more edges than necessary, and therefore might have stronger hypotheses than necessary. Theorem 11 shows that, even if the probability space consists of independent $\{0,1\}$ random variables, the LLL cannot in general be made algorithmic if the dependency graph of the variable model is not used, and instead the true dependencies are considered.

*Proof.* Consider an instance of the discrete logarithm problem in the finite field $\mathrm{GF}(2^n)$. We define an instance of $n$ events on $\Omega = \{0,1\}^n$ as follows. First identify $\Omega = \{0,1\}^n$ with $[2^n]$ as well as $\mathrm{GF}(2^n)$ in a natural way. We define $f : [2^n] \to \mathrm{GF}(2^n)$ by $f(0) = 0$ and $f(x) = g^x$ for $x \neq 0$, where the exponentiation is performed in $\mathrm{GF}(2^n)$. For each $i \in [n]$, we define an event $E_i$ that occurs for $\omega \in \{0,1\}^n$ iff $(f(\omega))_i = 1 - h_i$. This is a condition that can be checked in time $\mathrm{poly}(n)$, by computing $f(\omega) = g^\omega$ where we interpret $\omega$ as $\sum_{i=0}^{n-1} \omega_i 2^i$ and compute $g^\omega$ by taking squares iteratively.

Observe that for $\omega$ distributed uniformly in $\Omega = \{0,1\}^n$, $f(\omega)$ is again distributed uniformly in $\Omega$, since $f$ is a bijection. (Note that 0 is mapped to 0, and $f(\omega)$ for $\omega \neq 0$ generates each element of the multiplicative group $\mathrm{GF}^*(2^n)$ exactly once). Therefore, the probability of $E_i$ is $1/2$, for each $i \in [n]$. Further, the events $E_1, \ldots, E_n$ are mutually independent, since for any $J \subseteq [n]$, $\bigcap_{j \in J} E_j \cap \bigcap_{j' \notin J} \overline{E_{j'}}$ occurs iff $f(\omega) = h \oplus \mathbf{1}_J$, which happens with probability $1/2^n$. Here $\mathbf{1}_J \in \{0,1\}^n$ is the indicator vector for the set $J$, and $\oplus$ denotes addition in $\mathrm{GF}(2^n)$ (i.e., component-wise xor in $\{0,1\}^n$). Hence the dependency graph is empty, and the LLL with parameters $x_i = 1/2$ trivially implies that there exists a state $\omega$ avoiding all the events. In this instance, we know explicitly that the state avoiding all the events is $f^{-1}(h)$. Therefore, if we had an efficient algorithm to find this point for any given $h \in \mathrm{GF}^*(2^n)$, we would also have an efficient algorithm for the discrete logarithm problem in $\mathrm{GF}(2^n)$. ∎

**3. Implementation of resampling oracles and applications.** In this section, we present efficient implementations of resampling oracles in two application settings: the variable model introduced by Moser and Tardos [35], and a setting of random spanning trees in complete graphs. Other settings are discussed in our technical report [25].

To be more precise, resampling oracles also depend on the types of events and dependencies that can be handled. For example, if the probability space consists of independent random variables, the variable model allows any events but assumes dependency between any two events that share any variables. In the setting of spanning trees, we consider the "canonical events" defined by [31], characterized by the appearance of a certain subset of edges.

Finally, Section 3.3 shows how resampling oracles for a certain probability space can be extended in a natural way to products of such probability spaces. This allows us to generalize from resampling oracles for one random spanning tree to a collection of independent random spanning trees. Section 3.4 gives an application of these techniques to a problem involving rainbow spanning trees.

**3.1. The variable model.** This is the most common setting, considered originally by Moser and Tardos [35]. Here, $\Omega$ has a product structure corresponding to independent random variables $\{ X_a : a \in \mathcal{U} \}$. The probability measure $\mu$ here is a product measure. Each bad event $E_i$ depends on a particular subset of variables $A_i$.

Adjacency in the dependency graph is defined by $i \backsim j$ iff $A_i \cap A_j \neq \varnothing$.

Here our algorithmic assumptions correspond exactly to the Moser-Tardos framework [35]. Sampling from $\mu$ means generating a fresh set of random variables independently. The resampling oracle $r_i$ takes a state $\omega$ and replaces the random variables $\{ X_a : a \in A_i \}$ by fresh random samples. It is easy to see that the assumptions are satisfied: in particular, a random state sampled from $\mu$ conditioned on $E_i$ has all variables outside of $A_i$ independently random (according to $\mu$). Hence, resampling the variables of $A_i$ produces the distribution $\mu$. Clearly, resampling $\{ X_a : a \in A_i \}$ does not affect any events whose variables do not intersect $A_i$.

We note that this resampling oracle is also consistent with the notion of lopsidependency on product spaces considered by [35]: They call two events $E_i, E_j$ lopsidependent, if $A_i \cap A_j \neq \varnothing$ and it is possible to cause $E_j$ to occur by resampling $A_i$ in a state where $E_i$ holds but $E_j$ does not. (The definition in [35] is worded differently but equivalent to this.) This is exactly the condition that we require our resampling oracle to satisfy.

**3.2. Spanning trees.** Here, the probability space $\Omega$ is the set of all spanning trees in $K_n$, the complete graph on $n$ vertices. Let us consider an event $E_A$ for a set of edges $A$, where $E_A$ occurs for $T \in \Omega$ iff $A \subseteq T$. We now define an adjacency relation between edge sets as follows: $A \backsim B$ if there exist edges $e \in A$ and $f \in B$ such that $e$ and $f$ share exactly one vertex. Lu et al. [31, Lemma 7] show that this relation in fact defines a *lopsidependency* graph (which they call a negative dependency graph) for spanning trees.

To implement a resampling oracle in this setting, we will use as a subroutine an algorithm to generate a uniformly random spanning tree in a given graph $G$. This can be done efficiently by several methods, for example by a random walk [9].

**Algorithm 2** Resampling oracle for spanning trees

**Function** $r_A(T)$:
Check that $A \subseteq T$, otherwise **fail**.
Let $W = V(A)$, the vertices covered by $A$.
Let $T_1 = \binom{V \setminus W}{2} \cap T$, the edges of $T$ disjoint from $W$.
Let $F_1 = \binom{V \setminus W}{2} \setminus T$, the edges disjoint from $W$ not present in $T$.
Let $G_2 = (K_n \setminus F_1)/T_1$ be the multigraph obtained by deleting $F_1$ and contracting $T_1$.
Generate a uniformly random spanning tree $T_2$ in $G_2$.
**return** $T_1 \cup T_2$.

LEMMA 12. *If $A$ is a fixed forest and $T$ is a uniformly random spanning tree in $K_n$ conditioned on $A \subseteq T$, then $r_A(T)$ produces a uniformly random spanning tree in $K_n$.*

*Proof.* First, observe that since $T_2$ is a spanning tree of $G_2 = (K_n \setminus F_1)/T_1$, it is also a spanning tree of $K_n/T_1$ where $T_1$ is a forest, and therefore $T_1 \cup T_2$ is a spanning tree of $K_n$. We need to prove that it is a uniformly random spanning tree.

First, we appeal to a known result [31, Lemma 6] concerning spanning trees in $K_n$. Let $F$ be a forest in $K_n$ with $m$ components, and let $f_i$ be the number of vertices

in the $i^{\text{th}}$ component. Then the number of spanning trees containing $F$ is exactly

$$n^{n-2}\prod_{i=1}^{m}\frac{f_i}{n^{f_i-1}}. \tag{3}$$

Equivalently (since $n^{n-2}$ is the total number of spanning trees), for a uniformly random spanning tree $T$, $\Pr[F \subseteq T] = \prod_{i=1}^{m} f_i/n^{f_i-1}$. This has the surprising consequence that for vertex-disjoint forests $F_1, F_2$, we have

$$\Pr[F_1 \cup F_2 \subseteq T] \;=\; \Pr[F_1 \subseteq T] \cdot \Pr[F_2 \subseteq T],$$

i.e., the containment of $F_1$ and $F_2$ are independent events. (In a general graph, the appearances of different edges in a random spanning tree are negatively correlated, but here we are in a complete graph.)

Let $W = V(A)$ and let $B$ be any forest on $V\backslash W$, i.e., vertex-disjoint from $A$. By the above, the appearance of $B$ in a uniformly random spanning tree is independent of the appearance of $A$. Hence, if $T$ is uniformly random, we have $\Pr[B \subseteq T \mid A \subseteq T] = \Pr[B \subseteq T]$. This implies that the distribution of $T \cap \binom{V\backslash W}{2}$ is exactly the same for a uniformly random spanning tree $T$ as it is for one conditioned on $A \subseteq T$ (formally, by applying the inclusion-exclusion formula). Recall the lemma's hypothesis that $T$ is uniform conditioned on $A \subseteq T$. It follows that the forest $T_1 = T \cap \binom{V\backslash W}{2}$ has the same distribution as a uniformly random spanning tree, restricted to $V\backslash W$.

The final step is that we extend $T_1$ to a spanning tree $T_1 \cup T_2$, where $T_2$ is a uniform spanning tree in $G_2 = (K_n\backslash F_1)/T_1$. Note that $G_2$ is a multigraph, i.e., it is important that we preserve the multiplicity of edges after contraction. The spanning trees $T_2$ in $G_2 = (K_n\backslash F_1)/T_1$ are in a one-to-one correspondence with spanning trees in $K_n$ conditioned on $T \cap \binom{V\backslash W}{2} = T_1$. This is because each such tree $T_2$ extends $T_1$ to a different spanning tree of $K_n$, and each spanning tree where $T \cap \binom{V\backslash W}{2} = T_1$ can be obtained in this way. Therefore, for a fixed $T_1$, $T_1 \cup T_2$ is a uniformly random spanning tree conditioned on $T \cap \binom{V\backslash W}{2} = T_1$. Finally, since the distribution of $T_1$ is equal to that of a uniformly random spanning tree restricted to $V\backslash W$, $T_1 \cup T_2$ is a uniformly random spanning tree. ∎

LEMMA 13. *The resampling oracle $r_A(T)$ applied to a spanning tree satisfying $E_A$ does not cause any new event $E_B$ such that $B \notin \Gamma^+(A)$.*

*Proof.* Note that the only edges that we modify are those incident to $W = V(A)$. Therefore, any new event $E_B$ that the operation of $r_A$ could cause must be such that $B$ contains an edge incident to $W$ and not contained in $A$. Such an edge shares exactly one vertex with some edge in $A$ and hence $B \backsim A$. ∎

**3.3. Composition of resampling oracles for product spaces.** Suppose we have a product probability space $\Omega = \Omega_1 \times \Omega_2 \times \ldots \times \Omega_N$, where on each $\Omega_i$ we have resampling oracles $r_{ij}$ for events $E_{ij}, j \in \mathcal{E}_i$, with respect to a graph $G_i$. Our goal is to show that there is a natural way to combine these resampling oracles in order to handle events on $\Omega$ that are obtained by taking intersections of the events $E_{ij}$. The following theorem formalizes this notion.

THEOREM 14. *Let $\Omega_1, \ldots, \Omega_N$ be probability spaces, where for each $\Omega_i$ we have resampling oracles $r_{ij}$ for events $E_{ij}, j \in \mathcal{E}_i$ with respect to a graph $G_i$. Let $\Omega = \Omega_1 \times \Omega_2 \times \ldots \Omega_N$ be a product space with the respective product probability measure. For any set $J$ of pairs $(i,j), j \in \mathcal{E}_i$ where each $i \in [N]$ appears at most once, define*

*an event $E_J$ on $\Omega$ to occur in a state $\omega = (\omega_1, \ldots, \omega_N)$ iff $E_{ij}$ occurs in $\omega_i$ for each $(i,j) \in J$. Define a graph $G$ on these events by $J \rightharpoonup J'$ iff there exist pairs $(i,j) \in J, (i,j') \in J'$ such that $j \rightharpoonup j'$ in $G_i$. Then there exist resampling oracles $r_J$ for the events $E_J$ with respect to $G$, which are obtained by calling in succession each of the oracles $r_{ij}$ for $(i,j) \in J$.*

*Proof.* For notational simplicity, let us assume that on each $\Omega_i$ we have a trivial event $E_{i0} = \Omega_i$ and the respective resampling oracle $r_{i0}$ is the identity on $\Omega_i$. Then we can assume that each collection of events $J$ is in the form

$$J = \{(1, j_1), (2, j_2), \ldots, (N, j_N)\},$$

where we set $j_\ell = 0$ for components where there is no event to resample. We define

$$r_J(\omega_1, \ldots, \omega_N) \;=\; (r_{1j_1}(\omega_1), r_{2j_2}(\omega_2), \ldots, r_{Nj_N}(\omega_N)).$$

We claim that these are resampling oracles with respect to $G$ as defined in the theorem.

Let us denote by $\mu_i$ the probability distribution on $\Omega_i$ and by $\mu$ the product distribution on $\Omega$. For the first condition, suppose that $\omega \sim \mu|_{E_J}$. By the product structure of $\Omega$, this is the same as having $\omega = (\omega_1, \ldots, \omega_N)$ where the components are independent and $\omega_\ell \sim \mu_\ell|_{E_{\ell j_\ell}}$ for each $(\ell, j_\ell) \in J$, and $\omega_\ell \sim \mu_\ell$ for components such that $j_\ell = 0$. By the properties of the resampling oracles $r_{\ell j_\ell}$, we have $r_{\ell j_\ell}(\omega_\ell) \sim \mu_\ell$. Since the resampling oracles are applied with independent randomness for each component, we have

$$r_J(\omega) \;=\; (r_{1j_1}(\omega_1), r_{2j_2}(\omega_2), \ldots, r_{Nj_N}(\omega_N)) \;\sim\; \mu_1 \times \mu_2 \times \ldots \times \mu_N \;=\; \mu.$$

For the second condition, note that if $\omega \notin E_{J'}$ and $r_J(\omega) \in E_{J'}$, it must be the case that there is $(\ell, j_\ell) \in J$ and $(\ell, j'_\ell) \in J'$ such that $\omega_\ell \notin E_{\ell j'_\ell}$ and $r_{\ell j_\ell}(\omega) \in E_{\ell j'_\ell}$. However, this is possible only if $j_\ell \rightharpoonup j'_\ell$ in the graph $G_\ell$. By the definition of $G$, this means that $J \rightharpoonup J'$ as well. ∎

The next section illustrates this result by extending our resampling oracles for random spanning trees to $N$-tuples of independent random spanning trees.

**3.4. Application: rainbow spanning trees.** This section presents an application of our framework in the setting of spanning trees. To the best of our knowledge, no previous algorithmic form of the LLL could handle this scenario.

The setting under consideration involves an edge-coloring of $K_n$. This coloring may be improper, meaning that intersecting edges may have the same color. Given such an edge-coloring of $K_n$, a spanning tree is called rainbow if the colors of its edges are distinct.

The existence of a single rainbow spanning tree is completely resolved by the matroid intersection theorem: It can be decided efficiently whether a rainbow spanning tree exists for a given edge coloring, and it can be found efficiently if it exists. However, the existence of multiple edge-disjoint rainbow spanning trees is more challenging. An attractive conjecture of Brualdi and Hollingsworth [10] states that if $n \geqslant 6$ is even and $K_n$ is properly edge-colored by $n-1$ colors, then the edges can be decomposed into $n/2$ rainbow spanning trees, each tree using each color exactly once. Until recently, it was only known that every such edge-coloring contains 2 edge-disjoint rainbow spanning trees [4]. In [11], it was proved that for any proper edge-coloring (and more generally for any coloring where each color appears at most $n/2$ times) then there exist $\Omega(n/\log n)$ edge-disjoint rainbow spanning trees. We prove that there exist

$\Omega(n)$ rainbow spanning trees under a somewhat stronger coloring assumption, and using our framework we can also do this algorithmically.[2]

THEOREM 15. *Given an edge-coloring of $K_n$ such that each color appears on at most $\frac{1}{32}(\frac{3}{4})^3 n$ edges, at least $\frac{1}{32}(\frac{3}{4})^3 n$ edge-disjoint rainbow spanning trees exist and can be found in $O(n^4)$ resampling oracle calls with high probability.*

Our result relies on Theorem 6, our algorithmic version of the LLL under the cluster expansion criterion. To obtain the result with high probability, we appeal to a more refined bound that we state in Theorem 57. We note that if there is constant multiplicative slack in the assumption on color appearances, the number of resamplings improves to $O(n^2)$, using the result in Theorem 57 with constant $\epsilon$ slack.

To prove the existential statement, we simply sample $\frac{1}{32}(\frac{3}{4})^3 n$ independently random spanning trees and hope that they will be both pairwise edge-disjoint and rainbow. This unlikely proposition happens to be true with positive probability, thanks to the LLL and the independence properties of random spanning trees that we mentioned in Section 3.2. Given this setup, our framework implies that we can also find the rainbow trees efficiently.

*Proof.* We apply our algorithm in the setting of $t$ independent and uniformly random spanning trees $T_1, \dots, T_t \subset K_n$. The bad events in our probability space are of two types:

- $E^i_{ef}$: For each $i \in [t]$ and two edges $e \neq f$ in $K_n$ of the same color, $E^i_{ef}$ occurs if $\{e, f\} \subset T_i$;
- $E^{ij}_e$: For each $i \neq j \in [t]$ and an edge $e$ in $K_n$, $E^{ij}_e$ occurs if $e \in T_i \cap T_j$.

Clearly, if no bad event occurs then the $t$ trees are rainbow and pairwise edge-disjoint.

By (3) the probability of a bad event of the first type is $\Pr[E^i_{ef}] = 3/n^2$ if $|e \cup f| = 3$ and $\Pr[E^i_{ef}] = 4/n^2$ if $|e \cup f| = 4$. The probability of a bad event of the second type is $\Pr[E^{ij}_e] = (2/n)^2 = 4/n^2$, since each of the two trees contains $e$ independently with probability $2/n$. Hence, the probability of each bad event is upper-bounded by $p = 4/n^2$.

In Section 3.2 we constructed a resampling oracle $r_A$ for a single spanning tree. By Theorem 14, this resampling oracle extends in a natural way to the setting of $t$ independent random spanning trees. In particular, for an event $E^i_{ef}$, we define $r^i_{ef}$ as an application of the resampling oracle $r_{\{e,f\}}$ to the tree $T_i$. For an event $E^{ij}_e$, we define $r^{ij}_e$ as an application of the resampling oracle $r_{\{e\}}$ independently to the trees $T_i$ and $T_j$. It is easy to check using Theorem 14 that for independent uniformly random spanning trees conditioned on either type of event, the respective resampling oracle generates independent uniformly random spanning trees.

Let us define the following lopsidependency graph, which potentially has more edges than necessary. The graph contains the following kinds of edges:

- $E^i_{ef} \backsim E^i_{e'f'}$ whenever $e \cup f$ intersects $e' \cup f'$;
- $E^i_{ef} \backsim E^{ij}_{e'}$ and $E^j_{ef} \backsim E^{ij}_{e'}$ whenever $e'$ intersects $e \cup f$;
- $E^{ij}_e \backsim E^{ij'}_{e'}$ and $E^{ij}_e \backsim E^{i'j}_{e'}$ whenever $e'$ intersects $e$.

We claim that the resampling oracle for any bad event can cause new bad events only in its neighborhood. This follows from the fact that the resampling oracle affects

[2] Subsequently to our work, several papers showed independently that for any proper edge-coloring by $n-1$ colors, there exist $\Omega(n)$ edge-disjoint rainbow spanning trees [26, 5, 40].

only the trees relevant to the event (in the superscript), and the only edges modified are those incident to those relevant to the event (in the subscript), due to Lemma 13.

Let us now verify the cluster expansion criterion, introduced as (CLL) in Section 1.4, so that we may apply Theorem 57. Let us assume that each color appears on at most $q$ edges, and we generate $t$ random spanning trees. We claim that the neighborhood of each bad event can be partitioned into 4 cliques of size at most $(n-1)(q+t-2)$.

First, consider an event of type $E^i_{ef}$. The neighborhood of $E^i_{ef}$ consists of:

- Events $E^i_{e'f'}$ where $e'$ or $f'$ shares a vertex with $e \cup f$. For each vertex $v \in e \cup f$, there are at most $(n-1)(q-1)$ events $E^i_{e'f'}$ where $v \in e' \cup f'$, since there are $n-1$ ways to pick an edge incident to $v$, and the number of other edges of the same color is at most $q-1$.
- Events $E^{ij}_{e'}$ where $e'$ intersects $e \cup f$. For each vertex $v \in e \cup f$, there are at most $(n-1)(t-1)$ events $E^{ij}_{e'}$ where $v \in e'$, since there are $(n-1)$ edges incident to $v$ and $t-1$ choices for the tree index $j$.

All these events can be covered by 4 cliques, one for each vertex $v \in e \cup f$, where the clique associated with a vertex $v$ contains all the events that involve some edge incident to $v$. By our definition of the dependency graph, all such events are pairwise neighbors. As discussed above, the number of such events is at most $(n-1)(q-1)+(n-1)(t-1)$. The cliques are not necessarily disjoint but we can make them disjoint trivially by keeping each event in exactly one clique.

Secondly, consider an event of type $E^{ij}_e$. The neighborhood of $E^{ij}_e$ consists of:

- Events $E^i_{e'f'}$ and $E^j_{e'f'}$ where $e$ intersects $e' \cup f'$. For each choice of $v \in e$ and either $i$ or $j$ in the superscript, the number of such events where $v \in e' \cup f'$ is at most $(n-1)(q-1)$, because there are $n-1$ ways to pick an edge incident to $v$ and at most $q-1$ other edges of the same color.
- Events $E^{i'j}_{e'}, E^{ij'}_{e'}$ where $e'$ intersects $e$. For each choice of $v \in e$ and either $i'j$ or $ij'$ in the superscript, the number of such events is at most $(n-1)(t-1)$, since there are $(n-1)$ edges incident to $v$ and $t-1$ choices for the second tree index.

Again, these events can be covered by 4 cliques, one associated with each choice of a vertex $v \in e$ and either $i$ or $j$ in the superscript. By our construction of the dependency graph, all such events are pairwise neighbors. As discussed above, the size of each clique is again at most $(n-1)(q-1)+(n-1)(t-1)$.

Considering the symmetry of the dependency graph, we set the variables for all events equal to $y^i_{ef} = y^{ij}_e = y$. The cluster expansion criteria will be satisfied if we set the parameters so that

$$p \leqslant \frac{y}{(1+(n-1)(q+t-2)y)^4} \leqslant \frac{y}{\sum_{I \subseteq \Gamma^+(E), I \in \mathsf{Ind}} y^I},$$

where $E$ denotes either $E^i_{ef}$ or $E^{ij}_e$. The second inequality holds due to the structure of the neighborhood of each event that we described above. We set $y = \beta p = 4\beta/n^2$ and assume $q+t-2 \leqslant \gamma n$. The assumption of the theorem allows $\gamma = \frac{1}{16}(\frac{3}{4})^3$ and we choose $\beta = (\frac{4}{3})^4$; the reader can verify that $\frac{\beta}{(1+4\gamma\beta)^4} = 1$. Therefore, using these relations,

$$p = \frac{\beta p}{(1+4\gamma\beta)^4} = \frac{y}{(1+\gamma n^2 y)^4} \leqslant \frac{y}{(1+(n-1)(q+t-2)y)^4}$$

which verifies the assumption of Theorem 57. Theorem 57 implies that MaximalSetResample terminates after $O((\sum y^i_{ef} + \sum y^{ij}_e)^2)$ resampling oracle calls with high probability. The total number of events here is $O(tqn^2 + t^2n^2) = O(n^4)$ and for each event the respective variable is $y = O(1/n^2)$. Therefore, the expected number of resampling oracle calls is $O(n^4)$. ∎

**4. Analysis of the algorithm.** The previous sections have explained the context of our resampling oracle setting, described some example resampling oracles, and illustrated them with an application. This section proves our main technical results, including analyses of the MaximalSetResample algorithm.

In Section 4.1, we begin with the basic notions necessary for our analysis and a coupling argument which forms the basis of all our algorithmic results. In Section 4.2, we introduce the independence polynomial of a graph and summarize its fundamental properties that are important for our analysis. In Section 4.3, we prove that our algorithm is efficient if Shearer's criterion is satisfied with an $\epsilon$ slack. In Section 4.4, we show that in some sense this assumption is not necessary, because every point satisfying Shearer's criterion has some slack available, and we quantify how large this slack is. Finally, we return to the weaker (but more practical) variants of the local lemma: the (GLL) and (CLL) criteria. We present new combinatorial connections between these criteria and Shearer's criterion, which in turn imply our main results on the efficiency of our algorithm under the (GLL) and (CLL) criteria (in Sections 4.5 and 4.6, respectively).

**4.1. Stable set sequences and the coupling argument.** An important notion in our analysis is that of *stable set sequences*. This concept originated in the work of Kolipaka and Szegedy [28], which builds on Shearer's work [45]. There are some similarities but also differences in how this concept is applied here: most notably, our stable set sequences grow forward in time, while the stable set sequences in [28] grow backward in time, as the Moser-Tardos analysis [35] does.

Definition 16. *One execution of the outer repeat loop in MaximalSetResample is called an* iteration. *For a sequence of non-empty sets $\mathcal{I} = (I_1, \dots, I_t)$, we say that the algorithm* follows *$\mathcal{I}$ if $I_s$ is the set resampled in iteration $s$ for $1 \leqslant s < t$, and $I_t$ is a set of the first $m$ events resampled in iteration $t$ for some $m \geqslant 1$ (a prefix of the maximal independent set constructed in iteration $t$).*

Recall that $\mathsf{Ind} = \mathsf{Ind}(G)$ denotes the independent sets (including the empty set) in the graph under consideration.

Definition 17. *$\mathcal{I} = (I_1, I_2, \dots, I_t)$ is called a* stable set sequence *if $I_1, \dots, I_t \in \mathsf{Ind}(G)$ and $I_{s+1} \subseteq \Gamma^+(I_s)$ for each $1 \leqslant s < t$. We call the sequence $\mathcal{I}$* proper *if each independent set $I_s$ is nonempty.*

Note that if $I_s = \varnothing$ for some $s$, then $I_t = \varnothing$ for all $t > s$. Therefore, the nonempty sets always form a prefix of the stable set sequence. An empty sequence is also considered to be a stable set sequence, of length 0.

Lemma 18. *If MaximalSetResample follows a sequence $\mathcal{J} = (J_1, \dots, J_t)$, then $\mathcal{J}$ is a stable set sequence.*

*Proof.* By construction, the set $J_s$ chosen in each iteration is independent in $G$. For each $i \in J_s$, we execute the resampling oracle $r_i$. Recall that $r_i$ executed on a satisfied event $E_i$ can only cause new events in the neighborhood $\Gamma^+(i)$ (and this neighborhood is not explored again until the following iteration). Since $J_s$ is a maximal independent set of satisfied events, all the events satisfied in the following

iteration are neighbors of some event in $J_s$, i.e., $J_{s+1} \subseteq \Gamma^+(J_s)$. In iteration $t$, this also holds since $J_t$ is still a set of satisfied events (not necessarily maximal). ∎

We use the following notation: For $i \in [n]$, $p_i = \Pr_\mu[E_i]$. For $S \subseteq [n]$, $p^S = \prod_{i \in S} p_i$. For a stable set sequence $\mathcal{I} = (I_1, \ldots, I_t)$, $p_\mathcal{I} = \prod_{s=1}^t p^{I_s}$. We relate stable set sequences to executions of the algorithm by the following coupling argument. Although the use of stable set sequences is inspired by [28], their coupling argument is different due to its backward-looking nature (similar to [35]), and their restriction to the variable model.

LEMMA 19. *For any proper stable set sequence $\mathcal{I} = (I_1, I_2, \ldots, I_t)$, the probability that the MaximalSetResample algorithm follows $\mathcal{I}$ is at most $p_\mathcal{I}$.*

*Proof.* Given $\mathcal{I} = (I_1, I_2, \ldots, I_t)$, let us consider the following "$\mathcal{I}$-checking" random process. We start with a random state $\omega \sim \mu$. In iteration $s$, we process the events of $I_s$ in the ascending order of their indices. For each $i \in I_s$, we check whether $\omega$ satisfies $E_i$; if not, we terminate. Otherwise, we apply the resampling oracle $r_i$ and replace $\omega$ by $r_i(\omega)$. We continue for $s = 1, 2, \ldots, t$. We say that the $\mathcal{I}$-checking process succeeds if every event is satisfied when checked and the process runs until the end.

By induction, the state $\omega$ after each resampling oracle call is distributed according to $\mu$: Assuming this was true in the previous step and conditioned on $E_i$ satisfied, we have $\omega \sim \mu|_{E_i}$. By assumption, the resampling oracle $r_i$ removes this conditioning and produces again a random state $r_i(\omega) \sim \mu$. Therefore, whenever we check event $E_i$, it is satisfied with probability $\Pr_\mu[E_i]$ (conditioned on the past). By a telescoping product of conditional probabilities, the probability that the $\mathcal{I}$-checking process succeeds is exactly $\prod_{s=1}^t \prod_{i \in I_s} \Pr_\mu[E_i] = \prod_{s=1}^t p^{I_s} = p_\mathcal{I}$.

To conclude, we argue that the probability that MaximalSetResample follows the sequence $\mathcal{I}$ is at most the probability that the $\mathcal{I}$-checking process succeeds. To see this, suppose that we couple MaximalSetResample and the $\mathcal{I}$-checking process, so they use the same source of randomness. In each iteration, if MaximalSetResample includes $i$ in $J_t$, it means that $E_i$ is satisfied. Both procedures apply the resampling oracle $r_i(\omega)$ and by coupling the distribution in the next iteration is the same. Therefore, the event that MaximalSetResample follows the sequence $\mathcal{I}$ is contained in the event that the $\mathcal{I}$-checking process succeeds, which happens with probability $p_\mathcal{I}$. ∎

The main difference between MaximalSetResample and the $\mathcal{I}$-checking process is that the stable set sequence $\mathcal{I}$ is not required to be maximal in any sense. So, even though MaximalSetResample is coupled with the $\mathcal{I}$-checking process, it is possible that MaximalSetResample resamples additional events not appearing in $\mathcal{I}$. It is also work observing that, when executing MaximalSetResample, we do *not* claim that the distribution of the current state $\omega \in \Omega$ is $\mu$ after each resampling oracle call. This would mean that the algorithm is not making any progress in its search for a state avoiding all events. It is only the $\mathcal{I}$-checking process that has this property.

DEFINITION 20. *Let* $\mathsf{Stab}$ *denote the set of all stable set sequences and* $\mathsf{Prop}$ *the set of proper stable set sequences. Let us denote by* $\mathsf{Stab}_\ell$ *the set of stable set sequences $(I_1, \ldots, I_\ell)$ of length $\ell$, and by* $\mathsf{Stab}_\ell(J)$ *the subset of* $\mathsf{Stab}_\ell$ *such that the first set in the sequence is $J$. Similarly, denote by* $\mathsf{Prop}_\ell$ *the set of proper stable set sequences of length $\ell$. For $\mathcal{I} = (I_1, \ldots, I_t) \in \mathsf{Prop}$, let us call $\sigma(\mathcal{I}) = \sum_{s=1}^t |I_s|$ the total size of the sequence.*

LEMMA 21. *The probability that MaximalSetResample runs for at least $\ell$ itera-*

*tions is at most* $\sum_{\mathcal{I}\in\mathsf{Prop}_\ell} p_{\mathcal{I}}$. *The probability that MaximalSetResample resamples at least* $s$ *events is at most* $\sum_{\mathcal{I}\in\mathsf{Prop}:\sigma(\mathcal{I})=s} p_{\mathcal{I}}$.

*Proof.* If the algorithm runs for at least $\ell$ iterations, it means that it follows some proper sequence $\mathcal{I} = (I_1, I_2, \ldots, I_\ell)$. By Lemma 19, the probability that the algorithm follows a particular stable set sequence $\mathcal{I}$ is at most $p_{\mathcal{I}}$. By the union bound, the probability that the algorithm runs for at least $\ell$ iterations is at most $\sum_{\mathcal{I}=(I_1,\ldots,I_\ell)\in\mathsf{Prop}} p_{\mathcal{I}}$.

Similarly, if the algorithm resamples at least $s$ events, it means that it follows some proper sequence $\mathcal{I}$ of total size $\sigma(\mathcal{I}) = s$. By the union bound, the probability of resampling at least $s$ events is upper-bounded by $\sum_{\mathcal{I}\in\mathsf{Prop}:\sigma(\mathcal{I})=s} p_{\mathcal{I}}$. ∎

We note that these bounds could be larger than 1 and thus vacuous. The events that "the algorithm follows $\mathcal{I} = (I_1, \ldots, I_\ell)$" are disjoint for different sequences of fixed total size $\sigma(\mathcal{I})$, while they could overlap for a fixed length $\ell$ (because we can take $I_\ell$ to be different prefixes of the sequence of events resampled in iteration $t$). In any case, the upper bound of $p_{\mathcal{I}}$ on each of the events could be quite loose.

**4.2. Preliminaries on Shearer's criterion.** In this section we discuss a strong version of the local lemma due to Shearer [45]. Shearer's lemma is based on certain forms of the multivariate independence polynomial. Recall that $p^I$ denotes $\prod_{i\in I} p_i$.

Definition 22. *Given a graph* $G$ *and values* $p_1, \ldots, p_n$, *define for each* $S \subseteq [n]$

$$q_S \;=\; q_S(p) \;=\; \sum_{\substack{I\in\mathsf{Ind}\\ S\subseteq I}} (-1)^{|I\setminus S|} p^I. \tag{4}$$

Note that $q_S = 0$ for $S \notin \mathsf{Ind}$. An alternative form of these polynomials that is also useful is obtained by summing over subsets of $S$.

Definition 23. *Given a graph* $G$ *and values* $p_1, \ldots, p_n$, *define*

$$\breve{q}_S \;=\; \breve{q}_S(p) \;=\; \sum_{\substack{I\in\mathsf{Ind}\\ I\subseteq S}} (-1)^{|I|} p^I.$$

The following set plays a fundamental role.

Definition 24. *Given a graph* $G$, *the Shearer region is the semialgebraic set*

$$\mathcal{S} \;=\; \{\, p \in (0,1)^n \,:\, \forall I \in \mathsf{Ind},\ q_I(p) > 0 \,\} \tag{5a}$$

$$\;=\; \{\, p \in (0,1)^n \,:\, \forall S \subseteq [n],\ \breve{q}_S(p) > 0 \,\} \tag{5b}$$

The equivalence between (5a) and (5b) is proven below in Claim 32.

Shearer's Lemma can be stated as follows.

Lemma 25 (Shearer [45]). *Let* $G$ *be a lopsidependency graph for the events* $E_1, \ldots, E_n$. *Let* $p_i = \Pr_\mu[E_i] \in (0,1)$. *If* $p \in \mathcal{S}$ *then* $\Pr_\mu[\bigcap_{i=1}^n \overline{E_i}] \geqslant q_\varnothing$.

It is known that Shearer's Lemma implies Theorem 1, as we will see in Section 4.5, and in fact gives the tight criterion under which all events can be avoided for a given dependency graph $G$. The polynomials $q_S(p)$ and $\breve{q}_S(p)$ have a natural interpretation in the Shearer region: There is a "tight instance" where $q_S(p)$ is the probability that the set of occurring events is exactly $S$, and $\breve{q}_S(p)$ is the probability that none of the events in $S$ occur. In particular, $q_\varnothing(p) = \breve{q}_{[n]}(p)$ is exactly the probability that no event occurs. (See [45] for more details.)

**4.2.1. Properties of independence polynomials.** In this section we summarize some of the important properties of these polynomials, most of which may be found in earlier work. Since some of the proofs are not easy to recover due to different notation or their analytic nature (in case of [44]), we provide short combinatorial proofs in the appendix for completeness.

CLAIM 26 (The "fundamental identity". Shearer [45], Scott-Sokal [44, Eq. (3.5)]). *For any $a \in S$, we have*

$$\breve{q}_S \;=\; \breve{q}_{S\setminus\{a\}} \;-\; p_a \cdot \breve{q}_{S\setminus\Gamma^+(a)}.$$

CLAIM 27 (Shearer [45], Scott-Sokal [44, Eq. (2.52)]). *For every $S \subseteq [n]$,*

$$\breve{q}_S \;=\; \sum_{Y\subseteq[n]\setminus S} q_Y.$$

CLAIM 28 (Shearer [45]).

$$\sum_{J\in\mathsf{Ind}} q_J \;=\; \sum_{S\subseteq[n]} q_S \;=\; 1.$$

CLAIM 29 (Scott-Sokal [44, Eq. (2.48)]). *For $I \in \mathsf{Ind}$,*

$$q_I \;=\; p^I \cdot \breve{q}_{[n]\setminus\Gamma^+(I)}.$$

LEMMA 30 (Kolipaka-Szegedy [28, Lemma 15]). *For any $I \in \mathsf{Ind}$*

$$q_I \;=\; p^I \cdot \sum_{S\subseteq\Gamma^+(I)} q_S.$$

CLAIM 31 (Simultaneous positivity of $q_S$ and $\breve{q}_S$). *Assume that $p \in [0,1]^n$. Then*

$$q_I \geqslant 0 \quad \forall I \in \mathsf{Ind} \qquad\Longrightarrow\qquad \breve{q}_S \geqslant q_\varnothing \quad \forall S \subseteq [n] \tag{6}$$

$$\breve{q}_S \geqslant 0 \quad \forall S \subseteq [n] \qquad\Longrightarrow\qquad q_I \geqslant p^{[n]} \cdot \breve{q}_{[n]} \quad \forall I \in \mathsf{Ind}. \tag{7}$$

CLAIM 32. *The two characterizations of the Shearer region, (5a) and (5b), are equivalent.*

CLAIM 33 (Monotonicity of $\breve{q}$, Scott-Sokal [44, Theorem 2.10]). *Let $p \in [0,1]^n$.*

$$\breve{q}_S(p) \geqslant 0 \quad \forall S \subseteq [n] \qquad\Longrightarrow\qquad \breve{q}_S(p') \geqslant \breve{q}_S(p) \quad \forall 0 \leqslant p' \leqslant p,\ \forall S \subseteq [n].$$

COROLLARY 34. *For any $p \in \mathcal{S}$ and $A \subseteq B \subseteq [n]$, we have $\breve{q}_A(p) \geqslant \breve{q}_B(p)$.*

CLAIM 35 (Log-submodularity of $\breve{q}_S$, Scott-Sokal [44, Corollary 2.27]). *For any $p \in \mathcal{S}$ and $A, B \subseteq [n]$, we have $\breve{q}_A \cdot \breve{q}_B \geqslant \breve{q}_{A\cup B} \cdot \breve{q}_{A\cap B}$.*

CLAIM 36 (Log-submodularity of $q_S$). *For any $p \in \mathcal{S}$ and $A, B \subseteq [n]$, we have $q_A \cdot q_B \geqslant q_{A\cup B} \cdot q_{A\cap B}$.*

CLAIM 37. *Suppose that $p \in \mathcal{S}$. For any set $S \subseteq [n]$,*

$$\sum_{J\subseteq S} \frac{q_J}{q_\varnothing} \;\leqslant\; \prod_{j\in S}\Big(1 + \frac{q_{\{j\}}}{q_\varnothing}\Big).$$

CLAIM 38. *If $q_\varnothing > 0$ then $\frac{q_{\{i\}}}{q_\varnothing} = \frac{\breve{q}_{[n]\setminus\{i\}}}{\breve{q}_{[n]}} - 1$.*

CLAIM 39 (Kolipaka-Szegedy [28, Theorem 5]). *If $(1+\epsilon)p \in \mathcal{S}$ then $\frac{q_{\{i\}}}{q_\varnothing} \leqslant \frac{1}{\epsilon}$ for each $i \in [n]$.*

**4.2.2. Connection to stable set sequences.** Kolipaka and Szegedy showed that stable set sequences relate to the independence polynomials $q_S$. The following is the crucial upper-bound for stable set sequences when Shearer's criterion holds. A proof appears in the appendix.

LEMMA 40 (Kolipaka-Szegedy [28]). *If $q_S \geqslant 0$ for all $S \subseteq [n]$ and $q_\varnothing > 0$, then*

$$\sum_{\mathcal{I} \in \mathsf{Stab}_\ell(J)} p_{\mathcal{I}} \;\leqslant\; \frac{q_J}{q_\varnothing} \qquad \forall J \in \mathsf{Ind}, \forall \ell \geqslant 1.$$

The inequality in Lemma 40 actually becomes an equality as $\ell \to \infty$. This stronger result is also due to Kolipaka and Szegedy [28, Theorem 14], and an alternative exposition may be found in our technical report [25]. This paper does not use the stronger result.

From Claim 28 and Lemma 40 we obtain immediately the following.

COROLLARY 41. *If $q_S \geqslant 0$ for all $S \subseteq [n]$ and $q_\varnothing > 0$,*

$$\sum_{\mathcal{I} \in \mathsf{Prop}} p_{\mathcal{I}} \;\leqslant\; \frac{1}{q_\varnothing}.$$

**Summary at this point.** By Lemma 21 and Corollary 41, MaximalSetResample produces a state in $\bigcap_{i=1}^n \overline{E_i}$ after at most $1/q_\varnothing$ iterations in expectation. However, this should not be viewed as a statement of efficiency. Shearer's Lemma proves that $\Pr_\mu[\bigcap_{i=1}^n \overline{E_i}] \geqslant q_\varnothing$ so, in expectation, $1/q_\varnothing$ independent samples from $\mu$ would also suffice to find a state in $\bigcap_{i=1}^n \overline{E_i}$.

Section 4.3 improves this analysis by assuming that Shearer's criterion holds with some slack. Section 4.4 then removes the need for that assumption — it argues that Shearer's criterion always holds with some slack, and provides quantitative bounds on that slack.

**4.3. Shearer's criterion with slack.** In this section we consider scenarios in which Shearer's criterion holds with a certain amount of slack. To make this formal, we will consider another vector $p'$ of probabilities with $p \leqslant p' \in \mathcal{S}$. For notational convenience, we will let $q'_S$ denote the value $q_S(p')$ and let $q_S$ denote $q_S(p)$ as before. Let us assume that Shearer's criterion holds with some slack in the following natural sense.

DEFINITION 42. *We say that $p \in (0,1)^n$ satisfies Shearer's criterion with coefficients $q'_S$ at a slack of $\epsilon$, if $p' = (1+\epsilon)p$ is still in the Shearer region $\mathcal{S}$ and $q'_S = q_S(p')$.*

THEOREM 43. *Recall that $p_i = \Pr_\mu[E_i]$. If the $p_i$ satisfy Shearer's criterion with coefficient $q'_\varnothing$ at a slack of $\epsilon \in (0,1)$, then the probability that MaximalSetResample resamples more than $\frac{2}{\epsilon}\big(\ln \frac{1}{q'_\varnothing} + t\big)$ events is at most $e^{-t}$.*

*Proof.* By Lemma 21, the probability that MaximalSetResample resamples more than $s$ events is at most $\sum_{\mathcal{I} \in \mathsf{Prop}:\sigma(\mathcal{I}) = \lceil s \rceil} p_{\mathcal{I}}$. By the slack assumption, we have

$$\Pr[\text{resample more than } s \text{ events}] \;\leqslant\; \sum_{\substack{\mathcal{I} \in \mathsf{Prop} \\ \sigma(\mathcal{I}) = \lceil s \rceil}} p_{\mathcal{I}} \;\leqslant\; (1+\epsilon)^{-s} \sum_{\substack{\mathcal{I} \in \mathsf{Prop} \\ \sigma(\mathcal{I}) = \lceil s \rceil}} p'_{\mathcal{I}}$$

since we have $p'_i = (1+\epsilon)p_i$ for each event appearing in a sequence $\mathcal{I}$. The hypothesis is that the probabilities $p'_i$ satisfy Shearer's criterion with a bound of $q'_\varnothing$. Consequently,

Corollary 41 implies that $\sum_{\mathcal{I}\in\mathsf{Prop}:\sigma(\mathcal{I})=[s]} p'_{\mathcal{I}} \leqslant \sum_{\mathcal{I}\in\mathsf{Prop}} p'_{\mathcal{I}} \leqslant 1/q'_{\varnothing}$. Thus, for $s = \frac{2}{\epsilon}\big(\ln\frac{1}{q'_{\varnothing}} + t\big)$ we obtain

$$\begin{aligned}\Pr[\text{resample more than } s \text{ events}] &\leqslant (1+\epsilon)^{-s}\frac{1}{q'_{\varnothing}}\\ &\leqslant e^{-s\epsilon/2}\frac{1}{q'_{\varnothing}}\\ &\leqslant e^{-(\ln(1/q'_{\varnothing})+t)}\frac{1}{q'_{\varnothing}} = e^{-t}.\end{aligned}$$

∎

In other words, the probability that MaximalSetResample requires more than $\frac{2}{\epsilon}\ln(1/q'_{\varnothing})$ resamplings decays exponentially fast; in particular the expected number of resampled events is $O\big(\frac{1}{\epsilon}\ln(1/q'_{\varnothing})\big)$. This appears significantly better than the trivial bound of $1/q_{\varnothing}$; still, it is not clear whether this bound can be considered "polynomial". In the following, we show that this leads in fact to efficient bounds, comparable to the best known bounds in the variable model.

Corollary 44. *If the $p_i$ satisfy Shearer's criterion with coefficients $q'_S$ at a slack of $\epsilon \in (0,1)$, then the probability that MaximalSetResample resamples more than*

$$\frac{2}{\epsilon}\left(\sum_{j=1}^{n}\ln\left(1+\frac{q'_{\{j\}}}{q'_{\varnothing}}\right)+t\right)$$

*events is at most $e^{-t}$.*

*Proof.* By Claim 28 and Claim 37, we have

$$\ln\frac{1}{q'_{\varnothing}} = \ln\sum_{J\subseteq[n]}\frac{q'_J}{q'_{\varnothing}} \leqslant \sum_{j=1}^{n}\ln\left(1+\frac{q'_{\{j\}}}{q'_{\varnothing}}\right).$$

The result follows from Theorem 43. ∎

Next, we provide a simplified bound that depends only on the amount of slack and the number of events. It is similar to the bound of $O(n/\epsilon)$ given by Kolipaka-Szegedy [28] in the variable model. These bounds may be considered efficient, depending on the parameter by which one measures efficiency. For example, in some applications $n$ can be exponentially large in another parameter, in which case the theorem does not guarantee efficiency.

Theorem 45. *If $p_1,\dots,p_n$ satisfy Shearer's criterion at a slack of $\epsilon\in(0,1)$, then the expected number of events resampled by MaximalSetResample is $O(\frac{n}{\epsilon}\log\frac{1}{\epsilon})$.*

*Proof.* Let $p' = (1+\epsilon/2)p$. By assumption, $(1+\epsilon/3)p' \leqslant (1+\epsilon)p \in \mathcal{S}$. So $p'$ still has $\epsilon/3$ slack and, by Claim 39, the coefficients $q'_S = q_S(p')$ satisfy $\frac{q'_{\{i\}}}{q'_{\varnothing}} \leqslant \frac{3}{\epsilon}$. The point $p$ satisfies Shearer's criterion with coefficients $q'_S$ at a slack of $\epsilon/2$ so, by Corollary 44, the probability that we resample more than $\frac{4}{\epsilon}(n\ln(1+\frac{3}{\epsilon})+t)$ events is at most $e^{-t}$. In expectation, we resample $O(\frac{n}{\epsilon}\log\frac{1}{\epsilon})$ events as claimed. ∎

**4.4. Quantification of slack in Shearer's criterion.** In the previous section, we proved a bound on the number of resamplings in the MaximalSetResample algorithm, provided that Shearer's criterion is satisfied with a certain slack. In fact, from Definition 24 one can observe that the Shearer region is an *open set* and therefore

there is always a certain amount of slack. However, how large a slack we can assume is not a priori clear. In particular, one can compare with Kolipaka-Szegedy [28] where a bound is proved on the expected number of events one has to resample in the variable model: If Shearer's criterion is satisfied with coefficients $q_S$, then the expected number of resamplings is at most $\sum_{i=1}^n q_{\{i\}}/q_\varnothing$ [28]. In this section, we prove that anywhere in the Shearer region, there is an amount of slack *inversely proportional to this quantity*, which leads to a bound similar to that of Kolipaka and Szegedy [28].

LEMMA 46. *Let $(p_1, \dots, p_n) \in (0,1)^n$ be a point in the Shearer region. Let $\epsilon = q_\varnothing(p)/(2\sum_{i=1}^n q_{\{i\}}(p))$ and $p'_i = (1+\epsilon)p_i$. Then $(p'_1, \dots, p'_n)$ is also in the Shearer region, and $q_\varnothing(p') \geqslant \frac{1}{2} q_\varnothing(p)$.*

Before proving the lemma, let us consider the partial derivatives of the $\breve{q}_S$ polynomials.

CLAIM 47. *For any $i \in S$,*

$$\frac{\partial \breve{q}_S}{\partial p_i} \;=\; -\breve{q}_{S\setminus\Gamma^+(i)}$$

*and for any $j \in S\setminus\Gamma^+(i)$,*

$$\frac{\partial^2 \breve{q}_S}{\partial p_i \partial p_j} \;=\; \breve{q}_{S\setminus\Gamma^+(i)\setminus\Gamma^+(j)}.$$

*For other choices of $i, j$, the partial derivatives are $0$. In particular, for any point in the Shearer region, $\frac{\partial \breve{q}_S}{\partial p_i} \leqslant 0$ and $\frac{\partial^2 \breve{q}_S}{\partial p_i \partial p_j} \geqslant 0$.*

Due to Claim 47, we may say that $\breve{q}_S(p_1, \dots, p_n)$ is "continuous supermodular" in the Shearer region.

*Proof.* For any $i \in S$, we have $\breve{q}_S = \breve{q}_{S\setminus\{i\}} - p_i \breve{q}_{S\setminus\Gamma^+(i)}$ by Claim 26. The polynomials $\breve{q}_{S\setminus\{i\}}$ and $\breve{q}_{S\setminus\Gamma^+(i)}$ do not depend on $p_i$ and hence $\frac{\partial \breve{q}_S}{\partial p_i}$ is equal to $-\breve{q}_{S\setminus\Gamma^+(i)}$. Repeating this argument one more time for $j \in S\setminus\Gamma^+(i)$, we get $\frac{\partial \breve{q}_S}{\partial p_i} = -\breve{q}_{S\setminus\Gamma^+(i)} = -\breve{q}_{S\setminus\Gamma^+(i)\setminus\{j\}} + p_j \breve{q}_{S\setminus\Gamma^+(i)\setminus\Gamma^+(j)}$. Again, $\breve{q}_{S\setminus\Gamma^+(i)\setminus\{j\}}$ and $\breve{q}_{S\setminus\Gamma^+(i)\setminus\Gamma^+(j)}$ do not depend on $p_j$ and hence $\frac{\partial^2 \breve{q}_S}{\partial p_i \partial p_j} = \breve{q}_{S\setminus\Gamma^+(i)\setminus\Gamma^+(j)}$.

Clearly, we have $\frac{\partial \breve{q}_S}{\partial p_i} = 0$ unless $i \in S$, and $\frac{\partial^2 \breve{q}_S}{\partial p_i \partial p_j} = 0$ unless $i \in S$ and $j \in S\setminus\Gamma^+(i)$. Since all the coefficients $\breve{q}_S$ are positive in the Shearer region, we have $\frac{\partial \breve{q}_S}{\partial p_i} \leqslant 0$ and $\frac{\partial^2 \breve{q}_S}{\partial p_i \partial p_j} \geqslant 0$ for all $i, j$. ∎

Now we can prove Lemma 46.

*Proof.* Consider the line segment from $p = (p_1, \dots, p_n)$ to $p' = (p'_1, \dots, p'_n)$ where $p'_i = (1+\epsilon)p_i$, $\epsilon = \frac{q_\varnothing}{2\sum_{i=1}^n q_{\{i\}}}$. Note that $p'_i \leqslant (1+\frac{q_\varnothing}{q_{\{i\}}})p_i = \frac{q_{\{i\}}+q_\varnothing}{q_{\{i\}}} p_i = \frac{\breve{q}_{[n]\setminus\{i\}}}{p_i \breve{q}_{[n]\setminus\Gamma^+(i)}} p_i \leqslant 1$ by Claim 27, Claim 29 and Corollary 34. Let us define

$$Q_\varnothing(\lambda) \;=\; q_\varnothing((1+\lambda)p_1, \dots, (1+\lambda)p_n).$$

Observe from Claim 27 that $q_\varnothing = \breve{q}_{[n]}$. By the chain rule, Claim 47 and Claim 29 we have

$$\frac{dQ_\varnothing}{d\lambda}\Big|_{\lambda=0} \;=\; \sum_{i=1}^n p_i \frac{\partial q_\varnothing}{\partial p_i} \;=\; -\sum_{i=1}^n p_i \breve{q}_{[n]\setminus\Gamma^+(i)} \;=\; -\sum_{i=1}^n q_{\{i\}}.$$

Assuming that $(1+\lambda)p = ((1+\lambda)p_1, \ldots, (1+\lambda)p_n)$ is in the Shearer region, we also have by Claim 47

$$\frac{d^2 Q_\varnothing}{d\lambda^2} \;=\; \sum_{i,j=1}^{n} \frac{\partial^2 q_\varnothing}{\partial p_i \partial p_j} p_i p_j \geqslant 0.$$

That is, $Q_\varnothing(\lambda)$ is a convex function for $\lambda \geqslant 0$ as long as $(1+\lambda)p$ is in the Shearer region. Our goal is to prove that this indeed happens for $\lambda \in [0, \epsilon]$.

Assume for the sake of contradiction that $(1+\lambda)p$ is not in the Shearer region for some $\lambda \in [0, \epsilon]$, and let $\lambda^*$ be the minimum such value (which exists since the complement of the Shearer region is closed). By Claim 31, anywhere in the Shearer region, $q_\varnothing = \breve{q}_{[n]}$ is the minimum of the $\breve{q}_S$ coefficients; hence by continuity it must be the case that $\breve{q}_{[n]}((1+\lambda^*)p)$ is the minimum coefficient among $\breve{q}_S((1+\lambda^*)p)$ for all $S \subseteq [n]$, and $Q_\varnothing(\lambda^*) = \breve{q}_{[n]}((1+\lambda^*)p) \leqslant 0$. On the other hand, by the minimality of $\lambda^*$, $Q_\varnothing(\lambda)$ is positive and convex on $[0, \lambda^*)$ and therefore

$$Q_\varnothing(\lambda^*) \;\geqslant\; Q_\varnothing(0) + \lambda^* \frac{dQ_\varnothing}{d\lambda}\Big|_{\lambda=0} \;=\; q_\varnothing - \lambda^* \sum_{i=1}^{n} q_{\{i\}} \;\geqslant\; q_\varnothing - \epsilon \sum_{i=1}^{n} q_{\{i\}} \;=\; \frac{1}{2} q_\varnothing \;>\; 0,$$

which is a contradiction. Therefore, $Q_\varnothing(\lambda)$ is positive and convex for all $\lambda \in [0, \epsilon]$. By the same computation as above, $Q_\varnothing(\epsilon) \geqslant \frac{1}{2} q_\varnothing$. ∎

This implies our main algorithmic result under Shearer's criterion.

THEOREM 48. *Let $E_1, \ldots, E_n$ be events and let $p_i = \Pr_\mu[E_i]$. Suppose that the three subroutines described in Section 1.1.1 exist. If $p \in \mathcal{S}$ then the probability that MaximalSetResample resamples more than $4 \sum_{i=1}^{n} \frac{q_{\{i\}}}{q_\varnothing} \big( \sum_{j=1}^{n} \ln(1 + \frac{q_{\{j\}}}{q_\varnothing}) + 1 + t \big)$ events is at most $e^{-t}$.*

We note that the corresponding result in the variable model [28] was that the expected number of resamplings is at most $\sum_{i=1}^{n} \frac{q_{\{i\}}}{q_\varnothing}$. Here, we obtain a bound which is at most quadratic in this quantity.

*Proof.* Directly from Theorem 43 and Lemma 46: Given $p$ in the Shearer region, Lemma 46 implies that $p$ in fact satisfies Shearer's criterion with a bound of $q'_\varnothing \geqslant \frac{q_\varnothing}{2}$ at a slack of $\epsilon = \frac{q_\varnothing}{2} / \sum_{i=1}^{n} q_{\{i\}}$. By Theorem 43, the probability that MaximalSetResample resamples more than $s$ events is at most $e^{-t}$, where

$$s \;=\; \frac{2}{\epsilon} \left( \ln \frac{1}{q'_\varnothing} + t \right) \;\leqslant\; \frac{4}{q_\varnothing} \sum_{i=1}^{n} q_{\{i\}} \left( \ln \frac{1}{q_\varnothing} + 1 + t \right).$$

Using Claim 37, we can replace $\ln \frac{1}{q_\varnothing}$ by $\sum_{j=1}^{n} \ln(1 + \frac{q_{\{j\}}}{q_\varnothing})$. ∎

**4.5. The General LLL criterion.** Shearer's Lemma (Lemma 25) is a strengthening of the original Lovász Local Lemma (Theorem 1): if $p_1, \ldots, p_n$ satisfy (GLL) then they must also satisfy Shearer's criterion $p \in \mathcal{S}$. Nevertheless, there does not seem to be a direct proof of this fact in the literature. Shearer [45] indirectly proves this fact by showing that, when $p \notin \mathcal{S}$ it is possible that $\Pr[\bigcap_{i=1}^{n} \overline{E_i}] = 0$, so the contrapositive of Theorem 1 implies that (GLL) cannot hold. Scott and Sokal prove this fact using analytic properties of the partition function [44, Corollary 5.3]. In this section we establish this fact by an elementary, self-contained proof.

We then establish Theorem 3, our algorithmic form of Theorem 1 in the general framework of resampling oracles.

Lemma 49. *Suppose that $p$ satisfies (GLL). Then, for every $S \subseteq [n]$ and $a \in S$, we have*

$$\frac{\breve{q}_S}{\breve{q}_{S\setminus\{a\}}} \;\geqslant\; 1 - x_a.$$

Corollary 50 ((GLL) implies Shearer). *If $p$ satisfies (GLL) then $p \in \mathcal{S}$.*

*Proof.* For any $S \subseteq [n]$, write it as $S = \{s_1, \ldots, s_k\}$. Induction yields

$$\frac{\breve{q}_S}{\breve{q}_\varnothing} \;=\; \prod_{i=1}^{k} \frac{\breve{q}_{\{s_1,\ldots,s_i\}}}{\breve{q}_{\{s_1,\ldots,s_{i-1}\}}} \;\geqslant\; \prod_{a\in S}(1-x_a) \;>\; 0.$$

The claim follows since $\breve{q}_\varnothing = 1$. ∎

Corollary 51. *If $p$ satisfies (GLL) then $\frac{q_{\{a\}}}{q_\varnothing} \leqslant \frac{x_a}{1-x_a}$.*

*Proof.* Lemma 49 yields $\frac{\breve{q}_{[n]-a}}{\breve{q}_{[n]}} \leqslant \frac{1}{1-x_a}$, so the result follows from Claim 38. ∎

*Proof* (of Lemma 49). We proceed by induction on $|S|$. The base case, $S = \varnothing$, is trivial: there is no $a \in S$ to choose. Consider $S \neq \varnothing$ and an element $a \in S$. By Claim 26, we have $\breve{q}_S = \breve{q}_{S\setminus\{a\}} - p_a \breve{q}_{S\setminus\Gamma^+(a)}$. By the inductive hypothesis applied iteratively to the elements of $(S\setminus\{a\})\setminus(S\setminus\Gamma^+(a)) = \Gamma(a)\cap S$, we have

$$\breve{q}_{S\setminus\{a\}} \;\geqslant\; \breve{q}_{S\setminus\Gamma^+(a)} \prod_{i\in\Gamma(a)\cap S}(1-x_i).$$

Therefore, we can write

$$\breve{q}_S \;=\; \breve{q}_{S\setminus\{a\}} - p_a\breve{q}_{S\setminus\Gamma^+(a)} \;\geqslant\; \breve{q}_{S\setminus\{a\}}\left(1 - \frac{p_a}{\prod_{i\in\Gamma(a)\cap S}(1-x_i)}\right).$$

By the claim's hypothesis, $p_a \leqslant x_a \prod_{i\in\Gamma(a)}(1-x_i) \leqslant x_a \prod_{i\in\Gamma(a)\cap S}(1-x_i)$, so we conclude that $\breve{q}_S \geqslant (1-x_a)\breve{q}_{S\setminus\{a\}}$. ∎

These results, together with our analysis of Shearer's criterion with slack (Corollary 44), immediately provide an analysis under the assumption that (GLL) holds with slack. However, this connection to Shearer's criterion allows us to prove more.

We show that our algorithm is in fact efficient even when the (GLL) criterion is tight. This holds because tightness of (GLL) does not imply tightness of Shearer's criterion. In fact, Shearer's criterion is *never tight*: as we argued already, it defines an open set, and Section 4.4 derives a quantitative bound on the slack that is always available under Shearer's criterion.

Theorem 52. *Let $E_1, \ldots, E_n$ be events and let $p_i = \Pr_\mu[E_i]$. Suppose that the three subroutines described in Section 1.1.1 exist. If $p$ satisfies (GLL) then the probability that MaximalSetResample resamples more than $4\sum_{i=1}^n \frac{x_i}{1-x_i}(\sum_{j=1}^n \ln\frac{1}{1-x_j} + 1 + t)$ events is at most $e^{-t}$.*

*If (GLL) is satisfied with a slack of $\epsilon \in (0,1)$, i.e., $(1+\epsilon)p_i \leqslant x_i\prod_{j\in\Gamma(i)}(1-x_j)$, then with probability at least $1-e^{-t}$, MaximalSetResample resamples no more than $\frac{2}{\epsilon}(\sum_{j=1}^n \ln\frac{1}{1-x_j} + t)$ events.*

*Proof.* The first part follows directly from Theorem 48, since Corollary 50 shows that $p \in \mathcal{S}$ and Corollary 51 shows that $\frac{q_{\{i\}}}{q_\varnothing} \leqslant \frac{x_i}{1-x_i}$. The second part follows from Corollary 44, using again that $\frac{q_{\{i\}}}{q_\varnothing} \leqslant \frac{x_i}{1-x_i}$. ∎

Theorem 3 follows immediately from Theorem 52.

**4.6. The cluster expansion criterion.** Recall that Section 1.4 introduced the cluster expansion criterion, which often gives improved quantitative bounds compared to the General LLL. One example of this is the application discussed in Section 3.4.

For convenience, let us restate the cluster expansion criterion here. Given parameters $y_1, \ldots, y_n$, define the notation

$$Y_S = \sum_{\substack{I \subseteq S \\ I \in \mathsf{Ind}}} y^I \qquad \forall S \subseteq [n].$$

The cluster expansion criterion for a vector $p \in [0,1]^n$, with respect to a graph $G$, is

$$\text{(CLL)} \qquad \exists y_1, \ldots, y_n > 0 \qquad \text{such that} \qquad p_i \leqslant y_i / Y_{\Gamma^+(i)}.$$

This criterion was introduced in the following non-constructive form of the LLL.

THEOREM 53 (Bissacot et al. [7]). *Let $E_1, \ldots, E_n$ be events. Let $G$ be a lopsidependency graph for these events and let $p_i = \Pr_\mu[E_i]$. If $p$ and $G$ satisfy (CLL) then $\Pr_\mu[\bigcap_{i=1}^n \overline{E_i}] > 0$.*

To see that this strengthens the original LLL (Theorem 1), one may verify that (GLL) implies (CLL): if $p_i \leqslant x_i \prod_{j \in \Gamma(i)} (1 - x_j)$, we can take $y_i = \frac{x_i}{1 - x_i}$ (so $1 - x_i = \frac{1}{1 + y_i}$) and then use the simple bound

$$\sum_{\substack{I \subseteq \Gamma^+(i) \\ I \in \mathsf{Ind}}} y^I \leqslant \sum_{I \subseteq \Gamma^+(i)} y^I = \prod_{j \in \Gamma^+(i)} (1 + y_j).$$

On the other hand, Shearer's Lemma (Lemma 25) strengthens Theorem 53, in the sense that (CLL) implies $p \in \mathcal{S}$. This fact was established by Bissacot et al. [7] by analytic methods that relied on earlier results [18]. In this section we establish this fact by a new proof that is elementary and self-contained.

An algorithmic form of Theorem 53 in the variable model was proven by Pegden [38]. In fact, that result is subsumed by the algorithm of Kolipaka and Szegedy in Shearer's setting, since (CLL) implies $p \in \mathcal{S}$. In this section, we prove a new algorithmic form of Theorem 53 in the general framework of resampling oracles.

To begin, we establish the following connection between the $y_i$ parameters and the $\breve{q}_S$ polynomials. For convenience, let us introduce the notation $S^c = [n] \backslash S$, $S + a = S \cup \{a\}$ and $S - a = S \backslash \{a\}$.

LEMMA 54. *Suppose that $p$ satisfies (CLL). Then, for every $S \subseteq [n]$ and $a \in S$, we have*

$$\frac{\breve{q}_S}{\breve{q}_{S-a}} \geqslant \frac{Y_{S^c}}{Y_{(S-a)^c}}.$$

The proof is in Section 4.6.1 below.

COROLLARY 55 ((CLL) implies Shearer). *If $p$ satisfies (CLL) then $p \in \mathcal{S}$.*

*Proof.* For any $S \subseteq [n]$, write it as $S = \{s_1, \ldots, s_k\}$. Applying Lemma 54 repeatedly, we obtain

$$\frac{\breve{q}_S}{\breve{q}_\varnothing} = \prod_{i=1}^k \frac{\breve{q}_{\{s_1,\ldots,s_i\}}}{\breve{q}_{\{s_1,\ldots,s_{i-1}\}}} \geqslant \prod_{i=1}^k \frac{Y_{\{s_1,\ldots,s_i\}^c}}{Y_{\{s_1,\ldots,s_{i-1}\}^c}} = \frac{Y_{S^c}}{Y_{[n]}} > 0$$

since $Y_T > 0$ for all $T \subseteq [n]$ under the (CLL) criterion. Recall that $\breve{q}_\varnothing = 1$. Hence $\breve{q}_S > 0$ for all $S \subseteq [n]$, which means that $p$ is in the Shearer region. ∎

Corollary 56. *If $p$ satisfies (CLL) then $\frac{q_{\{a\}}}{q_\varnothing} \leqslant y_a$.*

*Proof.* Lemma 54 yields $\frac{\breve{q}_{[n]-a}}{\breve{q}_{[n]}} \leqslant \frac{Y_{([n]-a)^c}}{Y_{[n]^c}} = 1 + y_a$, so the result follows from Claim 38. ∎

These corollaries lead to our algorithmic result under the cluster expansion criterion. The following theorem subsumes Theorem 6 and adds a statement under the assumption of slack.

Theorem 57. *Let $E_1, \dots, E_n$ be events and let $p_i = \Pr_\mu[E_i]$. Suppose that the three subroutines described in Section 1.1.1 exist. If $p$ satisfies (CLL) then, with probability at least $1 - e^{-t}$, MaximalSetResample resamples no more than*

$$4 \sum_{i=1}^{n} y_i \left( \sum_{j=1}^{n} \ln(1 + y_j) + 1 + t \right) \text{ events.}$$

*If (CLL) is satisfied with a slack of $\epsilon \in (0, 1)$, i.e., $(1+\epsilon)p_i \leqslant y_i / Y_{\Gamma^+(i)}$, then with probability at least $1 - e^{-t}$, MaximalSetResample resamples no more than*

$$\frac{2}{\epsilon} \left( \sum_{j=1}^{n} \ln(1 + y_j) + t \right) \text{ events.}$$

*Proof.* The first statement follows directly from Theorem 48, since Corollary 55 shows that $p \in \mathcal{S}$ and Corollary 56 shows that $\frac{q_{\{i\}}}{q_\varnothing} \leqslant y_i$. Next assume that (CLL) is satisfied with $\epsilon$ slack. We apply Corollary 55 and Corollary 56 to the point $p' = (1+\epsilon)p$, obtaining that $p' \in \mathcal{S}$ and $q'_{\{j\}}/q'_\varnothing \leqslant y_j$, where $q'_S$ denotes $q_S(p')$. The second statement then follows directly from Corollary 44. ∎

### 4.6.1. Proof of Lemma 54.

Claim 58 (The "fundamental identity" for $Y$). *$Y_A = Y_{A-a} + y_a Y_{A \setminus \Gamma^+(a)}$ for all $a \in A$.*

*Proof.* Every summand $y^J$ on the left-hand side either appears in $Y_{A-a}$ if $a \notin J$, or can be written as $y_a \cdot y^B$ where $B = J \setminus \Gamma^+(a)$, in which case it appears as a summand in $y_a Y_{A \setminus \Gamma^+(a)}$. ∎

Claim 59 (Log-subadditivity of $Y$). *$Y_{A \cup B} \leqslant Y_A \cdot Y_B$ for any $A, B \subseteq [n]$.*

*Proof.* It suffices to consider the case that $A$ and $B$ are disjoint, as replacing $B$ with $B \setminus A$ decreases the right-hand side and leaves the left-hand side unchanged. Every summand $y^J$ on the left-hand side can be written as $y^{J'} \cdot y^{J''}$ with $J' = J \cap A$ and $J'' = J \cap B$. The product $y^{J'} \cdot y^{J''}$ appears as a summand on the right-hand side, and all other summands are non-negative. ∎

*Proof* (of Lemma 54). We proceed by induction on $|S|$. The base case is $S = \{a\}$. In that case we have $\frac{\breve{q}_{\{a\}}}{\breve{q}_\varnothing} = \breve{q}_{\{a\}} = 1 - p_a$. On the other hand, by the two claims above and (CLL), we have

$$Y_{[n]} \;=\; Y_{[n]-a} + y_a Y_{[n] \setminus \Gamma^+(a)} \;\geqslant\; Y_{[n]-a} + p_a Y_{\Gamma^+(a)} Y_{[n] \setminus \Gamma^+(a)} \;\geqslant\; Y_{[n]-a} + p_a Y_{[n]}.$$

Therefore, $\frac{Y_{[n]-a}}{Y_{[n]}} \leqslant 1 - p_a$ which proves the base case.

We prove the inductive step by similar manipulations. By Claim 26, we have

$$\frac{\breve{q}_S}{\breve{q}_{S-a}} \;=\; 1 - p_a \frac{\breve{q}_{S \setminus \Gamma^+(a)}}{\breve{q}_{S-a}}.$$

The inductive hypothesis applied repeatedly to the elements of $S \cap \Gamma(a)$ yields

$$1 - p_a \frac{\breve{q}_{S \setminus \Gamma^+(a)}}{\breve{q}_{S-a}} \;\geqslant\; 1 - p_a \frac{Y_{(S \setminus \Gamma^+(a))^c}}{Y_{(S-a)^c}} \;=\; 1 - p_a \frac{Y_{S^c \cup \Gamma^+(a)}}{Y_{S^c + a}}.$$

By the two claims above and (CLL), we have

$$Y_{S^c+a} \;=\; Y_{S^c} + y_a Y_{S^c \setminus \Gamma^+(a)} \;\geqslant\; Y_{S^c} + p_a Y_{\Gamma^+(a)} Y_{S^c \setminus \Gamma^+(a)} \;\geqslant\; Y_{S^c} + p_a Y_{S^c \cup \Gamma^+(a)}.$$

We conclude that

$$\frac{\breve{q}_S}{\breve{q}_{S-a}} \;\geqslant\; 1 - p_a \frac{Y_{S^c \cup \Gamma^+(a)}}{Y_{S^c+a}} \;\geqslant\; 1 - \frac{Y_{S^c+a} - Y_{S^c}}{Y_{S^c+a}} = \frac{Y_{S^c}}{Y_{(S-a)^c}}. \qquad \square$$

**5. Conclusions.** We have shown that the Lovász Local Lemma can be made algorithmic in the abstract framework of resampling oracles. This framework captures the General LLL as well as Shearer's Lemma in the existential sense, and leads to efficient algorithms for the primary examples of probability spaces and events satisfying lopsidependency that have been considered in the literature (as surveyed in [31]).

Our algorithmic form of the General LLL, stated in Theorem 3, performs more resampling operations than the Moser-Tardos algorithm [35]. Specifically, our algorithm uses $O\big(\sum_{i=1}^n \frac{x_i}{1-x_i} \sum_{j=1}^n \log \frac{1}{1-x_j}\big)$ resampling operations, which is roughly quadratically worse than the $\sum_{i=1}^n \frac{x_i}{1-x_i}$ bound of Moser-Tardos [35]. Similarly, our algorithmic result under Shearer's condition (Theorem 48) uses $O\big(\sum_{i=1}^n \frac{q_{\{i\}}}{q_\varnothing} \sum_{j=1}^n \ln(1 + \frac{q_{\{j\}}}{q_\varnothing})\big)$ resampling operations, which is roughly quadratically worse than the $\sum_{i=1}^n \frac{q_{\{i\}}}{q_\varnothing}$ bound of Kolipaka-Szegedy [28]. Can this quadratic loss be eliminated?

One way to prove that result would be to prove an analogue of the witness tree lemma, which is a centerpiece of the Moser-Tardos analysis [35]. The witness tree lemma has other advantages, for example in deriving parallel and deterministic algorithms. Unfortunately, the witness tree lemma is not true in the general setting of resampling oracles [25]. It is, however, true in the variable model [35] as well as in the setting of random permutations [24]. Is there a variant of our framework which can handle the same scenarios that we do, but in which the witness tree lemma is true?

Subsequent to the initial publication of this paper, Kolmogorov [30] and Iliopoulos [27] have made progress on this question. Their framework based on the notion of "commutativity" supports a witness tree lemma and captures the scenarios of permutations [24, 25] and perfect matchings [25]. However, developing a framework that supports a witness tree lemma and can handle the spanning tree scenario of Section 3.2 remains an open question.

**Acknowledgements.** We thank Mohit Singh for discussions at the early stage of this work, David Harris for suggesting the statement of Theorem 14, and the anonymous referees for several suggestions, including improved parameters in Theorem 15.

REFERENCES

[1] Dimitris Achlioptas and Themis Gouleakis. Algorithmic improvements of the Lovász local lemma via cluster expansion. In *Proceedings of FSTTCS*, 2012.

[2] Dimitris Achlioptas and Fotis Iliopoulos. Focused stochastic local search and the Lovász local lemma. In *Proc. of* $27^{th}$ *ACM-SIAM SODA*, 2016.

[3] Dimitris Achlioptas and Fotis Iliopoulos. Random walks that find perfect objects and the Lovász local lemma. *Journal of the ACM*, 63(3):22:1–22:29, 2016.

[4] Saieed Akbari and Alireza Alipour. Multicolored trees in complete graphs. *J. Graph Theory*, 54(3):221–232, 2007.

[5] József Balogh, Hong Liu, and Richard Montgomery. Rainbow spanning trees in properly coloured complete graphs. *Discrete Applied Mathematics*, 247:97–101, 2018.

[6] Anton Bernshteyn. The local cut lemma. *Eur. J. Comb.*, 63:95–114, 2017.

[7] R. Bissacot, R. Fernández, A. Procacci, and B. Scoppola. An improvement of the Lovász local lemma via cluster expansion. *Combin. Probab. Comput.*, 20:709–719, 2011.

[8] Julia Böttcher, Yoshiharu Kohayakawa, and Aldo Procacci. Properly coloured copies and rainbow copies of large graphs with small maximum degree. *Random Structures and Algorithms*, 40(4), 2012.

[9] Andrei Broder. Generating random spanning trees. In *Proceedings of FOCS*, pages 442–447, 1989.

[10] Richard A. Brualdi and Susan Hollingsworth. Multicolored trees in complete graphs. *J. Combin. Theory Ser. B*, 68, 1996.

[11] James M. Carraher, Stephen G. Hartke, and Paul Horn. Edge-disjoint rainbow spanning trees in complete graphs. *European Journal of Combinatorics*, 57:71–84, 2016.

[12] Karthekeyan Chandrasekaran, Navin Goyal, and Bernhard Haeupler. Deterministic algorithms for the Lovász local lemma. *SIAM Journal on Computing*, 42(6), 2013.

[13] Kai-Min Chung, Seth Pettie, and Hsin-Hao Su. Distributed algorithms for the Lovász local lemma and graph coloring. In *Proceedings of PODC*, 2014.

[14] Vida Dujmovic, Gwenaël Joret, Jakub Kozik, and David R. Wood. Nonrepetitive colouring via entropy compression. *Combinatorica*, 36(6):661–686, 2016.

[15] Paul Erdös and László Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In A. Hajnal et al., editor, *Infinite and finite sets*, volume 10 of *Colloquia Mathematica Societatis János Bolyai*, pages 609–628. North-Holland, Amsterdam, 1975.

[16] Paul Erdös and Joel Spencer. The Lopsided Lovász Local Lemma and Latin transversals. *Discrete Applied Mathematics*, 30:151–154, 1991.

[17] Louis Esperet and Aline Parreau. Acyclic edge-coloring using entropy compression. *Eur. J. Comb.*, 34(6):1019–1027, 2013.

[18] R. Fernández and A. Procacci. Cluster expansion for abstract polymer models: New bounds from an old approach. *Comm. Math. Phys*, 274:123–140, 2007.

[19] Heidi Gebauer, Tibor Szabó, and Gábor Tardos. The local lemma is tight for SAT. In *Proceedings of SODA*, 2011.

[20] Ioannis Giotis, Lefteris M. Kirousis, Kostas I. Psaromiligkos, and Dimitrios M. Thilikos. Acyclic edge coloring through the Lovász Local Lemma. *Theor. Comput. Sci.*, 665:40–50, 2017.

[21] Daniel Gonçalves, Mickaël Montassier, and Alexandre Pinlou. Entropy compression method applied to graph colorings. *CoRR*, abs/1406.4380, 2014.

[22] Bernhard Haeupler, Barna Saha, and Aravind Srinivasan. New constructive aspects of the Lovász local lemma. *Journal of the ACM*, 58(6), 2011.

[23] David G. Harris. Lopsidependency in the Moser-Tardos framework: Beyond the Lopsided Lovász Local Lemma. In *Proceedings of* $26^{th}$ *ACM-SIAM SODA*, 2015.

[24] David G. Harris and Aravind Srinivasan. A constructive algorithm for the Lovász Local Lemma on permutations. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 907–925, 2014.

[25] Nicholas J. A. Harvey and Jan Vondrák. An algorithmic proof of the Lovász local lemma via resampling oracles. *CoRR*, 1504.02044, 2015.

[26] Paul Horn. Rainbow spanning trees in complete graphs colored by one-factorizations. *Journal of Graph Theory*, 87(3):333–346, 2018.

[27] Fotis Iliopoulos. Backtracking and commutative algorithms for the LLL. *CoRR*, 1704.02796, 2017.

[28] Kashyap Kolipaka and Mario Szegedy. Moser and Tardos meet Lovász. In *Proceedings of STOC*, 2011.

[29] Kashyap Kolipaka, Mario Szegedy, and Yixin Xu. A sharper local lemma with improved applications. In *Proceedings of APPROX/RANDOM*, 2012.

[30] Vladimir Kolmogorov. Commutativity in the algorithmic Lovász local lemma. In *57th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 780–787, 2016.

[31] Lincoln Lu, Austin Mohr, and László Székely. Quest for negative dependency graphs. *Recent Advances in Harmonic Analysis and Applications*, 25:243–258, 2013.

[32] Austin Mohr. *Applications of the lopsided Lovász local lemma regarding hypergraphs*. PhD thesis, University of South Carolina, 2013.

[33] Robin Moser. *Exact Algorithms for Constraint Satisfaction Problems*. PhD thesis, ETH Zürich,

2012.

[34] Robin A. Moser. A constructive proof of the Lovász local lemma. In *Proceedings of STOC*, 2009.

[35] Robin A. Moser and Gábor Tardos. A constructive proof of the general Lovász Local Lemma. *Journal of the ACM*, 57(2), 2010.

[36] Sokol Ndreca, Aldo Procacci, and Benedetto Scoppola. Improved bounds on coloring of graphs. *European Journal of Combinatorics*, 33(4), 2012.

[37] Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48:498–532, 1994.

[38] Wesley Pegden. An extension of the Moser-Tardos algorithmic local lemma. *SIAM J. Discrete Math*, 28:911–917, 2014.

[39] Guillem Perarnau and Oriol Serra. Rainbow perfect matchings in complete bipartite graphs: Existence and counting. *Combinatorics, Probability and Computing*, 22:783–799, 2013.

[40] Alexey Pokrovskiy and Benny Sudakov. Linearly many rainbow trees in properly edge-coloured complete graphs. *J. Comb. Theory, Ser. B*, 132:134–156, 2018.

[41] Jakub Przybylo. On the facial thue choice index via entropy compression. *Journal of Graph Theory*, 77(3):180–189, 2014.

[42] Jakub Przybylo, Jens Schreyer, and Erika Skrabul'áková. On the facial thue choice number of plane graphs via entropy compression method. *Graphs and Combinatorics*, 32(3):1137–1153, 2016.

[43] Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2004.

[44] Alexander D. Scott and Alan D. Sokal. The Repulsive Lattice Gas, the Independent-Set Polynomial, and the Lovász Local Lemma. *Journal of Statistical Physics*, 118(5):1151–1261, 2005.

[45] James B. Shearer. On a problem of Spencer. *Combinatorica*, 5(3), 1985.

[46] Joel Spencer. Asymptotic lower bounds for Ramsey functions. *Discrete Mathematics*, 20:69–76, 1977.

## Appendix A. Proofs omitted from Section 4.2.

*Proof* (of Claim 26). Every independent set $I \subseteq S$ either contains $a$ or does not. In addition, if $a \in I$ then $I$ is independent iff $I\backslash\{a\}$ is an independent subset of $S\backslash\Gamma^+(a)$. ∎

*Proof* (of Claim 27). By definition of $q_Y$,

$$\sum_{Y\subseteq[n]\backslash S} q_Y \;=\; \sum_{Y\subseteq[n]\backslash S}\; \sum_{\substack{I\in\mathsf{Ind}\\ Y\subseteq I}} (-1)^{|I\backslash Y|}p^I \;=\; \sum_{I\in\mathsf{Ind}} p^I \sum_{Y\subseteq I\backslash S} (-1)^{|I\backslash Y|}.$$

If $I\backslash S \neq \varnothing$ then the last alternating sum is zero. Therefore, the sum simplifies to $\sum_{I\in\mathsf{Ind}:I\subseteq S}(-1)^{|I|}p^I = \breve{q}_S$ as required. ∎

*Proof* (of Claim 28). Set $S = \varnothing$ in Claim 27 and use the fact that $\breve{q}_\varnothing = 1$. ∎

*Proof* (of Claim 29). Given $I \in \mathsf{Ind}$, each independent set $J \supseteq I$ can be written uniquely as $J = I \cup K$ where $K$ is independent and $K \cap \Gamma^+(I) = \varnothing$. So,

$$q_I \;=\; \sum_{J\in\mathsf{Ind}:I\subseteq J} (-1)^{|J\backslash I|}p^J \;=\; p^I \sum_{\substack{K\in\mathsf{Ind}\\ K\subseteq[n]\backslash\Gamma^+(I)}} (-1)^{|K|}p^K \;=\; p^I\cdot\breve{q}_{[n]\backslash\Gamma^+(I)}.$$

∎

*Proof* (of Lemma 30). By Claim 29 and Claim 27, we have

$$q_I \;=\; p^I\cdot\breve{q}_{[n]\backslash\Gamma^+(I)} \;=\; p^I \sum_{S\subseteq\Gamma^+(I)} q_S,$$

as required. ∎

*Proof* (of Claim 31). (6) follows from Claim 27 (since $q_Y = 0$ for $Y \notin \mathsf{Ind}$). To see (7), first note that $q_I \geqslant 0$ for all $I \in \mathsf{Ind}$, by Claim 29. Consequently, by Claim 27, $\breve{q}_{[n]} = \min_S \breve{q}_S$. Clearly, $p^{[n]} = \min_I p^I$. It follows from Claim 29 again that $q_I = p^I\cdot\breve{q}_{[n]\backslash\Gamma^+(I)} \geqslant p^{[n]}\cdot\breve{q}_{[n]}$. ∎

*Proof* (of Claim 32). By Claim 31, if $q_\varnothing > 0$ and $q_S \geqslant 0\ \forall S \subseteq [n]$, then $\breve{q}_S > 0$ for all $S \subseteq [n]$. Conversely, if $\breve{q}_S > 0$ for all $S \subseteq [n]$, then $q_I \geqslant p^{[n]}\breve{q}_{[n]} > 0$ for all $I \in \mathsf{Ind}$. ∎

*Proof* (of Claim 33). First consider the case that $p$ and $p'$ differ only in coordinate $i$. For any $S \subseteq [n]$, Claim 26 implies that $\frac{\partial}{\partial p_i}\breve{q}_S(p) = -\breve{q}_{S\backslash\Gamma^+(i)}(p)$ and $\frac{\partial^2}{\partial p_i^2}\breve{q}_S = 0$. Thus,

$$\breve{q}_S(p') \;=\; \breve{q}_S(p) + (p_i - p_i')\cdot\breve{q}_{S\backslash\Gamma^+(i)}(p) \;\geqslant\; \breve{q}_S(p).$$

The case that $p'$ and $p$ differ in multiple coordinates is handled by induction. ∎

*Proof* (of Claim 35). We claim that for any $a \in S \subseteq T$, we have

$$\frac{\breve{q}_S}{\breve{q}_{S\backslash\{a\}}} \;\geqslant\; \frac{\breve{q}_T}{\breve{q}_{T\backslash\{a\}}}. \tag{8}$$

By induction, this implies that for any $R \subseteq S$, $\frac{\breve{q}_S}{\breve{q}_{S\backslash R}} \geqslant \frac{\breve{q}_T}{\breve{q}_{T\backslash R}}$. We obtain the claim above by setting $S = A$, $T = A \cup B$, and $R = A\backslash B$.

We prove (8) again by induction, on $|T|$. For $|T| = 1$, the statement is trivial. Let $|T| > 1$. By Claim 26, we have

$$\breve{q}_S \;=\; \breve{q}_{S\backslash\{a\}} - p_a\breve{q}_{S\backslash\Gamma^+(a)}$$

and

$$\breve{q}_T \;=\; \breve{q}_{T\setminus\{a\}} - p_a \breve{q}_{T\setminus\Gamma^+(a)}.$$

Let us denote $S\cap\Gamma^+(a)=\{a,s_1,\dots,s_k\}$. We apply (8) to strict subsets of $S$ and $T$, to obtain

$$\begin{aligned}\frac{\breve{q}_{S\setminus\Gamma^+(a)}}{\breve{q}_{S\setminus\{a\}}} \;&=\; \prod_{i=1}^{k}\frac{\breve{q}_{S\setminus\{a,s_1,\dots,s_{i-1},s_i\}}}{\breve{q}_{S\setminus\{a,s_1,\dots,s_{i-1}\}}}\\ &\leqslant\; \prod_{i=1}^{k}\frac{\breve{q}_{T\setminus\{a,s_1,\dots,s_{i-1},s_i\}}}{\breve{q}_{T\setminus\{a,s_1,\dots,s_{i-1}\}}} \;=\; \frac{\breve{q}_{T\setminus(S\cap\Gamma^+(a))}}{\breve{q}_{T\setminus\{a\}}} \;\leqslant\; \frac{\breve{q}_{T\setminus\Gamma^+(a)}}{\breve{q}_{T\setminus\{a\}}},\end{aligned}$$

where in the last step we used the monotonicity of $\breve{q}_T$ in $T$ (from Corollary 34). This implies (8):

$$\frac{\breve{q}_S}{\breve{q}_{S\setminus\{a\}}} \;=\; 1-p_a\frac{\breve{q}_{S\setminus\Gamma^+(a)}}{\breve{q}_{S\setminus\{a\}}} \;\geqslant\; 1-p_a\frac{\breve{q}_{T\setminus\Gamma^+(a)}}{\breve{q}_{T\setminus\{a\}}} \;=\; \frac{\breve{q}_T}{\breve{q}_{T\setminus\{a\}}}.$$ ∎

*Proof* (of Claim 36). We can assume $A\cup B\in\mathsf{Ind}$; otherwise the right-hand side is zero. By Claim 29, we have

$$q_A\cdot q_B \;=\; p^A\breve{q}_{[n]\setminus\Gamma^+(A)}\cdot p^B\breve{q}_{[n]\setminus\Gamma^+(B)}.$$

By Claim 35,

$$\breve{q}_{[n]\setminus\Gamma^+(A)}\cdot\breve{q}_{[n]\setminus\Gamma^+(B)} \;\geqslant\; \breve{q}_{[n]\setminus(\Gamma^+(A)\cup\Gamma^+(B))}\cdot\breve{q}_{[n]\setminus(\Gamma^+(A)\cap\Gamma^+(B))}.$$

Next we use the fact that $\Gamma^+(A)\cup\Gamma^+(B)=\Gamma^+(A\cup B)$, and $\Gamma^+(A)\cap\Gamma^+(B)\supseteq\Gamma^+(A\cap B)$. Therefore, by the monotonicity of $\breve{q}_S$ (from Corollary 34),

$$\breve{q}_{[n]\setminus\Gamma^+(A)}\cdot\breve{q}_{[n]\setminus\Gamma^+(B)} \;\geqslant\; \breve{q}_{[n]\setminus\Gamma^+(A\cup B)}\cdot\breve{q}_{[n]\setminus\Gamma^+(A\cap B)}.$$

Also, $p^Ap^B=p^{A\cup B}p^{A\cap B}$. Using Claim 29 one more time, we obtain

$$q_A\cdot q_B \;\geqslant\; p^{A\cup B}\breve{q}_{[n]\setminus\Gamma^+(A\cup B)}\cdot p^{A\cap B}\breve{q}_{[n]\setminus\Gamma^+(A\cap B)} \;=\; q_{A\cup B}\cdot q_{A\cap B}.$$ ∎

*Proof* (of Claim 37). The proof is by induction on $S$, the case $|S|\leqslant 1$ being trivial. Fix any $s\in S$. Claim 36 implies that $q_{J\cup\{s\}}\cdot q_\varnothing\leqslant q_{\{s\}}\cdot q_J$ for any $J\subseteq S\setminus\{s\}$. Summing over $J$ yields

$$\sum_{J\subseteq S\setminus\{s\}}\frac{q_{J\cup\{s\}}}{q_\varnothing} \;\leqslant\; \frac{q_{\{s\}}}{q_\varnothing}\sum_{J\subseteq S\setminus\{s\}}\frac{q_J}{q_\varnothing}.$$

Adding $\sum_{J\subseteq S\setminus\{s\}}\frac{q_J}{q_\varnothing}$ to both sides yields

$$\sum_{J\subseteq S}\frac{q_J}{q_\varnothing} \;\leqslant\; \Big(1+\frac{q_{\{s\}}}{q_\varnothing}\Big)\sum_{J\subseteq S\setminus\{s\}}\frac{q_J}{q_\varnothing}.$$

The claim follows by induction. ∎

*Proof* (of Claim 38). By Claim 27,

$$1+\frac{q_{\{i\}}}{q_\varnothing}=\frac{q_\varnothing+q_{\{i\}}}{q_\varnothing}=\frac{\breve{q}_{[n]\setminus\{i\}}}{\breve{q}_{[n]}}.$$

*Proof* (of Claim 39). Note that $\breve{q}_{[n]\setminus\{i\}}(p)$ does not depend on $p_i$, while $\breve{q}_{[n]}(p)$ is linear in $p_i$. Also, both quantities are equal at $p_i = 0$: we have $\breve{q}_{[n]}(p_1, \ldots, 0 \cdot p_i, \ldots, p_n) = \breve{q}_{[n]\setminus\{i\}}(p)$. Since $(1+\epsilon)p \in \mathcal{S}$, we know that $\breve{q}_{[n]}(p_1, \ldots, (1+\epsilon)p_i, \ldots, p_n) \geqslant 0$. By linearity, $\breve{q}_{[n]}(p) \geqslant \frac{\epsilon}{1+\epsilon}\breve{q}_{[n]\setminus\{i\}}(p)$. Claim 38 then implies that $\frac{q_{\{i\}}}{q_\varnothing} \leqslant \frac{1}{\epsilon}$. ∎

*Proof* (of Lemma 40). We proceed by induction: for $\ell = 1$, there is only one such stable set sequence $\mathcal{I} = (J)$. By Lemma 30, we have $q_J = p^J \sum_{S \subseteq \Gamma^+(J)} q_S \geqslant p^J q_\varnothing$. (Recall that $q_S \geqslant 0$ for all $S \subseteq [n]$.) Hence, $p_{(J)} = p^J \leqslant q_J/q_\varnothing$.

The inductive step: every stable set sequence starting with $J$ has the form $\mathcal{I} = (J, J', \ldots)$ where $J' \subseteq \Gamma^+(J)$. Therefore,

$$\sum_{\mathcal{I} \in \mathsf{Stab}_\ell(J)} p_{\mathcal{I}} \;=\; p^J \sum_{\substack{J' \in \mathsf{Ind} \\ J' \subseteq \Gamma^+(J)}} \;\sum_{\mathcal{I} \in \mathsf{Stab}_{\ell-1}(J')} p_{\mathcal{I}}. \tag{9}$$

By the inductive hypothesis, $\sum_{\mathcal{I} \in \mathsf{Stab}_{\ell-1}(J')} p_{\mathcal{I}} \leqslant q_{J'}/q_\varnothing$. Also, recall that $q_{J'} = 0$ if $J' \notin \mathsf{Ind}$. Therefore,

$$\sum_{\mathcal{I} \in \mathsf{Stab}_\ell(J)} p_{\mathcal{I}} \leqslant p^J \sum_{J' \subseteq \Gamma^+(J)} \frac{q_{J'}}{q_\varnothing} = \frac{q_J}{q_\varnothing}$$

using Lemma 30 to obtain the last equality. ∎