Distrib. Comput. (2017) 30:261–280
DOI 10.1007/s00446-016-0287-6

Distributed algorithms for the Lovász local lemma and graph coloring

Kai-Min Chung[1] · **Seth Pettie**[2] · **Hsin-Hao Su**[3]

Received: 19 May 2016 / Accepted: 21 October 2016 / Published online: 21 November 2016

Abstract The Lovász local lemma (LLL), introduced by Erdős and Lovász in 1975, is a powerful tool of the probabilistic method that allows one to prove that a set of n "bad" events do not happen with non-zero probability, provided that the events have limited dependence. However, the LLL itself does not suggest how to find a point avoiding all bad events. Since the works of Alon (Random Struct Algorithms 2(4):367–378, 1991) and Beck (Random Struct Algorithms 2(4):343–365, 1991) there has been a sustained effort to find a constructive proof (i.e. an algorithm) for the LLL or weaker versions of it. In a major breakthrough Moser and Tardos (J ACM 57(2):11, 2010) showed that a point avoiding all bad events can be found efficiently. They also proposed a distributed/parallel version of their algorithm that requires $O(\log^2 n)$ rounds of communication in a distributed network. In this paper we provide two new distributed algorithms for the LLL that improve on both the efficiency and simplicity of the Moser–Tardos algorithm. For clarity we express our results in terms of the symmetric LLL though both algorithms deal with the asymmetric version as well. Let p bound the probability of any bad event and d be the maximum degree in the dependency graph of the bad events. When $epd^2 < 1$ we give a truly simple LLL algorithm running in $O(\log_{1/epd^2} n)$ rounds. Under the weaker condition $ep(d+1) < 1$, we give a slightly slower algorithm running in $O(\log^2 d \cdot \log_{1/ep(d+1)} n)$ rounds. Furthermore, we give an algorithm that runs in *sublogarithmic* rounds under the condition $p \cdot f(d) < 1$, where $f(d)$ is an exponential function of d. Although the conditions of the LLL are locally verifiable, we prove that any distributed LLL algorithm requires $\Omega(\log^* n)$ rounds. In many graph coloring problems the existence of a valid coloring is established by one or more applications of the LLL. Using our LLL algorithms, we give logarithmic-time distributed algorithms for frugal coloring, defective coloring, coloring girth-4 (triangle-free) and girth-5 graphs, edge coloring, and list coloring.

Keywords Lovász local lemma · Distributed algorithms · Randomized algorithms · Coloring · Locality

A preliminary version of this paper appeared in the 33rd Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC). Pettie and Su are supported by NSF Grants CCF-0746673, CCF-1217338, CNS-1318294, CCF-1514383, and a grant from the US-Israel Binational Science Foundation. Part of the work was done while visiting MADALGO at Aarhus University, supported by Danish National Research Foundation Grant DNRF84. Chung was supported by NSF Grants CNS-1217821, CCF-1214844, and R. Pass's Sloan Fellowship.

✉ Hsin-Hao Su
hsinhao@csail.mit.edu

Kai-Min Chung
kmchung@iis.sinica.edu.tw

Seth Pettie
pettie@umich.edu

[1] Academia Sinica, Taipei, Taiwan

[2] University of Michigan, Ann Arbor, MI, USA

[3] MIT, Cambridge, MA, USA

1 Introduction

Consider a system $\mathcal{P}$ of independent random variables and a set $\mathcal{A}$ of n *bad* events, where each $A \in \mathcal{A}$ depends solely on some subset $\mathrm{vbl}(A) \subseteq \mathcal{P}$. For example, in a hypergraph 2-coloring instance, $\mathcal{P}$ represents the vertex colors and $\mathcal{A}$ the events in which an edge is monochromatic. The dependency graph $G_{\mathcal{A}} = (\mathcal{A}, \{(A, B) \mid \mathrm{vbl}(A) \cap \mathrm{vbl}(B) \neq \emptyset\})$ includes edges between events if and only if they depend on at least one common variable. Let $\Gamma(A)$ be A's neighborhood in $G_{\mathcal{A}}$

and $\Gamma^+(A) = \Gamma(A) \cup \{A\}$ be its inclusive neighborhood. The (general, asymmetric) LLL states [14,41] that if there is a function $x : \mathcal{A} \to (0, 1)$ such that

$$\Pr(A) \leq x(A) \cdot \prod_{B \in \Gamma(A)} (1 - x(B))$$

then $\Pr(\bigcap_{A \in \mathcal{A}} \overline{A}) > 0$, that is, there is a *satisfying assignment* to the underlying variables in which no bad events occur. The symmetric LLL is a useful corollary of the general LLL. If p and d are such that $\Pr(A) \leq p$ and $|\Gamma(A)| \leq d$ for all A, and $ep(d + 1) < 1$, then $\Pr(\bigcap_{A \in \mathcal{A}} \overline{A}) > 0$. For example, consider a hypergraph in which each edge contains k vertices and intersects at most $d < 2^{k-1}/e - 1$ other edges. Under a uniformly random color assignment $\mathcal{P} \to \{\text{red, blue}\}$ the probability an edge is monochromatic is $p = 2^{-(k-1)}$, so $ep(d + 1) < 1$. The symmetric LLL proves the *existence* of a satisfying color assignment but does not yield an efficient algorithm to find one. Beginning with Alon [1] and Beck [7], a long line of research has sought to find efficient (and ideally deterministic) algorithms for computing satisfying assignments [1,7,9,11,16–19,22,28,31–34,42]. Most of these results required a major weakening of the standard symmetric LLL constraint $ep(d + 1) < 1$. In many applications we consider, the bad events are that the sum of $d^{\Theta(1)}$ random variables deviates away from its expectation. So the probability they are violated is often bounded by Chernoff-type tail bounds, e.g. $\exp(-d^{\Theta(1)})$.

In a relatively recent breakthrough, Moser and Tardos [33] gave an *algorithmic* proof of the general asymmetric LLL, with no weakening of the parameters. Their algorithm is simple though the analysis is not trivial. At initialization the algorithm chooses a random assignment to the variables $\mathcal{P}$. Call an event $A \in \mathcal{A}$ *violated* if it occurs under the current assignment to the variables. Let $\mathcal{F} \subseteq \mathcal{A}$ be the set of violated events. The algorithm repeatedly chooses some $A \in \mathcal{F}$ and *resamples* the variables in $\text{vbl}(A)$, until $\mathcal{F} = \emptyset$.

The distributed LLL problem We consider Linial's LOCAL model [35] of distributed computation in which the distributed network is identical to the dependency graph. In other words, each node $A \in \mathcal{A}$ hosts a processor, which is aware of n, the degree bound d, and its neighborhood $\Gamma(A)$. Computation proceeds in synchronized rounds in which each node may send an unbounded message to its neighbors. *Time* is measured by the number of rounds; computation local to each node is free. Upon termination each node A must commit to an assignment to its variables $\text{vbl}(A)$ that is consistent with its neighbors, i.e., the nodes must collectively agree on a satisfying assignment to $\mathcal{P}$ avoiding all bad events. We consider the LOCAL model because we will need to send the assignment of $\text{vbl}(A)$ in one message.

Moser and Tardos proposed a parallel version of their resampling algorithm (Algorithm 1), which can easily be implemented in the LOCAL model. Let $G_{\mathcal{F}}$ be the graph induced by the violated events $\mathcal{F}$ under the current variable assignment. They proved that $O(\log_{1/ep(d+1)} n)$ iterations of Algorithm 1 suffice to avoid all bad events with probability $1 - 1/\text{poly}(n)$, i.e., $O(\log n)$ iterations suffice if $ep(d + 1)$ is bounded away from 1.[1] (For the sake of a simpler presentation we shall state many results in the symmetric LLL language. Our algorithms and Moser–Tardos algorithm work for the asymmetric LLL as well.) Moser and Tardos suggested using Luby's randomized MIS algorithm [27], which runs in $\Theta(\log n)$ rounds w.h.p. (which can also be achieved by [2]), for a total running time of $\Theta(\log n \cdot \log_{1/ep(d+1)} n)$. This is, intuitively, a very wasteful LLL algorithm since nodes spend nearly all their time computing MISs rather than performing resampling steps. For certain values of d the running time can be improved by plugging in an MIS algorithm running in $O(d + \log^* n)$ time [5] or $O(\log^2 d) + \exp(O(\sqrt{\log \log n}))$ time w.h.p. [6].[2] However, it is not possible to find an MIS in constant time. Kuhn, Moscibroda, and Wattenhofer [23] gave an $\Omega\left(\min\left\{\frac{\log d}{\log \log d}, \sqrt{\frac{\log n}{\log \log n}}\right\}\right)$ lower bound on the complexity of MIS and other symmetry-breaking problems.

Initialize a random assignment to the variables $\mathcal{P}$.
while $\mathcal{F} \neq \emptyset$ **do**
  Compute a maximal independent set $\mathcal{I}$ in $G_{\mathcal{F}}$.
  Resample each variable in $\text{vbl}(\mathcal{I}) = \bigcup_{A \in \mathcal{I}} \text{vbl}(A)$.
end while

Algorithm 1: The Moser–Tardos parallel resampling algorithm. Here $\mathcal{F}$ is the set of bad events occurring under the current variable assignment and $G_{\mathcal{F}}$ is the dependency graph induced by $\mathcal{F}$.

New results We give a new distributed LLL algorithm in the Moser–Tardos resampling framework that avoids the computation of MISs altogether. Due to its simplicity we are happy to display the algorithm in its entirety. We assume that nodes possess unique IDs, which could be assigned in an adversarial manner. Let $\Gamma_{\mathcal{F}}(A)$ be A's neighborhood in $G_{\mathcal{F}}$.

One can see that $\mathcal{I}$ is computed in one round: each node A tells its neighbors whether $A \in \mathcal{F}$ under the current variable assignment. Once A receives messages from all neighbors it can determine if $\text{ID}(A)$ is a local minimum in $G_{\mathcal{F}}$. We prove that under the slightly stronger criterion $epd^2 < 1$, this

[1] Note that $\log_{1/ep(d+1)} n$ could be sublogarithmic or superlogarithmic depending on how close $ep(d + 1)$ is to 0 or 1.

[2] These MIS algorithms are significantly more complex than Luby's and use larger messages.

Initialize a random assignment to the variables $\mathcal{P}$

while $\mathcal{F} \neq \emptyset$ **do**
  Let $\mathcal{I} = \{A \in \mathcal{F} \mid \mathrm{ID}(A) = \min\{\mathrm{ID}(B) \mid B \in \Gamma^+_{\mathcal{F}}(A)\}\}$
  Resample $\mathrm{vbl}(\mathcal{I}) = \bigcup_{A\in\mathcal{I}} \mathrm{vbl}(A)$.
end while

Algorithm 2: A simple distributed LLL algorithm

algorithm halts in $O(\log_{1/epd^2} n)$ steps w.h.p. Most applications of the LLL satisfy the $epd^2 < 1$ criterion, though not all. We give another distributed LLL algorithm in the resampling framework that finds a satisfying assignment in $O(\log^2 d \cdot \log_{1/ep(d+1)} n)$ time under the usual $ep(d+1) < 1$ criterion.

We show that faster algorithms exist when the condition $ep(d+1) < 1$ is replaced by a stronger condition $p \cdot f(d) < 1$, where $f(d)$ is a faster growing function than $e(d+1)$. However, it is not clear whether there exists $f(d)$ so that the LLL can be solved in sublogarithmic time in n, independent of d. Moser and Tardos observed that any parallel algorithm in the resampling framework requires $\Omega(\log_{1/p} n)$ resampling steps, even if the dependency graph has no edges. We combine the resampling framework with a locality approach to give an $O(\log n/\log\log n)$ algorithm for an exponential function $f(d)$. On the other hand, we prove that no constant time distributed LLL algorithm exists and that the LLL for any $f(d)$ requires $\Omega(\log^* n)$ time.

New applications Existential results in graph coloring [29] (those taking the *Rödl nibble* approach) can often be phrased as distributed algorithms in which each step succeeds with some tiny but non-zero probability, as guaranteed by the LLL. By using our distributed LLL algorithms we are able to solve a number of graph coloring problems in $O(\log n)$ time or faster.[3] Some of these applications require minor changes to existing algorithms while others are quite involved. Below Δ is the maximum degree, and $\epsilon > 0$ an arbitrarily small parameter.

Frugal coloring A k-frugal vertex coloring is one in which each color appears at most k times in the neighborhood of any vertex. Pemmaraju and Srinivasan [36] showed the existence of $(\Delta+1)$-colorings that are $O(\log^2\Delta/\log\log\Delta)$-frugal, and proved that $(\log\Delta \cdot \log n/\log\log n)$-frugal colorings could be computed in $O(\log n)$ time. With some modifications to their proof we show that a $O(\log^2\Delta/\log\log\Delta)$-frugal $(\Delta+1)$-coloring can be computed in $O(\log n)$ time. Notice that the best existential bound on the frugality for $(\Delta+1)$-coloring is $O(\log\Delta/\log\log\Delta)$ by Molloy and Reed [30].

Hind, Molloy, and Reed [21] showed there exist β-frugal, $O\left(\Delta^{1+\frac{1}{\beta}}\right)$-colorings by using the asymmetric LLL. We show how to turn their proof into a distributed algorithm that runs in $O(\log n \cdot \log^2\Delta)$ time.

Girth 4 and 5 In prior work [37] we proved that triangle-free graphs have $(4+\epsilon)\Delta/\ln\Delta$-colorings and gave $\log^{1+o(1)} n$ time algorithms for $(4+\epsilon)\Delta/\ln\Delta$-coloring triangle-free graphs and $(1+\epsilon)\Delta/\ln\Delta$-coloring girth-5 graphs. Here we prove that both problems can be solved in $O(\log n)$ time.

Edge coloring Dubhashi et al. [12] gave a $(1+\epsilon)\Delta$ edge-coloring algorithm running in $O(\log n)$ time, provided that $\Delta = (\log n)^{1+\Omega(1)}$ is sufficiently large relative to n. In [13], Elkin, Pettie, and Su applied our LLL algorithm to show that $(1+\epsilon)\Delta$ edge-coloring can be obtained in $O(\log^*\Delta + \log n/\Delta^{1-o(1)})$ rounds for $\Delta \geq \Delta_\epsilon$, where Δ_ϵ is a sufficiently large constant depending on ϵ.

List-coloring Suppose each vertex is issued a list of $(1+\epsilon)D > D_\epsilon$ colors such that each color appears in at most D lists in the neighborhood of any vertex, where D_ϵ is a sufficiently large constant depending on ϵ. (D need not be close to the degree Δ.) Reed and Sudakov [39] proved that $(1+\epsilon)D$-list-colorings exist. We show how to construct them in $O(\log^* D + \log n/D^{1-o(1)})$ time. Furthermore, for *any* D and any constant $\epsilon > 0$, we show that $(2e+\epsilon)D$ list coloring can be solved in $O(\log n)$ time.

Defective coloring An f-defective coloring is one in which a vertex may share its color with up to f neighbors. Barenboim and Elkin [4], and implicitly, Kuhn and Wattenhofer [24] gave an $O(1)$ time procedure to compute a $O(\log n)$-defective $O(\Delta/\log n)$-coloring. We prove that for any $f > 0$, an f-defective $O(\Delta/f)$-coloring can be computed in $O((\log n)/f)$ time.

2 Preliminaries

Let $\Gamma^r(A)$ be the r-neighborhood of A (the set of nodes at distance at most r from A, excluding A) and $\Gamma^{r+}(A) = \Gamma^r(A) \cup \{A\}$ be its inclusive r-neighborhood. A node set in the subscript indicates a restriction of the neighborhood to that set, e.g., $\Gamma^{2+}_{\mathcal{F}}(A) = \Gamma^{2+}(A) \cap \mathcal{F}$.

Consider an execution of a Moser–Tardos-type resampling algorithm. Let $C : \mathbb{N} \to \mathcal{A}$ be such that $C(i)$ is the ith event selected by the algorithm for resampling; C is called the *record* of the execution. (If the algorithm selects events

[3] Suppose H is both the distributed network and the graph to be colored. When invoking the LLL, the dependency graph $G_{\mathcal{A}}$ is not *identical* to H. Typically bad events in $\mathcal{A}$ are associated with H-vertices and two bad events are adjacent in $G_{\mathcal{A}}$ only if the corresponding vertices are at distance $O(1)$ in H. Thus, a distributed LLL algorithm for $G_{\mathcal{A}}$ can be simulated in H with an $O(1)$ slowdown.

in independent batches then the events in each batch can be listed arbitrarily.) A *witness tree* $\tau = (T, \sigma_T)$ is a finite rooted tree where $\sigma_T : V(T) \to \mathcal{A}$ labels each vertex in T with an event such that the children of $u \in T$ receive labels from $\Gamma^+(\sigma_T(u))$. A *2-witness tree* $\tau = (T, \sigma_T)$ is defined in the same way except that the children of $u \in T$ may receive labels from $\Gamma^{2+}(\sigma_T(u))$. A witness tree (or 2-witness tree) is *proper* if the children of a vertex receive distinct labels.

Given a record C, the witness tree $\tau_C(t)$ is constructed as follows. First, create a root node labelled $C(t)$. Looking backward in time, for each $i = t-1, t-2, \ldots, 1$, check if an existing node is labeled with an event from $\Gamma^+(C(i))$. If so, let u be one of the *deepest* such nodes. Create a new node v labeled $C(i)$ and make it a child of u. Given a witness tree τ, we say τ *occurs in* C if there exists an index t such that $\tau_C(t) = \tau$. Moser and Tardos proved the following lemma:

Lemma 1 *Let τ be a fixed witness tree and C be the record produced by the algorithm.*

1. If τ occurs in C, then τ is proper.
2. The probability that τ occurs in C is at most $\prod_{v \in V(\tau)} \Pr(\sigma_T(v))$.

Similarly, for $r \geq 2$, we can define an r-witness tree $\tau^r_C(t)$ in the same way except that in each step we attach a node labelled $C(i)$ to the deepest node among nodes labelled $\Gamma^{r+}(C(i))$. Also, we say τ r-occurs in C if there exists $t \in \mathbb{N}$ such that $\tau^r_C(t) = \tau$. Then Lemma 2 holds analogously:

Lemma 2 *Let τ be a fixed r-witness tree and C be the record produced by the algorithm.*

1. If τ r-occurs in C, then τ is proper.
2. The probability that τ r-occurs in C is at most $\prod_{v \in V(\tau)} \Pr(\sigma_T(v))$.

3 Algorithms

Recall that the parallel/distributed Moser–Tardos algorithm iteratively selects maximal independent sets (MIS) of violated events for resampling. They proved that if there is some slack in the general LLL preconditions then the algorithm terminates in $O(\log n)$ rounds of MIS.

Theorem 1 (Moser and Tardos) *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a finite set of events determined by these variables. If there exists an assignment of reals $x : \mathcal{A} \to (0, 1)$ such that*

$$\forall A \in \mathcal{A} : \Pr(A) \leq (1-\epsilon)x(A) \prod_{B \in \Gamma(A)} (1 - x(B)),$$

then the probability any bad event occurs after k resampling rounds of Algorithm 1 is at most $(1-\epsilon)^k \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}$.

In other words, if $x(A)$ is bounded away from 1 then $O(\log_{\frac{1}{1-\epsilon}} n)$ resampling rounds suffice, w.h.p. A distributed implementation of this algorithm takes $O(\log_{\frac{1}{1-\epsilon}} n \cdot \mathrm{MIS}(n, d))$, where d is the maximum degree of $G_{\mathcal{A}}$ and $\mathrm{MIS}(n, d)$ is the time needed to find an MIS in an n-vertex degree-d graph. It is known that $\mathrm{MIS}(n, d) = \Omega\left(\min\left\{\frac{\log d}{\log\log d}, \sqrt{\frac{\log n}{\log\log n}}\right\}\right)$ [23]. Our algorithms avoid the computation of MISs. In Sect. 3.1 we analyze the simple distributed LLL algorithm presented in the introduction, which requires slightly weakening the general LLL conditions. In Sect. 3.2 we present an algorithm that works for the standard LLL conditions but is slower by a $O(\log^2 d)$ factor.

3.1 A simple distributed algorithm

Recall that in each round of Algorithm 2, a violated event $A \in \mathcal{F}$ is selected for resampling if $\mathrm{ID}(A)$ is a local minimum in the violated subgraph $G_{\mathcal{F}}$. In order to analyze this algorithm in the witness tree framework we must establish some connection between the depth of witness trees and the number of rounds of resampling. Lemma 3 will let us make such a connection.

Lemma 3 *Suppose an event A is resampled in round $j > 1$ of Algorithm 2. There must exist some $B \in \Gamma^{2+}(A)$ resampled in round $j-1$.*

Proof Let $\mathcal{F}'$ and $\mathcal{F}$ be the violated event sets just before and after the resampling step at round $j-1$. If A is not in $\mathcal{F}'$ but is in $\mathcal{F}$ then its variables $\mathrm{vbl}(A)$ must have been changed in round $j-1$, which could only occur if some $B \in \Gamma(A)$ were resampled. Now suppose A is in both $\mathcal{F}'$ and $\mathcal{F}$. It was not resampled in round $j-1$ but was in round j, meaning $\mathrm{ID}(A)$ is not a local minimum in $\Gamma_{\mathcal{F}'}(A)$ but is a local minimum in $\Gamma_{\mathcal{F}}(A)$. This implies that some neighbor $B \in \Gamma(A)$ with $\mathrm{ID}(B) < \mathrm{ID}(A)$ is in $\mathcal{F}'$ but not $\mathcal{F}$, which could only occur if some $C \in \Gamma^+(B) \subseteq \Gamma^{2+}(A)$ were resampled in round $j-1$. □

We can now proceed to bound the number of rounds of Algorithm 2 needed to find a satisfying assignment.

Theorem 2 (Asymmetric LLL) *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a finite set of events determined by these variables. If there exists an assignment of reals $x : \mathcal{A} \to (0, 1)$ such that*

$$\forall A \in \mathcal{A} : \Pr(A) \leq (1-\epsilon)x(A) \prod_{B \in \Gamma^2(A)} (1 - x(B)),$$

then the probability any bad event occurs after k resampling rounds of Algorithm 2 is at most $(1-\epsilon)^k \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}$.

Note the difference with Theorem 1 is that the product is over all $B \in \Gamma^2(A)$ not $B \in \Gamma(A)$.

Corollary 1 (Symmetric LLL) *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a finite set of events determined by these variables, such that for all $A \in \mathcal{A}$*

1. *$Pr(A) \leq p < 1$, and*
2. *A shares variables with at most d of the other events.*

If $epd^2 < 1$, then w.h.p. none of the bad events occur after $O(\log_{\frac{1}{epd^2}} n)$ rounds of Algorithm 2.

Proof Setting $x(A) = 1/d^2$ and $\epsilon = 1 - epd^2$ in Theorem 2, we have

$$\begin{aligned}
&(1-\epsilon)x(A) \prod_{B \in \Gamma^2(A)} (1 - x(B)) \\
&\geq \frac{1-\epsilon}{d^2} \cdot \left(1 - \frac{1}{d^2}\right)^{|\Gamma^2(A)|} \\
&\geq \frac{1-\epsilon}{d^2} \left(1 - \frac{1}{d^2}\right)^{(d^2-1)} \\
&\geq \frac{1-\epsilon}{ed^2} \geq p \geq \Pr(A).
\end{aligned}$$

Therefore, the probability a bad event occurs after k rounds of resampling is at most $(1-\epsilon)^k \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)} = (1-\epsilon)^k n/(d^2-1)$, which is $1/\operatorname{poly}(n)$ if $k = O(\log_{\frac{1}{1-\epsilon}} n) = O(\log_{\frac{1}{epd^2}} n)$. □

Following Moser and Tardos [33] we analyze the following Galton-Watson process for generating an r-witness tree T. Fix an event $A \in \mathcal{A}$. Begin by creating a root for T labelled A. To shorten the notation, we let $[v] := \sigma_T(v)$. In each subsequent step, consider each vertex v created in the previous step. For each $B \in \Gamma^{r+}([v])$, independently, attach a child labelled B with probability $x(B)$ or skip it with probability $1-x(B)$. Continue the process until no new vertices are born. We prove a lemma analogous to one in [33].

Lemma 4 *Let τ be a fixed proper r-witness tree with its root vertex labelled A. The probability p_τ that the Galton-Watson process yields exactly the tree τ is*

$$p_\tau = \frac{1-x(A)}{x(A)} \prod_{v \in V(\tau)} x'([v])$$

where $x'(B) = x(B) \cdot \Pi_{C \in \Gamma^r(B)}(1 - x(C))$.

Proof Let $W_v \subseteq \Gamma^{r+}([v])$ denote the set of inclusive r-neighbors of $[v]$ that do not occur as a label of some child node of v. Then,

$$\begin{aligned}
p_\tau &= \frac{1}{x(A)} \cdot \prod_{v \in V(\tau)} \left(x([v]) \cdot \prod_{u \in W_v} (1 - x([u]) \right) \\
&= \frac{1-x(A)}{x(A)} \cdot \prod_{v \in V(\tau)} \left(\frac{x([v])}{1-x([v])} \cdot \prod_{u \in \Gamma^{r+}([v])} (1 - x([u])) \right) \\
&= \frac{1-x(A)}{x(A)} \cdot \prod_{v \in V(\tau)} \left(x([v]) \cdot \prod_{u \in \Gamma^{r}([v])} (1 - x([u])) \right) \\
&= \frac{1-x(A)}{x(A)} \cdot \prod_{v \in V(\tau)} x'([v])
\end{aligned}$$

□

Lemma 5 *If for all $A \in \mathcal{A}$, we have $\Pr(A) \leq (1-\epsilon)x(A) \cdot \prod_{B \in \Gamma^r(A)}(1-x(B))$, then the probability that any r-witness tree of size at least k occurs is at most $(1-\epsilon)^k \cdot \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}$.*

Proof Let $\mathcal{T}_A^r(k)$ denote the infinite set of r-witness trees having root labelled A and containing at least k vertices. By Lemma 2 and the union bound, the probability there exists a violated event after k resampling rounds is at most

$$\begin{aligned}
&\sum_{A \in \mathcal{A}} \sum_{\tau \in \mathcal{T}_A^r(k)} \Pr(\tau \text{ } r\text{-occurs in } C) \\
&\leq \sum_{A \in \mathcal{A}} \sum_{\tau \in \mathcal{T}_A^r(k)} \prod_{v \in V(\tau)} \Pr([v]) && \text{by Lemma 2} \\
&\leq \sum_{A \in \mathcal{A}} \sum_{\tau \in \mathcal{T}_A^r(k)} \prod_{v \in V(\tau)} (1-\epsilon)x'([v]) \\
&\leq (1-\epsilon)^k \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)} \sum_{\tau \in \mathcal{T}_A^r(k)} p_\tau && \text{by Lemma 4} \\
&\leq (1-\epsilon)^k \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}
\end{aligned}$$

The last inequality follows since the Galton-Watson process grows exactly one tree. □

Let C be the record of Algorithm 2 and S_j be the segment of the record corresponding to resamplings in round j. The following lemma relates the number of resampling rounds with the occurence of 2-witness trees.

Lemma 6 *If there is still a violated event after k resampling rounds in Algorithm 2 then some 2-witness tree of size at least k occurs in C.*

Proof Let A_k be any event in S_k and t be its position in the record C. By Lemma 3 there exist events $A_{k-1}, \ldots, A_1$ in

$S_{k-1}, \cdots, S_1$ such that for all $j < k$, $A_j \in \Gamma^{2+}(A_{j+1})$. This implies that $A_{k-1}, \ldots, A_1$ are mapped to distinct nodes in the 2-witness tree $\tau_C(t)$, whose root is labeled A_k. □

Therefore, by Lemma 6, if there is a violated event after k resampling rounds, then a 2-witness tree of size at least k occurs. However, by Lemma 5, it happens with probability at most $(1-\epsilon)^k \cdot \sum_{A\in\mathcal{A}} \frac{x(A)}{1-x(A)}$. Thus, Theorem 2 holds. Note that if $x(A)$ is bounded away from 1, then after $O(\log_{\frac{1}{1-\epsilon}} n)$ rounds, w.h.p. no bad event occurs.

3.2 Resampling by weak MIS

In this section we analyze the efficiency of Moser and Tardos's Algorithm 1 when a new *weak MIS* procedure (Algorithm 3) is used in lieu of an actual MIS. The Weak-MIS procedure produces, in $O(\log^2 d)$ time, an independent set S such that the probability that a node is *not* in $\Gamma^+(S) = S \cup \Gamma(S)$ is $1/\operatorname{poly}(d)$. The procedure consists of $O(\log d)$ iterations where the probability that a vertex avoids $\Gamma^+(S)$ is constant per iteration. Each iteration consists of $\log d$ phases where, roughly speaking, the goal of phase i is to eliminate vertices with degree at least $d/2^i$ with constant probability. Each phase is essentially one step of Luby's MIS algorithm, though applied only to a judiciously chosen subset of the vertices. See Algorithm 3.

Our main results are as follows.

Theorem 3 (Asymmetric LLL) *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a finite set of events determined by these variables. If there exists an assignment of reals $x : \mathcal{A} \to (0, 1)$ such that*

$$\forall A \in \mathcal{A} : \Pr(A) \le (1-\epsilon)x(A) \prod_{B\in\Gamma(A)} (1-x(B)),$$

then the probability any bad event occurs after k resampling rounds using the Weak-MIS algorithm is at most $n(\frac{1}{d+1})^k + (1-\epsilon)^{k/2} \sum_{A\in\mathcal{A}} \frac{x(A)}{1-x(A)}$.

Corollary 2 (Symmetric LLL) *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a finite set of events determined by these variables, such that for $\forall A \in \mathcal{A}$,*

1. *$Pr(A) \le p < 1$, and*
2. *A shares variables with at most d of the other events.*

If $ep(d+1) < 1$, then w.h.p. none of the bad events occur after $O(\max(\log_{d+1} n, \log_{\frac{1}{ep(d+1)}} n))$ Weak-MIS resampling rounds.

Corollary 2 follows directly by plugging in $x(A) = 1/(d+1)$ for all $A \in \mathcal{A}$ and $k = O\left(\max\left(\log_{d+1} n, \log_{\frac{1}{ep(d+1)}} n\right)\right)$.

Notice that if $\frac{1}{ep(d+1)} > d+1$, we can apply the faster simple distributed algorithm, so the running time in Corollary 2 will be dominated by $O(\log_{\frac{1}{ep(d+1)}} n \cdot \log^2 d)$.

$S \leftarrow \emptyset$
for iteration $1 \ldots, t = 4e^2 \ln(2e(d+1)^4)$ **do**
  $G' \leftarrow G_{\mathcal{F}} \setminus \Gamma^+(S)$
  for phase $i = 1 \ldots \lceil \log d \rceil$ **do**
    $V_i \leftarrow \{v \in G' \mid \deg_{G'}(v) \ge d/2^i\}$.
    For each vertex $v \in G'$, set
    $b(v) \leftarrow \begin{cases} 1 & \text{with probability } p_i = 1/(\frac{d}{2^{i-1}} + 1) \\ 0 & \text{otherwise} \end{cases}$
    For each vertex $v \in G'$, if $b(v) = 1$ and $b(w) = 0$ for all $w \in \Gamma_{G'}(v)$, set $S \leftarrow S \cup \{v\}$.
    $G' \leftarrow G' \setminus (\Gamma^+(S) \cup V_i)$ (i.e., remove both $\Gamma^+(S)$ and V_i from G'.)
  end for
  Let S' be the (isolated) vertices that remain in G'.
  Set $S \leftarrow S \cup S'$
end for
return S

Algorithm 3: Weak-MIS

Consider the first iteration of the Weak-MIS algorithm. For each phase i, G' is the subgraph of $G_{\mathcal{F}}$ containing vertices with degree at most $d/2^i$ and not adjacent to the independent set S. Let $V_i = \{v \in G' \mid \deg_{G'}(v) \ge d/2^i\}$. Note that every vertex in $G_{\mathcal{F}}$ must end up isolated in S' or one of the V_i's. Let (u, v) be an edge in G'. Following Peleg's analysis [35], define $\mathcal{E}(u, v)$ to be the event that at phase i, $b(u) = 0$ and $b(v) = 1$ and for all other neighbors x of u and v, $b(x) = 0$. Define $\mathcal{E}(u) = \bigcup_{v\in\Gamma_{G'}(u)} \mathcal{E}(u, v)$ to be the event that exactly one neighbor joins S in this phase. Since these events are disjoint, we have $\Pr(\mathcal{E}(u)) = \sum_{v\in\Gamma_{G'}(u)} \Pr(\mathcal{E}(u, v))$.

Lemma 7 *If $v \in V_i$, then $\Pr(\mathcal{E}(u)) \ge \frac{1}{4e^2}$.*

Proof $\Pr(\mathcal{E}(u, v)) \ge p_i(1-p_i)^{\deg_{G'}(u)+\deg_{G'}(v)} \ge p_i(1-p_i)^{2d/2^{i-1}} \ge p_i e^{-2}$. Since $\deg_{G'}(u) \ge d/2^i$, $\Pr(\mathcal{E}(u)) \ge \frac{d}{2^i} p_i e^{-2} \ge \frac{1}{4e^2}$ □

Therefore, if $v \in G_{\mathcal{F}} \setminus \Gamma^+(S)$ at the beginning of iteration l, the probability that $v \in \Gamma^+(S)$ at the end of iteration l is at least $1/(4e^2)$. We say a vertex in $G_{\mathcal{F}}$ *fails* if, after all $t = 4e^2 \ln(2e(d+1)^4)$ iterations, it is still not in $\Gamma^+(S)$.

Lemma 8 *Let S be an independent set selected by Weak-MIS. If $v \in \mathcal{F}$ then $\Pr(\Gamma^+(v) \cap S = \emptyset) \le \frac{1}{2e(d+1)^4}$.*

Proof By Lemma 7, the probability that v survives iteration ℓ conditioned on it surviving iterations 1 through $\ell - 1$ is at most $1 - 1/(4e^2)$. Over $t = 4e^2 \ln(2e(d+1)^4)$ iterations the probability of failure is at most $(1 - 1/(4e^2))^t \le e^{-\ln(2e(d+1)^4)} = \frac{1}{2e(d+1)^4}$. □

The next step is to relate the number of rounds of Weak-MIS resampling with the size of witness trees.

Lemma 9 *Suppose a bad event is violated after k rounds of Weak-MIS resampling and the maximum depth of the witness trees is t, then there exists a sequence of not necessarily distinct vertices $v_1, \ldots, v_k$ such that the following hold:*

(1) $v_i \in G_i$, where G_i is the violated subgraph $G_{\mathcal{F}}$ at the beginning of round i.
(2) $v_{i+1} \in \Gamma^+(v_i)$ for $1 \le i \le k-1$.
(3) For at least $k - t$ indices $1 < l \le k$, v_l failed in the call to Weak-MIS in round $l - 1$.

Proof For $1 \le i \le k$, let S_i be the segment of the record C corresponding to events resampled at round i. Suppose that an event A is violated after k resampling rounds. Build a witness tree τ with root labeled A, adding nodes in the usual fashion, by scanning the record C in time-reversed order. For each j, in decreasing order, attach a node labelled $C(j)$ to the deepest node in τ whose label is in $\Gamma^+(C(j))$, if such a node in τ exists. Let $v_{k+1} = A$. We will build $v_k, v_{k-1}, \ldots, v_1$ in backward manner. For $k \ge i \ge 1$, we claim there is an event $v_i \in \Gamma^+(v_{i+1})$ such that either $v_i \in S_i$ or $v_i \in G_i$ and v_i failed at round i. If $v_{i+1} \notin G_i$ is not violated at the beginning of round i, then it must be the case that there exists an event $v_i \in \Gamma^+(v_{i+1})$ resampled at round i to cause $v_{i+1} \in G_{i+1}$. On the other hand, if $v_{i+1} \in G_i$ *is* violated at the beginning of round i, then either there exists $v_i \in \Gamma^+(v_{i+1})$ resampled at round i or v_{i+1} failed at round i. In the latter case, we let $v_i = v_{i+1}$. Notice that τ (excluding its artificial root labeled A) is a witness that occured and thus has depth at most t. Since in each of the k rounds, either the depth of our witness tree grows or a vertex fails, at least $k - t$ vertices must have failed in their respective rounds. □

Notice that the total possible number of sequences satisfying (2) in Lemma 9 is at most $n(d+1)^{k-1}$. Given a sequence of vertices $P = (v_1, \ldots, v_k)$ satisfying (2), define $X_P^{(i)}$ to be 1 if $v_i \in G_i$ and v_i failed, 0 otherwise. Let $X_P = \sum_{i=1}^{k} X_P^{(i)}$. If a sequence satisfying (1–3) occured, then there exists P such that $X_P \ge k - t$. Since $X_P^{(1)}, \ldots, X_P^{(i-1)}$ are determined by $S_1, \ldots, S_{i-1}$ and $G_1, \ldots, G_{i-1}$, $\mathrm{E}(X_P^{(i)} \mid X_P^{(1)}, \ldots, X_P^{(i-1)}) = \mathrm{E}(X_P^{(i)} \mid S_1, \ldots, S_{i-1}, G_1, \ldots, G_{i-1}) \le q \stackrel{\text{def}}{=} \frac{1}{2e(d+1)^4}$ by Lemma 8. Fixing $t = k/2$, we have $k - t = k/2 = kq \cdot e(d+1)^4 \le \mathrm{E}[X_P] \cdot e(d+1)^4$. By Lemma 19 (Conditional Chernoff Bound):

$$\Pr(X_P \ge k/2) \le \left(\frac{e^{e(d+1)^4 - 1}}{\left(e(d+1)^4\right)^{e(d+1)^4}} \right)^{\frac{k}{2e(d+1)^4}} \le \left(\frac{1}{(d+1)^2} \right)^k.$$

By the union bound over all possible P satisfying (2), the probability that any such sequence in Lemma 9 occurs is at most

$$n\,(d+1)^{k-1} \cdot \left(\frac{1}{(d+1)^2} \right)^k \le n \cdot \left(\frac{1}{d+1} \right)^k.$$

Moser and Tardos showed that the probability that any witness tree of size at least t occurs is at most $(1-\epsilon)^t \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}$. Thus, either a witness tree of depth at least $t = k/2$ occurs or there exists a sequence of vertices (as in Lemma 9) such that $t - k = k/2$ of them failed. The probability either of these occurs is at most $n \cdot \left(\frac{1}{d+1}\right)^k + (1-\epsilon)^{k/2} \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}$ by the union bound.

3.3 A sublogarithmic algorithm

We have seen a faster algorithm for LLL when the general condition $ep(d+1) < 1$ is replaced by a stronger condition $p \cdot f(d) < 1$, where $f(d)$ is a faster growing function than $e(d+1)$. The question of how fast we can do for a stronger condition arises. Does there exist a sublogarithmic algorithm for faster growing $f(d)$, independently of n? We answer this affirmatively for an exponential function of d.

Inspired by [3], our approach is a two-stage approach. In the first stage, we run Algorithm 2 for $k(n)$ rounds. Then we identify the *dangerous* events, who are likely to become violated if some subset of its neighborhood is resampled. We will show there is a feasible solution by re-assigning the variables belonging to dangerous events. Moreover, we show the components induced by the dangerous events are likely to have *weak diameter* at most $k(n)$. The weak diameter of a component is the maximum distance w.r.t. the original graph of any pair in the component. In the second stage, each component of dangerous events computes the answer independent of others in time proportional to its weak diameter.

Theorem 4 (Asymmetric LLL) *Let* $\Pr(A) \le P_2(A) \le 1$ *and* $P_1(A) = 2^d \cdot \frac{\Pr(A)}{P_2(A)}$, *where d is the maximum degree of the dependency graph. If there exists an assignments of reals $x_1, x_2 : \mathcal{A} \to (0, 0.99]$ such that for all $A \in \mathcal{A}$*

1. $P_1(A) \le (1-\epsilon) x_1(A) \prod_{B \in \Gamma^3(A)} (1 - x_1(B))$
2. $P_2(A) \le x_2(A) \prod_{B \in \Gamma(A)} (1 - x_2(B))$

then the LLL problem can be solved in $O\left(\log_{1/(1-\epsilon)} n / \log \log_{1/(1-\epsilon)} n\right)$ *rounds.*

Proof Sketch of Theorem 4. Given an assignment of each variables, we will classify the vertices into *safe* vertices and *dangerous* vertices. An event A is safe if the probability A becomes violated when any subset of its neighbors resample is at most $P_2(A)$. In contrast, the dangerous vertices are those where there exists a subset of neighbors whose resampling will cause it to be violated with probability greater than $P_2(A)$.

Using conditional probability, we can bound the probability that a vertex becomes dangerous after a random sampling of $\mathrm{vbl}(A)$ by $P_1(A) = 2^d \Pr(A)/P_2(A)$ (Lemma 10). Using Cond. 1 in Theorem 4, we show in Lemma 11 that after we resample dangerous vertices using the simple distributed algorithm for k rounds, if there exists a dangerous component whose weak diameter is at least k, then a 3-witness tree of size $\Omega(k \log k)$ would occur. When $k = \Theta(\log n / \log \log n)$, a 3-witness tree of size $O(\log n)$ would occur, which happens with probability at most $1/\mathrm{poly}(n)$. Therefore, with high probability, after $O(\log n / \log \log n)$ rounds of resampling, the weak diameters of the dangerous components are bounded by $O(\log n / \log \log n)$. Finally, a feasible assignment for a dangerous component can be found in $O(\log n / \log \log n)$ rounds locally, independent of other dangerous components, which can be argued using Cond. 2 in Theorem 4 and the definition of dangerous vertices.

Proof (Proof of Theorem 4) Fix $\emptyset \subseteq \mathcal{D} \subseteq \Gamma(A)$, let $T_{\mathcal{D}}$ denote the set of assignments b for $\mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D})$ such that $b \in T_{\mathcal{D}}$ iff when the variables in $\mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D})$ are fixed to be equal to b, the probability A becomes violated after sampling variables in $\mathrm{vbl}(\mathcal{D})$ exceeds $P_2(A)$, that is,

$$T_{\mathcal{D}} = \{b \mid \Pr(A \mid \mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D}) = b) > P_2(A)\}$$

Given an assignment of the variables of A, we call A "dangerous" if there exists $\emptyset \subseteq \mathcal{D} \subseteq \Gamma(A)$ such that $\mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D}) \in T_{\mathcal{D}}$. Otherwise, A is "safe". Notice that if A is violated then A is also dangerous, if we choose $\mathcal{D} = \emptyset$. □

Lemma 10 *The probability that A becomes dangerous after (re)sampling* $\mathrm{vbl}(A)$ *is at most* $P_1(A)$.

Proof By the union bound over each subset of neighbors, the probability that A becomes dangerous after sampling or resampling variables in $\mathrm{vbl}(A)$ is at most

$$\begin{aligned}
&\sum_{\emptyset \subseteq \mathcal{D} \subseteq \Gamma(A)} \Pr(\mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D}) \in T_{\mathcal{D}}) \\
&= \sum_{\emptyset \subseteq \mathcal{D} \subseteq \Gamma(A)} \sum_{b \in T_{\mathcal{D}}} \Pr(\mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D}) = b) \\
&= \sum_{\emptyset \subseteq \mathcal{D} \subseteq \Gamma(A)} \sum_{b \in T_{\mathcal{D}}} \frac{\Pr(A \cap (\mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D}) = b))}{\Pr(A \mid \mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D}) = b)} \\
&\le \sum_{\emptyset \subseteq \mathcal{D} \subseteq \Gamma(A)} \sum_{b \in T_{\mathcal{D}}} \frac{\Pr(A \cap (\mathrm{vbl}(A) \setminus \mathrm{vbl}(\mathcal{D}) = b))}{P_2(A)} \\
&\le \sum_{\emptyset \subseteq \mathcal{D} \subseteq \Gamma(A)} \frac{\Pr(A)}{P_2(A)} \\
&\le 2^d \cdot \frac{\Pr(A)}{P_2(A)} = P_1(A).
\end{aligned}$$

□

For each A, we define a new event A' to be that A becomes violated after resampling the variables of the dangerous events. Also, we let $\mathcal{A}'$ to be the set of all new events. If A is safe, then $\Pr(A') \le P_2(A)$ by definition of safe. If A is dangerous, then $\Pr(A') = \Pr(A) \le P_2(A)$. By the second condition in Theorem 4, there exists $x' : \mathcal{A}' \to (0, 0.99]$ such that $\Pr(A') \le x'(A') \prod_{B' \in \Gamma(A')}(1 - x'(B'))$ for all $A' \in \mathcal{A}'$. Therefore, by the standard asymmetric LLL, with non-zero probability, no new events $A' \in \mathcal{A}'$ occur. This implies there exists a feasible solution by reassigning only the variables of the dangerous events.

Let $E' \subseteq E$ be the edges having at least one endpoint that is dangerous. Let G' be the graph induced by E'. Each component of G' can compute the feasible solution independent of other components. (It is tempting to consider the components induced by only the dangerous vertices. However, when such components C_1 and C_2 are both adjacent to a safe vertex u, we have to consider C_1 and C_2 simultaneously to find an assignment that does not cause u to occur.)

Next we will show that the weak diameter of each component in G' is bounded. Note that if the weak diameter of each component in G' is at most D, then each component can find the feasible solution in $O(D)$ time. Each vertex will first learn the topology up to distance D, which is possible in the LOCAL model. Then the leader in each component (say the vertex with the smallest ID) computes the feasible solution locally and then broadcasts the solution back to other vertices in the component.

Lemma 11 *Suppose that the conditions in Theorem 4 hold, and there exists a component of weak diameter at least k after running k rounds of the simple distributed algorithm, then a 3-witness tree of size* $\Omega(k \log k)$ *occurs.*

Proof Suppose that there exists u, v in the same component in G' and $\mathrm{dist}_G(u, v) = D \ge k$. Since u, v are connected in G', there exists a shortest u-v path P_{uv} of length at least

D in G'. Notice that there are no consecutive safe vertices in P_{uv} by the definition of G'. Recall that S_i is the set of events resampled in round i. Let L_{k+1} be the set of dangerous vertices in P_{uv}. Ideally, one would build $|L_{k+1}|$ 2-witness trees of depth k, each rooted at each vertex in L_{k+1}, and then glue them together into a 3-witness tree of size $k \cdot |L_{k+1}|$. However, these 2-witness trees may overlap, so the final 3-witness tree may be much smaller. In the following, we will lower bound the size of the union of the 2-witness tree level by level and show that the size of the final 3-witness tree can be lower bounded.

For each dangerous vertex x in P_{uv} (i.e. $x \in L_{k+1}$), define $L_{k+1}(x) = \{x\}$. For $1 \le i \le k$, define $L_i(x)$ inductively to be the set of events sampled during round i that are within distance 2 to any events in $L_{i+1}(x)$. Define $L_i = \bigcup_{x \in P_{uv}} L_i(x)$. For each $1 \le i \le k$, we will show the size of L_i is at least $\frac{D-2}{4(k-i+1)+2}$.

Notice that $L_i(x)$ must be non-empty, because by Lemma 3, for each $k+1 \ge j > i$ and each vertex w_j in L_j, there exists a vertex $w_{j-1} \in S_{j-1}$ such that $w_{j-1} \in \Gamma^{2+}(w_j)$. Also, for all $w \in L_i(x)$, $\mathrm{dist}_G(x, w) \le 2(k-i+1)$, since by definition of $L_i(x)$, there exists a sequence of vertices $(x = v_{k+1}, v_k, \ldots, v_i = w)$ such that $v'_i \in L_i(x)$ for $k+1 \ge i' \ge i$ and $\mathrm{dist}_G(v_{i'+1}, v_{i'}) \le 2$ for $k+1 > i' \ge i$.

Let $P_{uv} = \{x_0, x_1, \ldots x_{|P_{uv}|}\}$. Let $j = 0$ if x_0 is dangerous; otherwise x_1 must be dangerous and we let $j = 1$. Repeat the following procedure (see Fig. 1): Select any $w \in L_i(x_j)$. Note that x_j must be dangerous and $L_i(x_j)$ is well-defined. Let $x_{j'}$ be the rightmost vertex in P_{uv} such that $w \in L_i(x'_j)$ (it can be the case that $j' = j$). If $x_{j'+1}$ is dangerous, set $j \leftarrow j'+1$; otherwise $x_{j'+2}$ must be a dangerous vertex, then we set $j \leftarrow j'+2$. Repeat until $j > |P_{uv}|$ (Fig. 2).

$|L_i|$ must be lower bounded by the total number of iterations l in the procedure above. We will show that we cannot move too far in each iteration, otherwise we would have a path shorter than $\mathrm{dist}_G(u, v)$ connecting u and v. Let Δ_t be the difference of j at the beginning of iteration t and

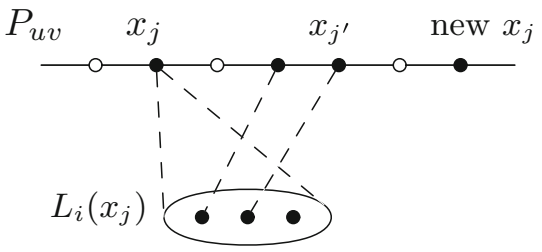

Fig. 1 An illustration of an iteration in the procedure for lower bounding L_i. The *dashed lines* are paths with length at most $2(k-i+1)$. In this iteration, the difference, Δ, between the new position and the old position of j is 5. Therefore, if $2 \cdot 2(k-i+1)+2 < 5$, then the detour from x_j to x'_j via $L_i(x_j)$ would be shorter the distance between x_j and x'_j on P_{uv}

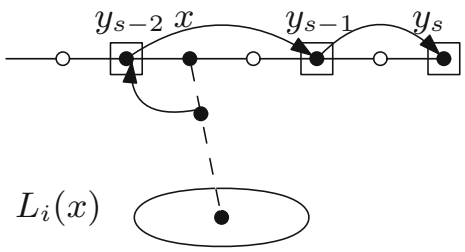

Fig. 2 An illustration showing that each resampled events in L_i is in the 3-witness tree rooted at y_s. The *vertices inside the boxes* are the independent set I. The *dashed line* is a sequence of vertices, where adjacent vertices have distance at most 2. The *arrow links* denote two vertices are within distance 3

at the end of iteration t. The procedure terminates only if $\sum_{t=1}^{l} \Delta_t \ge |P_{uv}| - 2$ (the minus 2 came from the fact that the first and the last vertex in P_{uv} can be safe). Consider iteration t, if $\Delta_t > 4(k-i+1)+2$, it must reduce the distance between u and v by at least $\Delta_t - 4(k-i+1) - 2$. However, the total distance we can reduce is at most $|P_{uv}| - D$, for otherwise we would have a path connecting u and v with length less D, contradicting with $\mathrm{dist}_G(u, v) = D$. Therefore,

$$\begin{aligned} |P_{uv}| - D &\ge \sum_{t=1}^{l} (\Delta_t - 4(k-i+1) - 2) \\ &\ge \left(\sum_{t=1}^{l} \Delta_t\right) - (4(k-i+1) - 2)\, l \\ &\ge |P_{uv}| - 2 - (4(k-i+1) - 2)\, l \end{aligned}$$

which implies

$$l \ge \frac{D-2}{4(k-i+1)-2} \ge \frac{k-2}{4(k-i+1)-2}.$$

Next, we will show that we can glue all the resampled events in $L_1, \ldots, L_k$ into a single 3-witness tree. We select an independent set $I = \{y_1, \ldots, y_s\} \subseteq L_{k+1}$ by starting from the leftmost vertex in L_{k+1} and repeatedly selecting the first non-adjacent vertex in L_{k+1}. Therefore, y_{j+1} is in distance at most 3 from y_j for $1 \le j < s$. Also, each $x_j \in L_{k+1}$ is adjacent to at least one vertex in I. Since I is an independent set, we can append $y_1, \ldots, y_s$ to our record artificially. We claim that each node in L_i for $1 \le i \le k$ corresponds to a node in the 3-witness tree rooted at y_s. For every node w in L_i, there must exist $x \in L_{k+1}$ such that $w \in L_i(x)$. Since x is adjacent to some $y_j \in I$, it implies w is in the 3-witness tree rooted at y_j. Finally, since y_j is a node in the 3-witness tree rooted at y_s, w must also be a node in the 3-witness tree rooted at y_s. The 3-witness tree rooted at y_s must have size at least $\sum_{i=1}^{k} \frac{k-2}{4(k-i+1)-2} = \Omega(k \log k)$. □

By choosing $k = \Omega\left(\frac{\log_{1/(1-\epsilon)} n}{\log\log_{1/(1-\epsilon)} n}\right)$, if there exists a component in G' with diameter at least k, then there exists a 3-witness of size at least $\Omega(\log_{1/(1-\epsilon)} n)$ w.h.p. However, by Condition 1 in Theorem 4 and by Lemma 5, the probability that such a 3-witness tree occurs is at most $1/\operatorname{poly}(n)$. Therefore, we can conclude that after $O\left(\frac{\log_{1/(1-\epsilon)} n}{\log\log_{1/(1-\epsilon)} n}\right)$ rounds, the weak diameter of each component in G' is at most $O\left(\frac{\log_{1/(1-\epsilon)} n}{\log\log_{1/(1-\epsilon)} n}\right)$ w.h.p. and the solution can be found in time proportional to the weak diameter. This completes the proof of Theorem 4. □

Corollary 3 (Symmetric LLL) *Suppose that for all $A \in \mathcal{A}$, $\Pr(A) \le p$ and A shares variables with at most d other events in $\mathcal{A}$. Let $z = 4ep2^d d^4$. If $z < 1$, then a satisfying assignment can be found in $O(\log_{1/z} n / \log\log_{1/z} n)$ rounds.*

Proof (Proof of Collorary 3) For each $A \in \mathcal{A}$, let $P_2(A) = \frac{1}{4d} \ge p \ge \Pr(A)$ and so $P_1(A) = 2^d \cdot \frac{\Pr(A)}{P_2(A)} \le 4pd2^d$. Let $x_1(A) = 1/d^3, x_2(A) = 1/(2d)$ and $1-\epsilon = 4ep2^d d^4$. First, we check that condition 1 in Theorem 4 holds

$$\begin{aligned}
&(1-\epsilon)x_1(A) \prod_{B\in\Gamma^3(A)} (1-x_1(A)) \\
&= 4ep2^d d^4 \cdot \frac{1}{ed^3} \cdot \left(1-\frac{1}{d^3}\right)^{|\Gamma^3(A)|} \\
&\ge 4ep2^d d \left(1-\frac{1}{d^3}\right)^{d^3-1} \\
&\ge 4p2^d d = \Pr(A).
\end{aligned}$$

Condition 2 also holds similarly,

$$\begin{aligned}
x_2(A) \prod_{B\in\Gamma(A)} (1-x_2(A)) &\ge \frac{1}{2d} \cdot \left(1-\frac{1}{2d}\right)^d \\
&= \frac{1}{2d} \cdot \frac{1}{2} = P_2(A).
\end{aligned}$$

□

3.4 Lower bound

Linial [26] proved that in an n-vertex ring, any distributed $(\log^{(k)} n)$-coloring algorithm requires $\Omega(k)$ rounds of communication, even if randomization is used. In particular, $O(1)$-coloring a ring requires $\Omega(\log^* n)$ time. We prove that Linial's lower bound implies that even weak versions of the Lovász local lemma cannot be computed in constant time.

Theorem 5 *Let $\mathcal{P}$, $\mathcal{A}$, and $G_{\mathcal{A}}$ be defined as usual. Let d be the maximum degree of any vertex in $G_{\mathcal{A}}$, $p = \max_{A\in\mathcal{A}} \Pr(A)$ be the maximum probability of any bad event, and $f : \mathbb{N} \to \mathbb{N}$ be an arbitrarily quickly growing function, where $f(d) \ge e(d+1)$. If $p \cdot f(d) < 1$ then $\Pr(\bigcap_{A\in\mathcal{A}} \overline{A}) > 0$. However, $\Omega(\log^* |\mathcal{A}|)$ rounds of communication are required for the vertices of $G_{\mathcal{A}}$ to agree on a point in $\bigcap_{A\in\mathcal{A}} \overline{A}$.*

The purpose of the function f is to show that our lower bound is insensitive to significant weakening of the standard criterion "$ep(d+1) < 1$." We could just as easily substitute $e^{e^d} p < 1$ or any similar criterion, for example.

Proof Consider the following coloring procedure. Each vertex in an n-vertex ring selects a color from $\{1, \ldots, c\}$ uniformly at random. An edge is *bad* if it is monochromatic, an event that holds with probability $p = 1/c$. Let $\mathcal{A}$ be the dependency graph for these events having maximum degree $d = 2$ and choose c to be (the constant) $f(2) + 1$, for any quickly growing function f. It follows from the LLL that a good c-coloring exists since $p \cdot f(2) < 1$. However, by [26], the vertices of $G_{\mathcal{A}}$ require $\Omega(\log^* n - \log^* c) = \Omega(\log^* n)$ time to find a good c-coloring. □

It is also possible to obtain *conditional* lower bounds on distributed versions of the LLL. For example, the best known randomized $O(\Delta)$-coloring algorithm takes $\exp(O(\sqrt{\log\log n}))$ time [6], though better bounds are possible if $\Delta \gg \log n$ [40]. If LLL could be solved in less than $\exp(O(\sqrt{\log\log n}))$ time then we could improve on [6], as follows. Each vertex in G selects a color from a palette of size $c \ge 2e\Delta$ uniformly at random. As usual, an edge is bad if it is monochromatic. The dependency graph of these bad events corresponds to the line graph of G, which has maximum degree $d = 2\Delta - 2$. Since $e(1/c)(d+1) < 1$, a valid coloring can be found with one invocation of an LLL algorithm. Therefore, if the result of [6] turns out to be tight, then there is an $\exp(O(\sqrt{\log\log n}))$ time lower bound of for LLL.

4 Applications

The Lovász local lemma has applications in many coloring problems, such as list coloring, frugal coloring, total coloring, and coloring triangle-free graphs [29]. We give a few examples of constructing these colorings distributively. In these applications, the existential bounds are usually achieved by the so called "Rödl Nibble" method or the semi-random method. The method consists of one or more iterations. Each iteration is a random process and some local properties are maintained in the graph. The properties depend on the randomness within a constant radius. Each property is associated with a bad event, which is the event that the property fails to hold. The Lovász local lemma can then be used to show the probability none of the bad events hold is positive, though

it may be exponentially small in the size of the graph. This probability can then be amplified in a distributed fashion using a Moser–Tardos-type resampling algorithm. Notice that we will need to find an independent set (e.g., an MIS or Weak-MIS or set of events with locally minimal IDs) *in the dependency graph* induced by the violated local properties. Since we assumed the LOCAL model, the violated local properties can be identified in constant time and the algorithms for MIS/Weak-MIS can be simulated with a constant factor overhead, where each property is taken care by one of the processors nearby (within constant distance). The important point here is that the dependency graph and the underlying distributed network are sufficiently similar so that distributed algorithms on one topology can be simulated on the other with $O(1)$ slowdown. For a simple example, see the defective coloring problem in the following subsection, where the dependency graph is G^2 (i.e. nodes are adjacent in G^2 iff they are within distance 2 in G).

Most applications of the LLL demand $epd^2 < 1$ or even weaker bounds. In this case, the efficient simple distributed algorithm can be applied. (The local properties are often that some quantities do not deviate too much from their expectations. Thus, the the failure probability of each local property is often bounded via standard Chernoff-type concentration inequalities.)

4.1 Distributed defective coloring

We begin with a simple single-iteration application that uses the local lemma. Let $\phi : V \to \{1, 2, \ldots, k\}$ be a k-coloring. Define $\mathrm{def}_\phi(v)$ to be the number of neighbors $w \in N(v)$ such that $\phi(v) = \phi(w)$. The coloring ϕ is said to be f-defective if $\max_v \mathrm{def}_\phi(v) \leq f$. Barenboim and Elkin ([4], Open Problem 10.7) raised the problem of devising an efficient distributed algorithm for computing an f-defective $O(\Delta/f)$-coloring. Note that this problem is equivalent to partitioning the vertices into $O(\Delta/f)$ sets such that each set induces a subgraph with maximum degree f.

To warm up, we give a simple procedure for obtaining an f-defective $O(\Delta/f)$-coloring in $O(\log n/f)$ time w.h.p., for $f \geq 60 \ln \Delta$. Suppose each vertex colors itself with a color selected from $\{1, 2, \ldots, \lceil 2\Delta/f \rceil\}$ uniformly at random. For every $v \in N(u)$, let X_v be 1 if v is colored the same as u, 0 otherwise. Let $X = \sum_{v \in N(u)} X_v$ denote the number of neighbors colored the same as v. Let A_u denote the bad event that $X > f$ at u. Clearly, whether A_u occurs is locally checkable by u in one round. Moreover, the event A_u only depends on the the random choices of u's neighbors. If A_u occured and is selected for resampling, the colors chosen by u and its neighbors will be resampled. Since two events share variables only if they are within distance two, the dependency graph, $G_{\mathcal{A}}$, is G^2. Therefore, $G_{\mathcal{A}}$ has maximum degree $d = \Delta^2$. Now we will calculate the probability that A_u occurs. If we expose the choice of u first, then $\Pr(X_v = 1) \leq f/(2\Delta)$ and it is independent among other $v \in N(u)$. Letting $M = f/2$, we have $E[X] \leq f/2 = M$. By Lemma 18, $\Pr(X > f) \leq e^{-f/6}$. Let A_u denote the bad event that $X > f$ at u. Therefore, $epd^2 \leq e^{-(f/6-1-4\ln\Delta)} \leq e^{-(f/12)}$, since $f \geq 60 \ln \Delta$. By using the simple distributed algorithm, it takes $O(\log_{1/epd^2} n) = O(\log n/f)$ rounds to avoid the bad events w.h.p.

Next, we show that there is a constant $C > 0$ such that for any $f \geq C$, an f-defective $O(\Delta/f)$-coloring can be obtained in $O(\log n/f)$ rounds. For $f < C$, we can use the $(\Delta+1)$-coloring algorithms to obtain 0-defective (proper) $(\Delta+1)$-colorings that runs in $O(\log n)$ rounds. Let $\Delta_0 = \Delta$ and $\Delta_i = \log^3 \Delta_{i-1}$.

if $f < 60 \ln \Delta_{i-1}$ **then**
  Each node in G' chooses a color from $\lceil (1 + 6\Delta_i^{-1/3}) \cdot \frac{\Delta_{i-1}}{\Delta_i} \rceil$ colors uniformly at random.
  Let A_u denote the event that more than Δ_i neighbors of u are colored the same with u.
  Run Algorithm 2 until no bad events A_u occurs.
  Let G_j denote the graph induced by vertices with color j.
  For $j = 1 \ldots, \lceil (1 + 6\Delta_i^{-1/3}) \cdot \frac{\Delta_{i-1}}{\Delta_i} \rceil$, call defective-coloring(G_j, $i+1$) in parallel.
else
  Obtain an f-defective, $(2\Delta_{i-1}/f)$-coloring for G'.
end if

Algorithm 4: Defective-coloring(G', i)

An f-defective $O(\Delta/f)$-coloring in G can be obtained by calling defective-coloring(G, 1), which is described in Algorithm 4. The procedure defective-coloring(G', i) is a recursive procedure whose halting condition is when $f \geq 60 \log \Delta_{i-1}$. When the condition occurs, we will use the procedure described above to obtain an f-defective $(2\Delta_{i-1}/f)$-coloring in G'. Let l denote the total number of levels of the recursion. The final color of node v is a vector $(c_1, c_2, \ldots, c_l)$, where c_i denotes the color received by v at level i. Clearly, such a coloring obtained by the procedure is f-defective. The total number of colors used is:

$$\left(\prod_{1 \leq i < l} \left(1 + 6\Delta_i^{-1/3}\right) \cdot \frac{\Delta_{i-1}}{\Delta_i} \right) \cdot \frac{2\Delta_{l-1}}{f}$$

$$= 2(\Delta/f) \cdot \prod_{1 \leq i < l} \left(1 + \frac{6}{\log \underbrace{\log^3 \ldots \log^3 \Delta}_{i-1}} \right)$$

$$= O(\Delta/f).$$

Now we will analyze the number of rounds needed in each level i. Suppose that each vertex colors itself with a color

selected from $\{1, 2, \ldots, \lceil(1+6\Delta_i^{-1/3})\cdot\frac{\Delta_{i-1}}{\Delta_i}\rceil\}$ uniformly at random. For every $v \in N(u)$, let X_v be 1 if v is colored the same as u, 0 otherwise. Let $X = \sum_{v\in N_{G'}(u)} X_v$ denote the number of neighbors colored the same as v. Let A_u denote the bad event that $X > \Delta_i$ at u. The dependency graph $G_{\mathcal{A}}$ has maximum degree $d = \Delta_{i-1}^2$, because two events share variables only if they are within distance two. If we expose the choice of u first, then $\Pr(X_v = 1) \le \frac{\Delta_i}{\Delta_{i-1}} \cdot \frac{1}{1+6\Delta_i^{-1/3}}$ and it is independent among other $v \in N_{G'}(u)$. Since the maximum degree of G' is Δ_{i-1}, $\mathrm{E}[X] \le \Delta_i \cdot \frac{1}{1+6\Delta_i^{-1/3}}$. By Chernoff Bound (Lemma 18),

$$\begin{aligned}\Pr(A_u) &= \Pr(X > \Delta_i)\\ &\le \Pr\left(X > \left(1+6\Delta_i^{-1/3}\right)\cdot \mathrm{E}[X]\right)\\ &\le e^{-6^2\Delta_i^{-2/3}\cdot \mathrm{E}[X]/3}\\ &\le e^{-6\Delta_i^{1/3}} = e^{-6\ln\Delta_{i-1}}.\end{aligned}$$

Therefore, $epd^2 \le e^{-\ln\Delta_{i-1}}$ and so Algorithm 2 runs in $O(\log n/\log\Delta_{i-1})$ rounds. The total number of rounds over all levels is therefore

$$\begin{aligned}&O\left(\log n\cdot\left(\frac{1}{\log\Delta}+\frac{1}{\log\log^3\Delta}+\cdots+\frac{1}{\log\Delta_{l-1}}+\frac{1}{f}\right)\right)\\ &= O\left(\frac{\log n}{f}\right).\end{aligned}$$

4.2 Distributed frugal coloring

A *β-frugal coloring* of a graph G is a proper vertex-coloring of G such that no color appears more than β times in any neighborhood. Molloy and Reed [29] showed the following by using an asymmetric version of the local lemma:

Theorem 6 *For any constant integer $\beta \ge 1$, if G has maximum degree $\Delta \ge \beta^\beta$ then G has a β-frugal proper vertex coloring using at most $16\Delta^{1+\frac{1}{\beta}}$ colors.*

Here we outline their proof and show how to turn it into a distributed algorithm that finds such a coloring in $O(\log n\cdot\log^2\Delta)$ rounds. If $\beta = 1$, then simply consider the square graph of G, which is obtained by adding the edges between vertices whose distance is 2. A proper coloring in the square graph is a 1-frugal coloring in G. Since the square graph has maximum degree Δ^2, it can be (Δ^2+1)-colored by simulating distributed algorithms for $(\Delta+1)$-coloring.

For $\beta \ge 2$, let $k = 16\Delta^{1+\frac{1}{\beta}}$. Suppose that each vertex colors itself with one of the k colors uniformly at random. Consider two types of bad events. For each edge uv, the Type I event $A_{u,v}$ denotes that u and v are colored the same. For each subset $\{u_1, \ldots, u_{\beta+1}\}$ of the neighborhood of a vertex, Type II event $A_{u_1,\ldots,u_{\beta+1}}$ denotes that $u_1, \ldots, u_{\beta+1}$ are colored the same. If none of the events occur, then the random coloring is a β-frugal coloring. For each Type I event $A_{u,v}$, $\Pr(A_{u,v})$ is at most $1/k$. For each Type II event $A_{u_1,\ldots,u_{\beta+1}}$, $\Pr(A_{u_1,\ldots,u_{\beta+1}}) \le 1/k^\beta$. For each bad event A, let $x(A) = 2\Pr(A)$. Notice that $x(A) \le 1/2$, we have:

$$\begin{aligned}&x(A)\prod_{B\in\Gamma(A)}(1-x(B))\\ &\ge x(A)\prod_{B\in\Gamma(A)}\exp(-x(B)\cdot 2\ln 2)\\ &\quad\{(1-x)\ge e^{-x\cdot 2\ln 2} \text{ for } x\le 1/2\}\\ &= x(A)\cdot\exp\left(-2\ln 2\cdot\sum_{B\in\Gamma(A)}2\Pr(B)\right)\end{aligned}$$

Since A shares variables with at most $(\beta+1)\Delta$ Type I events and $(\beta+1)\Delta\binom{\Delta}{\beta}$ Type II events,

$$\begin{aligned}\sum_{B\in\Gamma(A)}\Pr(B) &\le (\beta+1)\Delta\cdot\frac{1}{k}+(\beta+1)\Delta\binom{\Delta}{\beta}\cdot\frac{1}{k^\beta}\\ &< \frac{(\beta+1)\Delta}{k}+\frac{(\beta+1)\Delta^{\beta+1}}{\beta!k^\beta}\\ &= \frac{\beta+1}{16\Delta^{\frac{1}{\beta}}}+\frac{\beta+1}{\beta!(16)^\beta}\\ &< 1/8\end{aligned}$$

for $\Delta \ge \beta^\beta$ and $\beta \ge 2$

Therefore,

$$\begin{aligned}x(A)\prod_{B\in\Gamma(A)}(1-x(B)) &\ge x(A)\exp\left(-\frac{\ln 2}{2}\right)\\ &= \sqrt{2}\cdot\Pr(A).\end{aligned}$$

By letting $1-\epsilon = 1/\sqrt{2}$ in Theorem 3, we need at most $O(\log_{\sqrt{2}} n)$ rounds of weak MIS resampling. In each resampling round, we have to identify the bad events first. Type I events $A_{u,v}$ can be identified by either u or v in constant number of rounds, where ties can be broken by letting the node with smaller ID check it. If $\{u_1, \ldots, u_{\beta+1}\}$ is in the neighborhood of u, then the Type II event $A_{u_1,\ldots,u_{\beta+1}}$ will be checked by u. If $\{u_1, \ldots, u_{\beta+1}\}$ is in the neighborhood of multiple nodes, we can break ties by letting the one having the smallest ID to check it. All Type II events in the neighborhood of u can be identified from the colors selected by the neighbors of u. Next we will find a weak MIS induced by the bad events in the dependency graph. Each node will simulate the weak MIS algorithm on the events it is responsible to check. Each round of the weak MIS algorithm in the dependency graph can be simulated with constant rounds. The maximum degree d of the dependency graph is $O((\beta+1)\Delta\binom{\Delta}{\beta})$. Therefore, we need

at most $O(\log n \cdot \log^2 d) = O(\log n \cdot \log^2 \Delta)$ rounds, since β is a constant and $(\beta+1)\Delta\binom{\Delta}{\beta} \leq (\beta+1)\Delta^{\beta+1} = \text{poly}(\Delta)$.

4.2.1 β-frugal, $(\Delta+1)$-coloring

The frugal $(\Delta+1)$-coloring problem for general graphs is studied by Hind, Molloy, and Reed [21], Pemmaraju and Srinivasan [36], and Molloy and Reed [30]. In particular, the last one gave an upper bound of $O(\log \Delta / \log\log \Delta)$ on the frugality of $(\Delta+1)$-coloring. This is optimal up to a constant factor, because it matches the lower bound of $\Omega(\log \Delta / \log\log \Delta)$ given by Hind et al. [30]. However, it is not obvious whether it can be implemented efficiently in a distributed fashion, because they used a structural decomposition computed by a sequential algorithm. Pemmaraju and Srinivasan [36] showed an existential upper bound of $O(\log^2 \Delta / \log\log \Delta)$. Furthermore, they gave a distributed algorithm that computes an $O\big(\log \Delta \cdot \frac{\log n}{\log\log n}\big)$-frugal, $(\Delta+1)$-coloring in $O(\log n)$ rounds. We show how to improve it to find a $O(\log^2 \Delta / \log\log \Delta)$-frugal, $(\Delta+1)$-coloring also in $O(\log n)$ rounds.

They proved the following theorem:

Theorem 7 *Let G be a graph with maximum vertex degree Δ. Suppose that associated with each vertex $v \in V$, there is a palette $P(v)$ of colors, where $|P(v)| \geq \deg(v)+1$. Furthermore, suppose $|P(v)| \geq \Delta/4$ for all vertices v in G. Then, for some subset $C \subseteq V$, there is a list coloring of the vertices in C such that:*

(a) $G[C]$ is properly colored.
(b) For every vertex $v \in V$ and for every color x, there are at most $9 \cdot \frac{\ln \Delta}{\ln\ln \Delta}$ neighbors of v colored x.
(c) For every vertex $v \in V$, the number of neighbors of v not in C is at most $\Delta(1-\frac{1}{e^5}) + 27\sqrt{\Delta \ln \Delta}$.
(d) For every vertex $v \in V$, the number of neighbors of v in C is at most $\frac{\Delta}{e^5} + 27\sqrt{\Delta \ln \Delta}$.

The theorem was obtained by applying the LLL to the following random process: Suppose that each vertex v has an unique ID. Every vertex picks a color uniformly at random from its palette. If v has picked a color that is not picked by any of its neighbor whose ID is smaller than v, then v will be colored with that color. Let q_v denote the probability that v becomes colored. Then, if v is colored, with probability $1 - 1/(e^5 q_v)$, v uncolors itself. This ensures that the probability that v becomes colored in the process is exactly $1/e^5$, provided that $q_v \geq 1/e^5$, which they have shown to be true.

They showed by iteratively applying the theorem for $O(\log \Delta)$ iterations, an $O(\log^2 \Delta / \log\log \Delta)$-frugal, $(\Delta+1)$-coloring can be obtained. Let G_i be the graph after round i obtained by deleting already colored vertices and Δ_i be the maximum degree of G_i. The palette $P(u)$ for each vertex u contains colors that have not been used by its neighbors. It is always true that $|P(v)| \geq \deg(v)+1$. Notice that to apply Theorem 7, we also need the condition $|P(v)| \geq \Delta/4$. The worst case behavior of Δ_i and p_i is captured by the recurrences:

$$\Delta_{i+1} = \Delta_i\left(1 - \frac{1}{e^5}\right) + 27\sqrt{\Delta_i \ln \Delta_i}$$
$$p_{i+1} = p_i - \frac{\Delta_i}{e^5} - 27\sqrt{\Delta_i \ln \Delta_i}. \quad (1)$$

They showed the above recurrence can be solved to obtain the following bounds on Δ_i and p_i:

Lemma 12 *Let $\alpha = (1-1/e^5)$. There is a constant C such that for all i for which $\Delta_i \geq C$, $\Delta_i \leq 2\Delta_0\alpha^i$ and $p_i \geq \frac{\Delta_0}{2}\alpha^i$.*

Therefore, $|P(v)| \geq \Delta/4$ always holds. The two assumptions of Theorem 7 are always satisfied and so it can be applied iteratively until $\Delta_i < C$, which takes at most $\log_{1/\alpha}\left(\frac{2\Delta_0}{C}\right) = O(\log \Delta)$ iterations. Since each iteration introduces at most $O(\log \Delta / \log\log \Delta)$ neighbors of the same color to each vertex, the frugality will be at most $O(\log^2 \Delta / \log\log \Delta)$. In the end, when $\Delta_i < C$, one can color the remaining graph in $O(\Delta_i + \log^* n)$ time using existing (Δ_i+1)-coloring algorithms [5]. This will only add $O(1)$ copies of each color to the neighborhood, yielding a $O(\log^2 \Delta / \log\log \Delta)$-frugal, $(\Delta+1)$-coloring. In order to make it suitable for our simple distributed algorithm and achieve the running time of $O(\log n)$, we will relax the criteria of (b),(c),(d) in Theorem 7:

(b') For every vertex $v \in V$ and for every color x, there are at most $18 \cdot \frac{\ln \Delta_0}{\ln\ln \Delta_0}$ neighbors of v colored x.
(c') For every vertex $v \in V$, the number of neighbors of v not in C is at most $\Delta(1-\frac{1}{e^5}) + 40\sqrt{\Delta}\ln \Delta$.
(d') For every vertex $v \in V$, the number of neighbors of v in C is at most $\frac{\Delta}{e^5} + 40\sqrt{\Delta}\ln \Delta$.

In (b'), Δ is replaced by Δ_0, which is the maximum degree of the initial graph. Also, the constant 9 is replaced by 18. In (c') and (d'), the constant 27 is replaced by 40 and $\sqrt{\ln \Delta}$ is replaced by $\ln \Delta$. It is not hard to see that Lemma 12 still holds and an $O(\log^2 \Delta / \log\log \Delta)$-frugal coloring is still obtainable. Originally, by Chernoff Bound and Azuma's Inequality, they showed

$$\Pr\left(\#\text{ neighbors of } v \text{ colored } x \text{ exceeds } 9 \cdot \frac{\ln \Delta}{\ln\ln \Delta}\right) < \frac{1}{\Delta^6} \quad (2)$$

and

$$\Pr\left(\left|P_v - \frac{\deg(v)}{e^5}\right| > 27\sqrt{\Delta \ln \Delta}\right) < \frac{2}{\Delta^{4.5}} \tag{3}$$

where P_v is the number of colored neighbors of v. Theorem 7 can be derived from (2) and (3). The relaxed version (b'), (c'), and (d') can be shown to fail with a lower probability.

$$\Pr\left(\#\text{ neighbors of } v \text{ colored } x \text{ exceeds } 18 \cdot \frac{\ln \Delta_0}{\ln \ln \Delta_0}\right) < \frac{1}{\Delta_0^{12}} \tag{4}$$

and

$$\Pr\left(\left|P_v - \frac{\deg(v)}{e^5}\right| > 40\sqrt{\Delta} \ln \Delta\right) < \frac{2}{\Delta^{9 \ln \Delta}} \tag{5}$$

The bad event A_v is when the neighbors of v colored x exceeds $18 \cdot \frac{\ln \Delta_0}{\ln \ln \Delta_0}$ for some color x or $|P_v - \frac{\deg(v)}{e^5}| > 40\sqrt{\Delta} \ln \Delta$ happens. By (4), (5), and the union bound, $\Pr(A_v) \le (\Delta+1)/\Delta_0^{12} + 2/\Delta^{9 \ln \Delta}$. In their random process, they showed A_v depends on variables up to distance two. Thus, the dependency graph $G_{\mathcal{A}}$ has maximum degree d less than Δ^4. Note that

$$\begin{aligned} epd^2 &= e\Delta^8((\Delta+1)/(2\Delta_0^{12}) + 2/\Delta^{9\ln\Delta}) \\ &\le 1/(2\Delta_0) + 1/(2\Delta^{\ln\Delta}) \\ &< 2 \cdot \max(1/(2\Delta_0), 1/(2\Delta^{\ln\Delta})) \\ &= \max(1/\Delta_0, 1/\Delta^{\ln\Delta}). \end{aligned}$$

The number of resampling rounds needed is at most $O\left(\log_{\frac{1}{epd^2}} n\right)$, which is at most $\frac{\ln n}{\min(\ln \Delta_0, \ln^2 \Delta)} \le \frac{\ln n}{\ln \Delta_0} + \frac{\ln n}{\ln^2 \Delta}$. Therefore, the total number of rounds needed is at most:

$$\begin{aligned} &\sum_{i=1}^{c \ln \Delta_0} \left(\frac{\ln n}{\ln \Delta_0} + \frac{\ln n}{\ln^2 \Delta_i}\right) \\ &\le \sum_{i=1}^{c \ln \Delta_0} \left(\frac{\ln n}{\ln \Delta_0} + \frac{\ln n}{\ln^2(2\Delta_0 \alpha^i)}\right) \\ &= c \ln \Delta_0 \cdot \frac{\ln n}{\ln \Delta_0} + \sum_{i=1}^{c \ln \Delta_0} \frac{\ln n}{(\ln \Delta_0 - i \ln \frac{1}{\alpha} + \ln 2)^2} \\ &\le c \ln n + \ln n \cdot O\left(\sum_{i=1}^{\infty} \frac{1}{i^2}\right) = O(\log n) \end{aligned}$$

where $c > 0$ is some constant, and $\alpha = (1 - 1/e^5)$.

4.3 Distributed triangle-free graphs coloring

Pettie and Su [37] gave a distributed algorithm for (Δ/k)-coloring triangle-free graphs:

Theorem 8 *Fix a constant $\epsilon > 0$. Let Δ be the maximum degree of a triangle-free graph G, assumed to be at least some Δ_ϵ depending on ϵ. Let $k \ge 1$ be a parameter such that $2\epsilon \le 1 - \frac{4k}{\ln \Delta}$. Then G can be (Δ/k)-colored, in time $O(k + \log^* \Delta)$ if $\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon} = \Omega(\ln n)$, and, for any Δ, in time on the order of*

$$e^{O(\sqrt{\ln \ln n})} \cdot (k + \log^* \Delta) \cdot \frac{\log n}{\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon}} = \log^{1+o(1)} n.$$

The algorithm consists of $O(k + \log^* \Delta)$ iterations. For each iteration i, a property $\mathcal{H}_i(u)$ is maintained at each vertex u. If $\mathcal{H}_{i-1}(u)$ is true for all u in G, then after round i, it is shown $\mathcal{H}_i(u)$ fails with probability at most $p = \exp\left(-\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon+\Omega(\epsilon)}\right)$, which is at most $\exp\left(-\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon}\right)/e\Delta^4$ if $\Delta \ge \Delta_\epsilon$, for some constant Δ_ϵ. Note that if $\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon} = \Omega(\log n)$, then by union bound, with high probability all $\mathcal{H}_i(u)$ holds. Otherwise, they revert to the distributed constructive Lovász Local Lemma. Let G_i be the subgraph of G induced by uncolored vertices. The event $\mathcal{H}_i(u)$ shares random variables up to distance two from u in G_{i-1}. The bad events $\mathcal{A}$ is made up with $A_u = \overline{E}_i(u)$ for $u \in G_{i-1}$. Therefore, the dependency graph $G_{\mathcal{A}}$ is $G_{i-1}^{\le 4}$, where (u, v) is connected if the $\text{dist}_{G_{i-1}}(u, v) \le 4$. The maximum degree d of $G_{\mathcal{A}}$ is less than Δ^4. By the Lovász Local Lemma, since $ep(d+1) < 1$, the probability all $\mathcal{H}_i(u)$ simultaneously hold is positive. To achieve this constructively, note that by Theorem 1, it requires $O(\log_{\frac{1}{1-\epsilon}} n)$ resampling rounds, where $1-\epsilon = ep(d+1) \le \exp\left(-\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon}\right)$. Each resampling round involves finding an MIS. They showed in the case $\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon} = O(\log n)$, Δ will be at most polylog(n), where faster MIS algorithms can be applied. Now we will use the simple distributed algorithm presented in the previous section to resample without finding an MIS in each resampling round. First, notice that with some larger constant Δ_ϵ, if $\Delta \ge \Delta_\epsilon$, the failure probability p is at most $\exp\left(-\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon}\right)/e\Delta^8$. Since $epd^2 \le \exp\left(-\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon}\right)$, by Corollary 1, w.h.p. none of the bad events happen after $O\left(\log_{\frac{1}{epd^2}} n\right) = O\left(\frac{\log n}{\Delta^{1-\frac{4k}{\ln\Delta}-\epsilon}}\right)$ resampling rounds of the simple distributed algorithm, where each resampling round takes constant time. As a result, the number of rounds is reduced to $O(\log n)$.

Theorem 9 *Fix a constant $\epsilon > 0$. Let Δ be the maximum degree of a triangle-free graph G, assumed to be at least some Δ_ϵ depending on ϵ. Let $k \ge 1$ be a parameter such*

that $2\epsilon \leq 1 - \frac{4k}{\ln \Delta}$. *Then* G *can be* (Δ/k)*-colored, in time* $O(k + \log^* \Delta)$ *if* $\Delta^{1-\frac{4k}{\ln \Delta}-\epsilon} = \Omega(\ln n)$, *and, for any* Δ, *in time on the order of*

$$(k + \log^* \Delta) \cdot \frac{\log n}{\Delta^{1-\frac{4k}{\ln \Delta}-\epsilon}} = O(\log n).$$

Similarly, the $(1 + o(1))\Delta/\log\Delta$-coloring algorithm for girth-5 graphs in [37] can be obtained in $O(\log n)$ rounds by replacing Moser and Tardos' algorithm with the simple distributed algorithm.

4.4 Distributed list coloring

Given a graph G, each vertex v is associated with a list (or a *palette*) of available colors $P(v)$. Let $\deg_c(v)$ denote the number of neighbors $w \in N(v)$ such that $c \in P(w)$. Suppose that $\deg_c(v)$ is upper bounded by D. The list coloring constant is the minimum K such that for any graph G and any palettes $P(u)$ for $u \in G$, if $|P(u)| \geq K \cdot D$ and $\deg_c(u) \leq D$ for every $u \in G$ and every $c \in P(u)$, then a proper coloring can be obtained by assigning each vertex a color from its list. Reed [38] first showed the list coloring constant is at most $2e$ by a single application of LLL. Haxell [20] showed 2 is sufficient. Later, Reed and Sudakov [39] used a multiple iterations Rödl Nibble method to show the list coloring constant is at most $1 + o(1)$, where $o(1)$ is a function of D. Reed's upper bound of $2e$ can be made distributed and constructive with a slightly larger factor, say $2e + \epsilon$ for any constant $\epsilon > 0$. The LLL condition they need is close to tight and so we will need to use the weak MIS algorithm. The additional slack needed is due to the ϵ-slack needed in distributed LLL ($ep(d+1) \leq 1-\epsilon$). The constructive algorithm can be easily transformed from their proof. Here we outline their proof: Suppose $|P(v)| \geq (2e + \epsilon)D$ for all v. Each vertex is assigned a color from its palette uniformly at random. They showed that with positive probability, a proper coloring is obtained. Let $e = uv \in E$, and $c \in P(u) \cap P(v)$. Define $A_{e,c}$ to be the bad event that both u and v are assigned c. Clearly, $p = \Pr(A_{e,c}) = 1/((2e + \epsilon)D)^2$. Also, there are at most $(2e + \epsilon)D^2$ events that depend on the color u picks and at most $(2e + \epsilon)D^2$ events that depend on the color v picks. The dependency graph has maximum degree $d = 2(2e + \epsilon)D^2 - 2$. Since $ep(d + 1) \leq 2e/(2e + \epsilon)$ is upper bounded by a constant less than 1, we can construct the coloring in $O(\log n \cdot \log^2 D)$ rounds by using the weak MIS algorithm.

In the following, we shall show that for any constants $\epsilon, \gamma > 0$, there exists $D_{\epsilon,\gamma} > 0$ such that for any $D \geq D_{\epsilon,\gamma}$, any $(1 + \epsilon)D$-list coloring instance can be colored in $O(\log^* D \cdot \max(1, \log n/D^{1-\gamma}))$ rounds. The algorithm consists of multiple iterations. Let $P_i(u)$ and $\deg_{i,c}(u)$ be the palette and the c-degree of u at end of iteration i. Also, at the end of iteration i, denote the neighbor of u by $N_i(u)$ and the c-neighbor by $N_{i,c}(u)$, which are the neighbors of u having c in their palette. Suppose that each vertex u has an unique ID, ID(u). Let $N^*_{i,c}(u)$ denote the set of c-neighbors at the end of iteration i having smaller ID than u. Let $\deg^*_{i,c}(u) = |N^*_{i,c}(u)|$.

1: $G_0 \leftarrow G$
2: $i \leftarrow 0$
3: **repeat**
4: $\quad i \leftarrow i + 1$
5: $\quad$ **for** each $u \in G_{i-1}$ **do**
6: $\quad\quad (S_i(u), K_i(u)) \leftarrow \text{Select}(u, \pi_i, \beta_i)$
7: $\quad\quad$ Set $P_i(u) \leftarrow K_i(u) \setminus S_i(N^*_{i-1}(u))$
8: $\quad\quad$ **if** $S_i(u) \cap P_i(u) \neq \emptyset$ **then** color u with any color in $S_i(u) \cap P_i(u)$ **end if**
9: $\quad$ **end for**
10: $\quad G_i \leftarrow G_{i-1} \setminus \{\text{colored vertices}\}$
11: **until**

Algorithm 5: List-Coloring $(G, \{\pi_i\}, \{\beta_i\})$

1: Include each $c \in P_{i-1}(u)$ in $S_i(u)$ independently with probability π_i.
2: For each c, calculate $r_c = \beta_i/(1-\pi_i)^{\deg^*_{i-1,c}(u)}$.
3: Include $c \in P_{i-1}(u)$ in $K_i(u)$ independently with probability r_c.
4: **return** $(S_i(u), K_i(u))$.

Algorithm 6: Select(u, π_i, β_i)

In each iteration i, each vertex will select a set of colors $S_i(u) \subseteq P_{i-1}(u)$ and $K_i(u) \subseteq P_{i-1}(u)$, which are obtained from Algorithm 6. If a color is in $K_i(u)$ and it is not in $S_i(v)$ for any $v \in N^*_{i-1}(u)$, then it remains in its new palette $P_i(u)$. Furthermore, if $S_i(u)$ contains a color that is in $P_i(u)$, then u colors itself with the color (in case there are multiple such colors, break ties arbitrarily).

Given π_i, the selecting probability for each vertex u to include a color in $S_i(u)$, the probability that $u \notin S_i(N^*_{i-1}(u))$ is $(1-\pi_i)^{\deg^*_{i-1,c}(u)}$. Define $\beta_i = (1-\pi_i)^{t'_{i-1}}$, where t'_{i-1} is an upper bound on $\deg_{i-1,c}(u)$ for each vertex u and each color c. Then $r_c = \beta_i/(1-\pi_i)^{\deg^*_{i-1,c}(u)}$ is always at most 1 and thus it is a valid probability. Therefore, the probability that a color $c \in P_{i-1}(u)$ remains in $P_i(u)$ is $(1-\pi_i)^{\deg^*_{i-1,c}(u)} \cdot r_c = \beta_i$. As a result, the palette size shrinks by at most a β_i factor in expectation.

Suppose that p'_i is the lower bound on the palette size at the end of iteration i. Then the probability that u remains uncolored is upper bounded by the probability that any of the colors in $P_i(u)$ was not selected to be in $S_i(u)$. The probability is *roughly* $(1-\pi_i)^{p'_i}$, which we will define it to be α_i. The slight inaccuracy comes from the fact that we are conditioning on the new palette size $|P_i(u)|$ is lower bounded by p'_i. However, we will show the effect of this conditioning only affects the probability by a small amount.

Let $p_0 = (1+\epsilon)\cdot D$ and $t_0 = D$ be the initial palette size and upper bound on c-degree. In the following, p_i and t_i are the ideal lower bound of the palette size and the ideal upper bound of the c-degree at the end of each iteration i. p'_i and t'_i are the approximation of p_i and t_i, incoporating the errors from concentration bounds. K is a constant in the selecting probability that depends on ϵ. T is the threshold on the c-degree before we switch to a different analysis, since the usual concentration bound does not apply when the quantity is small. $\delta = 1/\log D$ is the error control parameter which is set to be small enough such that $(1\pm\delta)^i$ is $1\pm o(1)$ for every iteration i.

$$\begin{aligned}
&\pi_i = 1/(Kt'_{i-1}+1) && \delta = 1/\log D\\
&\alpha_i = (1-\pi_i)^{p'_i} && \beta_i = (1-\pi_i)^{t'_{i-1}}\\
&p_i = \beta_i p_{i-1} && t_i = \max(\alpha_i t_{i-1}, T)\\
&p'_i = (1-\delta)^i p_i && t'_i = (1+\delta)^i t_i\\
&K = 2+2/\epsilon && T = D^{1-0.9\gamma}/2
\end{aligned}$$

Intuitively, we would like to have t_i shrink faster than p_i. To ensure this happens, we must have $\alpha_1 \le \beta_1$, which holds under our setting of π_i. As we will show, α_i shrinks much faster than β_i as i becomes larger. Note that β_i is at least a constant, as

$$\begin{aligned}
\beta_i &= (1-1/(Kt'_{i-1}+1))^{t'_{i-1}}\\
&= (1-1/(Kt'_{i-1}+1))^{(Kt'_{i-1})\cdot(1/K)}\\
&\ge (e^{-1})^{1/K} = e^{-1/K}\\
&\quad \text{since } (1-1/(x+1))^x \ge e^{-1}.
\end{aligned}$$

Lemma 13 *$t_r = T$ after at most $r = O(\log^* D)$ iterations.*

Proof We divide the iterations into two stages, where the first stage consists of iterations i for which $t_{i-1}/p_{i-1} \ge 1/(1.1e^{2/K}K)$. During the first stage, we show that the ratio t_i/p_i decreases by a factor of $\exp\left(-(1-o(1))\frac{\epsilon^2}{4(1+\epsilon)}\right)$ in every round.

$$\begin{aligned}
\frac{t_i}{p_i} &= \frac{\alpha_i}{\beta_i}\frac{t_{i-1}}{p_{i-1}}\\
&= (1-\pi_i)^{p'_i - t'_{i-1}}\cdot\frac{t_{i-1}}{p_{i-1}} && \text{defn. } \alpha_i, \beta_i\\
&\le \exp\left(-\pi_i\cdot(p'_i - t'_{i-1})\right)\cdot\frac{t_{i-1}}{p_{i-1}} && 1-x\le e^{-x}\\
&\le \exp\left(-(1-o(1))\cdot\frac{1}{K}\left(\frac{p_i}{t_{i-1}}-1\right)\right)\cdot\frac{t_{i-1}}{p_{i-1}} && \text{defn. } \pi_i,\ \frac{p'_i}{t'_{i-1}} = (1-o(1))\frac{p_i}{t_{i-1}}\\
&\le \exp\left(-(1-o(1))\cdot\frac{1}{K}\left(\frac{\beta_i p_{i-1}}{t_{i-1}}-1\right)\right)\cdot\frac{t_{i-1}}{p_{i-1}} && \text{defn. } p_i\\
&\le \exp\left(-(1-o(1))\cdot\frac{1}{K}\left(e^{-1/K}(1+\epsilon)-1\right)\right)\cdot\frac{t_{i-1}}{p_{i-1}} && p_{i-1}/t_{i-1}\ge(1+\epsilon)\\
&\le \exp\left(-(1-o(1))\cdot\frac{((1-1/K)(1+\epsilon)-1)}{K}\right)\cdot\frac{t_{i-1}}{p_{i-1}} && e^{-x}\ge 1-x\\
&= \exp\left(-(1-o(1))\cdot\frac{\epsilon^2}{4(1+\epsilon)}\right)\cdot\frac{t_{i-1}}{p_{i-1}} && K = 2(1+\epsilon)/\epsilon
\end{aligned}$$

Therefore, the first stage ends after at most $(1+o(1))\frac{4(1+\epsilon)}{\epsilon^2}\ln(1.1Ke^{2/K})$ iterations. Let j be the first iteration when the second stage begins. For $i > j$, we show that $1/\alpha_i$ has an exponential tower growth.

$$\begin{aligned}
\alpha_i &= (1-\pi_i)^{p'_i}\\
&\le \exp\left(-(1-o(1))\frac{1}{K}\cdot\frac{p_i}{t_{i-1}}\right) && 1-x\le e^{-x}\\
&\le \exp\left(-(1-o(1))\frac{1}{K}\cdot\frac{\beta_i p_{i-1}}{t_{i-1}}\right) && \text{defn. } p_i\\
&\le \exp\left(-(1-o(1))\frac{1}{K}\cdot\frac{\beta_{i-1}}{\alpha_{i-1}}\cdot\frac{\beta_i p_{i-2}}{t_{i-2}}\right) && \frac{p_{i-1}}{t_{i-1}} = \frac{\beta_{i-1}}{\alpha_{i-1}}\frac{p_{i-2}}{t_{i-2}}\\
&\le \exp\left(-(1-o(1))\frac{1}{K}\cdot\frac{e^{-2/K}}{\alpha_{i-1}}\cdot\frac{p_{i-2}}{t_{i-2}}\right) && \beta_i\ge e^{-1/K}\\
&\le \exp\left(-1/\alpha_{i-1}\right) && \frac{t_{i-2}}{p_{i-2}} < \frac{1}{1.1Ke^{2/K}}
\end{aligned}$$

Therefore, $\frac{1}{\alpha_{j+\log^* D+1}} \ge \underbrace{e^{e^{\cdot^{\cdot^{e}}}}}_{\log^* D} \ge D$, and so $t_{j+\log^* D+1} \le \max(\alpha_{j+\log^* D+1}\cdot D, T) = T$. □

On the other hand, we show the bound on the palette size remains large throughout the algorithm.

Lemma 14 *$p'_i = D^{1-o(1)}$ for $i = O(\log^* D)$.*

Proof $p'_i = (1-\delta)^i p_i \ge (1-\delta)^i\prod_{j=1}^{i}\beta_j D \ge (1-\delta)^i e^{-i/K}D = (1-o(1))D^{-\frac{i}{K\log D}}\cdot D = D^{1-o(1)}$. □

In the following we shall show how to ensure that for each iteration i the palette sizes are lower bounded by p'_i and the c-degrees are upper bounded by t'_i. For convenience let $H_i(u)$ denote the event that $|P_i(u)| \geq p'_i$ and $\deg_{i,c}(u) \leq t'_i$ for u and $c \in P_{i-1}(u)$. Let H_i denote the event that $H_i(u)$ holds for every $u \in G_i$.

Lemma 15 *Suppose that H_{i-1} holds, then* $\Pr(|P_i(u)| < (1-\delta)\beta_i|P_{i-1}(u)|) < e^{-\Omega(\delta^2 p'_i)}$.

Proof Consider a color $c \in P_{i-1}(u)$. The probability that c remains in $P_i(u)$ is exactly β_i. Since the event that c remains in $P_i(u)$ is independent among other colors, by a Chernoff Bound, $\Pr(|P_i(u)| < (1-\delta)\beta_i|P_{i-1}(u)|) < e^{-\Omega(\delta^2 p_{i-1})}$. □

Lemma 16 *Suppose that H_{i-1} holds, then* $\Pr(\deg_{i,c}(u) > (1+\delta)\cdot\max(\alpha_i \cdot \deg_{i-1,c}(u), T)) < e^{-\Omega(\delta^2 T)} + D \cdot e^{-\Omega(\delta^2 p'_i)}$.

Proof Let $x_1, \ldots x_k \in N_{i-1,c}(u)$ be the c-neighbors of u, ordered by their ID. Let $\mathcal{E}_j$ denote the event that $|P_i(x_j)| \geq p'_i$, where $\Pr(\overline{\mathcal{E}_j}) < e^{-\Omega(\delta^2 p'_i)}$ by Lemma 15.

Let X_i denote the event that x_i remains uncolored after iteration i. Let $\mathbf{X}_j$ denote the shorthand for $(X_1, \ldots, X_j)$. We will show that for any realization of $\mathbf{X}_{j-1}$, $\Pr(X_j \mid \mathbf{X}_{j-1}, \mathcal{E}_1, \ldots, \mathcal{E}_j) \leq \alpha_i$. Then we can apply Lemma 19, which is a variant of Chernoff bound that works when conditioning on a sequence of likely events.

Let $U_2 = N_{i-1}(N_{i,c}(u)) \setminus N_{i,c}(u)$ be the neighbors of the c-neighbors excluding the c-neighbors themselves ($u \in U_2$ unless $\deg_{i-1,c}(u) = 0$). First, notice that the events $\mathbf{X}_{j-1}$ and $\mathcal{E}_1 \ldots, \mathcal{E}_j$ are functions of $S_i(U_2), S_i(x_1), \ldots, S_i(x_{j-1}), K_i(x_1), \ldots, K_i(x_j)$. Therefore, we can instead show that under any realization of $S_i(U_2), S_i(x_1), \ldots, S_i(x_{j-1}), K_i(x_1), \ldots, K_i(x_j)$ subject to the events $\mathcal{E}_1 \ldots, \mathcal{E}_j$ hold, $\Pr(X_j \mid S_i(U_2), S_i(x_1), \ldots, S_i(x_{j-1}), K_i(x_1), \ldots, K_i(x_j)) \leq \alpha_i$.

Obviously for any $c' \in P_{i-1}(x_j)$,

$$\Pr(c' \in S_i(x_j) \mid S_i(U_2), S_i(x_1), \ldots, S_i(x_{j-1}), K_i(x_1), \ldots, K_i(x_j)) = \pi_i.$$

Therefore,

$$\begin{aligned}
&\Pr(X_j \mid S_i(U_2), S_i(x_1), \ldots, S_i(x_{j-1}), K_i(x_1), \ldots, K_i(x_j)) \\
&\leq (1 - \Pr(c' \in S_i(x_j) \mid S_i(U_2), S_i(x_1), \ldots, S_i(x_{j-1}), K_i(x_1), \ldots, K_i(x_j)))^{|P_i(u)|} \\
&\leq (1-\pi_i)^{p'_i} = \alpha_i.
\end{aligned}$$

Therefore, by Lemma 19, Corollary 5, and by the fact that $\sum_j \Pr(\overline{\mathcal{E}_j}) \leq D \cdot e^{-\Omega(\delta^2 p'_i)}$, we have $\Pr(\deg_{i,c}(u) > (1+\delta)\cdot\max(\alpha_i \cdot \deg_{i-1,c}(u), T)) \leq e^{-\Omega(\delta^2 T)} + D \cdot e^{-\Omega(\delta^2 p'_i)}$. □

Corollary 4 *Suppose that H_{i-1} holds,* $\Pr(\overline{H}_i(u)) \leq D \cdot e^{-\Omega(\delta^2 T)} + 2D^2 \cdot e^{-\Omega(\delta^2 p'_i)}$.

Proof By taking union bound over the event in Lemma 15 and the events in Lemma 16 over each $c \in P_{i-1}(u)$, we get the desired result. □

Let r be the first iteration such that $t_r = T$. If H_r holds, then $\deg_{r,c}(u) \leq t'_r \leq (1+\delta)^r t_r \leq (1+o(1))t_r \leq 2T$ for all u and c. Now we switch to the following analysis, which shows the algorithm terminates in a constant number of iterations. For $i > r$, we define $t'_i = t'_{i-1} \cdot \frac{T}{p'_i}$. The definition for the rest of parameters remain the same. By Lemma 14, if D is large enough, we can assume that $p'_i \geq D^{1-0.8\gamma}$ for $i = r + \lceil 1/(0.1\gamma) \rceil$, since $r + \lceil 1/(0.1\gamma) \rceil = O(\log^* D)$. Then from the definition of t'_i, it shrinks to less than one in $\lceil \frac{1}{0.1\gamma} \rceil$ iterations, since $T/p'_i \leq D^{-0.1\gamma}$ and $t'_{r+1/(0.1\gamma)} < (D^{-0.1\gamma})^{\lceil 1/(0.1\gamma) \rceil} \cdot t'_r < 1$.

Now we will show that under this new definition of t_i for $i > r$, $H_i(u)$ is likely to hold, provided that H_{i-1} holds.

Lemma 17 *Suppose that H_{i-1} is true where $i > r$, then* $\Pr(\deg_{i,c}(u) > t'_i) < e^{-\Omega(T)} + D \cdot e^{-\Omega(\delta^2 p'_i)}$

Proof Let $x_1, \ldots x_k \in N_{i-1,c}(u)$ be the c-neighbors of u, ordered by their ID in the increasing order. Let $\mathcal{E}_j$ denote the event that $|P_i(x_j)| \geq p'_i$. Note that $\Pr(\overline{\mathcal{E}}_j) \leq e^{-\Omega(\delta^2 p'_i)}$. As we have shown in the proof of Lemma 16, $\Pr(X_j \mid \mathbf{X}_j, \mathcal{E}_1, \ldots, \mathcal{E}_j) \leq \alpha_i$. Therefore,

$$\begin{aligned}
&\Pr(\deg_{i,c}(u) > t'_i) \\
&= \Pr\left(\deg_{i,c}(u) > \left(\frac{t'_{i-1}}{\alpha_i t'_{i-1}}\right) \cdot \alpha_i t'_{i-1}\right)
\end{aligned}$$

Applying Lemma 19 and Corollary 5 with $1+\delta = t'_i/(\alpha_i t'_{i-1})$, and noticing that $\alpha_i \deg_{i-1,c}(u) \leq \alpha_i t'_{i-1}$, the probability above is bounded by

$$\begin{aligned}
&\leq \exp\left(-\alpha_i t'_{i-1}\left(\frac{t'_i}{\alpha_i t'_{i-1}} \ln \frac{t'_i}{\alpha_i t'_{i-1}} - \left(\frac{t'_i}{\alpha_i t'_{i-1}} - 1\right)\right)\right) \\
&\quad + De^{-\Omega(\delta^2 p'_i)} \\
&\leq \exp\left(-t'_i\left(\ln \frac{t'_i}{\alpha_i t'_{i-1}} - 1\right)\right) + De^{-\Omega(\delta^2 p'_i)} \\
&= \exp\left(-t_i\left(\ln\left(\frac{1}{\alpha_i}\right) - \ln\left(\frac{et'_{i-1}}{t'_i}\right)\right)\right) \\
&\quad + De^{-\Omega(\delta^2 p'_i)}
\end{aligned}$$

$$
\begin{aligned}
&\le \exp\left(-t'_i\left((1-o(1))\,\frac{p'_i}{Kt'_{i-1}} - \ln\left(\frac{et'_{i-1}}{t'_i}\right)\right)\right)\\
&\quad + De^{-\Omega(\delta^2 p'_i)} \quad \ln\frac{1}{\alpha_i} = (1-o(1))\frac{p'_i}{Kt'_{i-1}}\\
&\le \exp\left(-\left((1-o(1))\frac{T}{K} - t'_i\ln(eD)\right)\right)\\
&\quad + De^{-\Omega(\delta^2 p'_i)} \qquad \text{defn. } t'_i \text{ and } t'_{i-1}/t'_i < D\\
&\le \exp\left(-T\left(\frac{(1-o(1))}{K} - \frac{t'_{i-1}}{p'_i}\ln(eD)\right)\right)\\
&\quad + De^{-\Omega(\delta^2 p'_i)}\\
&\le \exp\left(-T\left((1-o(1))\frac{1}{K} - \frac{2\ln(eD)}{D^{0.1\gamma}}\right)\right)\\
&\quad + De^{-\Omega(\delta^2 p'_i)} \quad t'_{i-1}p'_i \le \frac{2T}{p'_i} \le \frac{2}{D^{0.1\gamma}}\\
&\le \exp\left(-\Omega(T)\right) + De^{-\Omega(\delta^2 p'_i)}
\end{aligned}
$$

□

Suppose that H_{i-1} holds, by taking the union bound over all the events $P_i(u) \ge p'_i$ for all $u \in G_{i-1}$ and $\Pr(\deg_{i,c}(u) > t'_i)$ for all $u \in G_{i-1}$ and all $c \in P_{i-1}(u)$, we get that $\Pr(\overline{H}_i(u)) \le D \cdot e^{-\Omega(T)} + 2D^2 \cdot e^{-\Omega(\delta^2 p'_i)}$.

Therefore, we conclude that for each iteration $i \ge 1$, if H_{i-1} holds, then $\Pr(\overline{H}_i(u)) \le D \cdot \exp(-\Omega(\delta^2 T)) + 2D^2 \cdot \exp(-\Omega(\delta^2 p'_i)) \le \exp(-D^{1-0.95\gamma})$ for large enough D. Now we want to ensure that H_i holds for every iteration i. If H_{i-1} is true, then $\Pr(\overline{H}_i(u)) \le \exp\left(-D^{1-0.95\gamma}\right)$. If $D^{1-\gamma} \ge \log n$, then each of the bad events occur with probability at most $1/\operatorname{poly}(n)$. Since there are $O(n)$ events, by the union bound, H_i holds w.h.p. On the other hand, if $D^{1-\gamma} \le \log n$, then we can use the LLL algorithm to make H_i hold w.h.p. The probability of the failure events are bounded by $p = \exp\left(-D^{1-0.95\gamma}\right)$. Each event depends on at most $d = O(\Delta^2)$ other events, since each event only depends on the outcomes of the random variables in its neighborhood. Therefore, $epd^2 \le \exp(-D^{1-\gamma})$ and we can apply the simple LLL algorithm to make all the events hold w.h.p. in $O(\log_{1/epd^2} n) \le O(\log n/D^{1-\gamma})$ iterations.

By Lemma 13 and the fact that t_i shrinks to 1 in a constant number of iterations after $i > r$, the algorithm uses $O(\log^* D)$ iterations. Each iteration uses $\max(1, O(\log n/D^{1-\gamma}))$ rounds. The total number of rounds is therefore $O(\log^* D \cdot \max(1, O(\log n/D^{1-\gamma})))$.

5 Discussion

We gave distributed LLL algorithms under the conditions $p \cdot f(d) < 1$ for different functions $f(d)$. When $f(d) = e(d+1)$ that matches the general condition of LLL, our weak-MIS resampling algorithm gives a running time of $O(\log^2 d \cdot \log_{1/ep(d+1)} n)$. Note that the weak-MIS algorithm was later applied in local computation algorithms for computing MIS [25]. Recently, Ghaffari's new MIS algorithm [15] can compute a weak-MIS in $O(\log d)$ time, which improves the overall running time for LLL to $O(\log d \cdot \log_{1/ep(d+1)} n)$.

The lower bound we showed in this paper is $\Omega(\log^* n)$. Very recently, Brandt et al. [8] obtained an $\Omega(\log\log n)$ lower bound for LLL from the *sinkless orientation* problem and the *sinkless coloring* problem in 3-regular graphs. Subsequently, Chang, Kopelowitz, and Pettie generalized [8] to show an $\Omega(\log_d n)$ lower bound for deterministic LLL algorithms and an $\Omega(\log_d \log n)$ lower bound for randomized LLL algorithms [10]. Note that the lower bounds they have obtained requires $f(d)$ to be upper bounded by 2^d, while ours allows it to grow unbounded.

Acknowledgements Thanks Mohsen Ghaffari for pointing out that by iteratively applying LLL, the range of f can be improved from $\Omega(\log \Delta)$ to any positive integer for f-defective, $O(\Delta/f)$-colorings.

Appendix: Tools

Lemma 18 (Chernoff Bound) *Let $X_1, \ldots, X_n$ be indicator variables such that $\Pr(X_i = 1) = p$. Let $X = \sum_{i=1}^n X_i$. Then, for $\delta > 0$:*

$$
\begin{aligned}
\Pr(X \ge (1+\delta)\,\mathrm{E}[X]) &< \left[\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right]^{\mathrm{E}[X]}\\
\Pr(X \le (1-\delta)\,\mathrm{E}[X]) &< \left[\frac{e^\delta}{(1-\delta)^{(1-\delta)}}\right]^{\mathrm{E}[X]}
\end{aligned}
$$

The two bounds above imply that for $0 < \delta < 1$, we have:

$$
\begin{aligned}
\Pr(X \ge (1+\delta)\,\mathrm{E}[X]) &< e^{-\delta^2\,\mathrm{E}[X]/3}\\
\Pr(X \le (1-\delta)\,\mathrm{E}[X]) &< e^{-\delta^2\,\mathrm{E}[X]/2}.
\end{aligned}
$$

Lemma 19 *Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be (likely) events and $X_1, \ldots, X_n$ be indicator variables such that for each $1 \le i \le n$ and $X = \sum_{i=1}^n X_i$,*

$$\max_{\boldsymbol{X}_{i-1}} \Pr(X_i \mid \boldsymbol{X}_{i-1}, \mathcal{E}_1, \ldots \mathcal{E}_i) \le p$$

where $\boldsymbol{X}_i$ denotes the shorthand for $(X_1, \ldots, X_i)$.[4] *Then for $\delta > 0$:*

$$\Pr\left((X > (1+\delta)np) \cap \left(\bigcap_i \mathcal{E}_i\right)\right) \le \left[\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right]^{np}$$

[4] We slightly abuse the notation that when conditioning on the random variable X_i, it means X_i may take arbitrary values, whereas when conditioning on the event $\mathcal{E}_i$, it means that $\mathcal{E}_i$ happens.

and thus by the union bound,

$$\Pr(X > (1+\delta)np) \le \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{np} + \sum_i \Pr(\overline{\mathcal{E}_i}).$$

Proof For now let us treat $\mathcal{E}_i$ as 0/1 random variables and let $\mathcal{E} = \prod_i \mathcal{E}_i$. For any $t > 0$,

$$\begin{aligned}
&\Pr\left((X > (1+\delta)np) \cap \left(\bigcap_i \mathcal{E}_i\right)\right) \qquad (6)\\
&= \Pr\left(\left(\prod_{i=1}^{n} \mathcal{E}_i\right) \cdot \exp(tX) > \exp(t(1+\delta)np)\right)\\
&\le \frac{\mathrm{E}\left[\left(\prod_{i=1}^{n} \mathcal{E}_i\right) \cdot \exp(tX)\right]}{\exp(t(1+\delta)np)}\\
&= \frac{\mathrm{E}\left[\left(\prod_{i=1}^{n} \mathcal{E}_i \cdot \exp(tX_i)\right)\right]}{\exp(t(1+\delta)np)} \qquad (7)
\end{aligned}$$

We will show by induction that

$$\mathrm{E}\left[\left(\prod_{i=1}^{k} \mathcal{E}_i \exp(tX_i)\right)\right] \le (1 + p(e^t - 1))^k$$

When $k = 0$, it is trivial that $\mathrm{E}[\mathcal{E}] \le 1$.

$$\begin{aligned}
&\mathrm{E}\left[\left(\prod_{i=1}^{k} \mathcal{E}_i \exp(tX_i)\right)\right]\\
&\le \mathrm{E}\left[\left(\prod_{i=1}^{k-1} \mathcal{E}_i \exp(tX_i)\right)\right.\\
&\quad \left.\cdot \mathrm{E}\left[\mathcal{E}_k \exp(tX_k) \mid \boldsymbol{X_{i-1}}, \mathcal{E}_1, \ldots, \mathcal{E}_{k-1}\right]\right]\\
&= \mathrm{E}\left[\left(\prod_{i=1}^{k-1} \mathcal{E}_i \exp(tX_i)\right)\right.\\
&\quad \left.\cdot \Pr(\mathcal{E}_k) \cdot \mathrm{E}\left[\exp(tX_k) \mid \boldsymbol{X_{i-1}}, \mathcal{E}_1, \ldots, \mathcal{E}_k\right]\right]\\
&\le \mathrm{E}\left[\left(\prod_{i=1}^{k-1} \mathcal{E}_i \exp(tX_i)\right)\right.\\
&\quad \left.\cdot \mathrm{E}\left[\exp(tX_k) \mid \boldsymbol{X_{i-1}}, \mathcal{E}_1, \ldots, \mathcal{E}_k\right]\right]\\
&= \mathrm{E}\left[\left(\prod_{i=1}^{k-1} \mathcal{E}_i \exp(tX_i)\right)\right.\\
&\quad \left.\cdot(1 + \Pr(X_k \mid \boldsymbol{X}_{i-1}, \mathcal{E}_1, \ldots, \mathcal{E}_k)(e^t - 1))\right]\\
&\le \mathrm{E}\left[\left(\prod_{i=1}^{k-1} \mathcal{E}_i \exp(tX_i)\right) \cdot (1 + p(e^t - 1))\right]\\
&= \mathrm{E}\left[\left(\prod_{i=1}^{k-1} \mathcal{E}_i \exp(tX_i)\right)\right] \cdot (1 + p(e^t - 1))\\
&\le (1 + p(e^t - 1))^k
\end{aligned}$$

Therefore, by (6),

$$\begin{aligned}
&\Pr\left((X > (1+\delta)np) \cap \left(\bigcap_i \mathcal{E}_i\right)\right)\\
&= \frac{\mathrm{E}[\mathcal{E} \cdot \prod_{i=1}^{n} \exp(tX_i)]}{\exp(t(1+\delta)np)}\\
&\le \frac{(1 + p(e^t - 1))^n}{\exp(t(1+\delta)np)}\\
&\le \frac{\exp(np(e^t - 1))}{\exp(t(1+\delta)np)}\\
&= \left[\frac{\exp(\delta)}{(1+\delta)^{1+\delta}}\right]^{np}.
\end{aligned}$$

The last equality follows from the standard derivation of Chernoff Bound by choosing $t = \ln(1+\delta)$. □

Corollary 5 *Suppose that for any $\delta > 0$,*

$$\Pr\left((X > (1+\delta)np) \cap \left(\bigcap_i \mathcal{E}_i\right)\right) \le \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{np}$$

then for any $M \ge np$ and $0 < \delta < 1$,

$$\begin{aligned}
\Pr\left((X > np + \delta M) \cap \left(\bigcap_i \mathcal{E}_i\right)\right) &\le \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{M}\\
&\le e^{-\delta^2 M/3}
\end{aligned}$$

Proof Without loss of generality, assume $M = tnp$ for some $t \ge 1$, we have

$$\begin{aligned}
&\Pr\left((X > np + \delta M) \cap \left(\bigcap_i \mathcal{E}_i\right)\right)\\
&\le \left[\frac{e^{t\delta}}{(1+t\delta)^{(1+t\delta)}}\right]^{np}\\
&= \left[\frac{e^{\delta}}{(1+t\delta)^{(1+t\delta)/t}}\right]^{M}\\
&\le \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{M} \qquad (*)\\
&\le e^{-\delta^2 M/3} \qquad \frac{e^{\delta}}{(1+\delta)^{(1+\delta)}} \le e^{-\delta^2/3} \text{ for } 0 < \delta < 1
\end{aligned}$$

Inequality (*) follows if $(1+t\delta)^{(1+t\delta)/t} \ge (1+\delta)^{(1+\delta)}$, or equivalently, $((1+t\delta)/t)\ln(1+t\delta) \ge (1+\delta)\ln(1+\delta)$. Letting $f(t) = ((1+t\delta)/t)\ln(1+t\delta) - (1+\delta)\ln(1+\delta)$, we have $f'(t) = \frac{1}{t^2}(\delta t - \ln(1+\delta t)) \ge 0$ for $t > 0$. Since $f(1) = 0$ and $f'(t) \ge 0$ for $t > 0$, we must have $f(t) \ge 0$ for $t \ge 1$. □

References

1. Alon, N.: A parallel algorithmic version of the local lemma. Random Struct. Algorithms **2**(4), 367–378 (1991)
2. Alon, N., Babai, L., Itai, A.: A fast and simple randomized parallel algorithm for the maximal independent set problem. J. Algorithms **7**(4), 567–583 (1986)
3. Alon, N., Krivelevich, M., Sudakov, B.: Coloring graphs with sparse neighborhoods. J. Comb. Theory Ser. B **77**(1), 73–82 (1999)
4. Barenboim, L., Elkin, M.: Distributed Graph Coloring: Fundamentals and Recent Developments Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool, San Rafael (2013)
5. Barenboim, L., Elkin, M., Kuhn, F.: Distributed $(\Delta + 1)$-coloring in linear (in Δ) time. SIAM J. Comput. **43**(1), 72–95 (2014)
6. Barenboim, L., Elkin, M., Pettie, S., Schneider, J.: The locality of distributed symmetry breaking. J. ACM **63**(3), 1–20 (2016)
7. Beck, J.: An algorithmic approach to the Lovász local lemma. I. Random Struct. Algorithms **2**(4), 343–365 (1991)
8. Brandt, S., Fischer, O., Hirvonen, J., Keller, B., Lempiäinen, T., Rybicki, J., Suomela, J., Uitto, J.: A lower bound for the distributed lovász local lemma. In: Proceedings of 48th ACM Symposium on Theory of Computing (STOC), pp. 479–488 (2016)
9. Chandrasekaran, K., Goyal, N., Haeupler, B.: Deterministic algorithms for the Lovász local lemma. SIAM J. Comput. **42**(6), 2132–2155 (2013)
10. Chang, Y., Kopelowitz, T., Pettie, S.: An exponential separation between randomized and deterministic complexity in the LOCAL model. In: Proceedings of 57th Symposium on Foundations of Computer Science (FOCS), pp. 195–197 (2016)
11. Czumaj, A., Scheideler, C.: A new algorithm approach to the general Lovász local lemma with applications to scheduling and satisfiability problems (extended abstract). In: Proceedings of 32nd ACM Symposium on Theory of Computing (STOC), pp. 38–47 (2000)
12. Dubhashi, D., Grable, D.A., Panconesi, A.: Near-optimal, distributed edge colouring via the nibble method. Theor. Comput. Sci. **203**(2), 225–251 (1998)
13. Elkin, M., Pettie, S., Su, H.-H.: $(2\Delta - 1)$-edge-coloring is much easier than maximal matching in the distributed setting. In: Proceedings of 26th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 355–370 (2015)
14. Erdős, P., Lovász, L.: Problems and results on 3-chromatic hypergraphs and some related questions. In: Hanjal, A., Rado, R., Sós, V.T. (eds.) Infinite and Finite Sets, vol. 11, pp. 609–627. North-Holland, Amsterdam (1975)
15. Ghaffari, M.: An improved distributed algorithm for maximal independent set. In: Proceedings of 27th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 270–277 (2016)
16. Haeupler, B., Saha, B., Srinivasan, A.: New constructive aspects of the Lovász local lemma. J. ACM **58**(6), 28 (2011)
17. Harris, D.G.: Lopsidependency in the Moser–Tardos framework: beyond the lopsided Lovász local lemma. In: Proceedings of 26th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1792–1808 (2015)
18. Harris, D.G., Srinivasan, A.: The Moser-Tardos framework with partial resampling. In: Proceedings of 54th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 469–478 (2013)
19. Harris, D.G., Srinivasan, A.: A constructive algorithm for the Lovász local lemma on permutations. In: Proceedings of 25th ACM-SIAM Symposium on Discrete Algorithms (SODA) pp. 907–925 (2014)
20. Haxell, P.E.: A note on vertex list colouring. Comb. Probab. Comput. **10**(4), 345–347 (2001)
21. Hind, H., Molloy, M., Reed, B.: Colouring a graph frugally. Combinatorica **17**(4), 469–482 (1997)
22. Kolipaka, K., Szegedy, M.: Moser and Tardos meet Lovász. In: Proceedings 43rd ACM Symposium on Theory of Computing (STOC), pp. 235–244 (2011)
23. Kuhn, F., Moscibroda, T., Wattenhofer, R.: Local computation: lower and upper bounds. J. ACM **63**(2), 17 (2016)
24. Kuhn, F., Wattenhofer, R.: On the complexity of distributed graph coloring. In: Proceedings 25th Annual ACM Symposium on Principles of Distributed Computing (PODC), pp. 7–15 (2006)
25. Levi, R., Rubinfeld, R., Yodpinyanee, A.: Local computation algorithms for graphs of non-constant degrees. Algorithmica, pp. 1–24 (2016)
26. Linial, N.: Locality in distributed graph algorithms. SIAM J. Comput. **21**(1), 193–201 (1992)
27. Luby, M.: A simple parallel algorithm for the maximal independent set problem. SIAM J. Comput. **15**(4), 1036–1053 (1986)
28. Molloy, M., Reed, B.: Further algorithmic aspects of the local lemma. In: Proceedings of 30th ACM Symposium on Theory of Computing (STOC), pp. 524–529 (1998)
29. Molloy, M., Reed, B.: Graph Colouring and the Probabilistic Method. Algorithms and Combinatorics. Springer, Berlin (2001)
30. Molloy, M., Reed, B.: Asymptotically optimal frugal colouring. J. Comb. Theory Ser. B **100**(2), 226–246 (2010)
31. Moser, R.A.: Derandomizing the Lovász local lemma more effectively. CoRR, abs/0807.2120 (2008)
32. Moser, R.A.: A constructive proof of the Lovász local lemma. In: Proceedings of 41st ACM Symposium on Theory of Computing (STOC), pp. 343–350 (2009)
33. Moser, R.A., Tardos, G.: A constructive proof of the general Lovász local lemma. J. ACM **57**(2), 11 (2010)
34. Pegden, W.: An extension of the Moser–Tardos algorithmic local lemma. SIAM J. Discrete Math. **28**(2), 911–917 (2014)
35. Peleg, D.: Distributed Computing: A Locality-Sensitive Approach. Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, Philadelphia (2000)
36. Pemmaraju, S., Srinivasan, A.: The randomized coloring procedure with symmetry-breaking. In: Proceedings of 35th Int'l Colloq. on Automata, Languages, and Programming (ICALP), pp. 306–319 (2008)
37. Pettie, S., Su, H.-H.: Fast distributed coloring algorithms for triangle-free graphs. In: Proceedings of 40th Int'l Colloq. on Automata, Languages, and Programming (ICALP), pp. 687–699, (2013)
38. Reed, B.: The list colouring constants. J. Graph Theory **31**(2), 149–153 (1999)
39. Reed, B., Sudakov, B.: Asymptotically the list colouring constants are 1. J. Comb. Theory Ser. B **86**(1), 27–37 (2002)
40. Schneider, J., Wattenhofer, R.: A new technique for distributed symmetry breaking. In: Proceedings of 29th ACM Symposium on Principles of Distributed Computing (PODC), pp. 257–266 (2010)
41. Spencer, J.: Asymptotic lower bounds for Ramsey functions. Discret. Math. **20**, 69–76 (1977)
42. Srinivasan, A.: Improved algorithmic versions of the Lovász local lemma. In: Proceedings of 19th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 611–620 (2008)