Algorithmic Improvements of the Lovász Local Lemma via Cluster Expansion *

Dimitris Achlioptas[1] and Themis Gouleakis[2]

1 **Department of Informatics and Telecommunications, University of Athens Athens, Greece**
optas@di.uoa.gr

2 **Department of Electrical Engineering and Computer Science, M.I.T. Cambridge, MA, USA**
tgoule@mit.edu

Abstract

The Lovász Local Lemma (LLL) is a powerful tool that can be used to prove that an object having none of a set of bad properties exists, using the probabilistic method. In many applications of the LLL it is also desirable to explicitly construct the combinatorial object. Recently it was shown that this is possible using a randomized algorithm in the full asymmetric LLL setting [R. Moser and G. Tardos, 2010]. A strengthening of the LLL for the case of dense local neighborhoods proved in [R. Bissacot et al., 2010] was recently also made constructive in [W. Pegden, 2011]. In another recent work [B. Haupler, B. Saha, A. Srinivasan, 2010], it was proved that the algorithm of Moser and Tardos is still efficient even when the number of events is exponential. Here we prove that these last two contributions can be combined to yield a new version of the LLL.

1998 ACM Subject Classification G.2.1 Combinatorics; G.3 Probabilistic Algorithms

Keywords and phrases Probabilistic Method, Lovász Local Lemma, Algorithms

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

The Lovász Local Lemma (LLL) [4] states that if one has a collection $\mathcal{A}$ of "bad" events in a probability space and each event is independent of all but "a few" other events in $\mathcal{A}$, then the probability that none of the bad events occurs is bounded away from 0. (We will give a precise and constructive statement shortly.)

Over the years there have been numerous efforts [1, 2, 7, 11, 8] to make the LLL constructive, culminating with the recent breakthrough of Moser and Tardos [9]. Specifically, in [9] one assumes that there is a finite set of n mutually independent random variables $\mathcal{P} = \{P_i\}$ and that each event in $\mathcal{A}$ is determined by a subset of $\mathcal{P}$. Let the dependency graph $G(\mathcal{A})$ have the events of $\mathcal{A}$ as its vertices and let two events A_i, A_j be deemed adjacent, denoted as $A_i \sim A_j$, iff they share at least one variable, i.e., $\mathrm{vbl}(A_i) \cap \mathrm{vbl}(A_j) \neq \emptyset$. The inclusive neighborhood of each event consists of itself and its adjacent events in G. (While this setting is not quite as general as the original LLL setting, as shown in [6], it suffices for the vast majority of applications of the LLL.) The goal is to find an assignment of values to the n

* Research supported by ERC IDEAS Starting Grant RIMACO and a Sloan Research Fellowship. Part of this work was done while the first author was at UC Santa Cruz and the second author was with the National Technical University of Athens.

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 1–8

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

variables in $\mathcal{P}$ such that none of the events in $\mathcal{A}$ occurs. To that end, Moser and Tardos [9] showed that the following astonishingly simple algorithm suffices.

ALGORITHM MT

- Initialize the variables in $\mathcal{P}$ by sampling from their product measure.
- While any event in $\mathcal{A}$ occurs, select any occurring event and resample its variables according to their product measure.

▶ **Theorem 1** ([9]). *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 with a dependency graph G determined by these variables. Let Γ_i denote the inclusive neighborhood of event A_i in G.*

If there exists $\mu : \mathcal{A} \to (0, +\infty)$ such that

$$\forall A_i \in \mathcal{A} : \Pr[A_i] \leq \mu(A_i) \cdot \left(\sum_{U \subseteq \Gamma_i} \prod_{B \in U} \mu(B) \right)^{-1}, \tag{1}$$

then the expected number of resamplings performed by the randomized algorithm MT is bounded by $\sum_{A_i \in \mathcal{A}} \mu(A_i) = O(|A|) = O(m)$.

Clearly, since the running time of the algorithm is finite, we can trivially conclude that whenever a collection $\mathcal{A}_i$ of events satisfies (1), i.e., the LLL conditions, there exists an assignment to the variables such that none of them occurs.

In [3], Bissacot et al. gave a non-constructive strengthening of the LLL using cluster-expansion methods from statistical physics. Specifically, they proved that in the hypothesis of the LLL, in the rhs of (1), one can replace the summation over all subsets $U \subseteq \Gamma_i$ with a summation over subsets forming independent sets in $G(\mathcal{A})$. This improvement is significant if the subgraphs induced by the vertex neighborhoods in G are dense. On the other hand, the improvement is vanishing if the dependency graph is triangle-free.

In [10], Pegden made the result of Bissacot et al. constructive, by proving a theorem identical to Theorem 1, except for the aforementioned change in the range of the summation.

In a different direction, Haeupler, Saha, and Srinivasan [5] gave a more careful analysis of the MT algorithm, establishing that under slightly stronger assumptions the running time of the algorithm is independent of the number of events. This speedup can be exponential in certain applications.

▶ **Theorem 2** ([5]). *In the setting of Theorem 1, assume that there is $\varepsilon > 0$ such that*

$$\forall A_i \in \mathcal{A} : \Pr[A_i] \leq (1-\varepsilon)\mu(A_i) \cdot \left(\sum_{U \subseteq \Gamma_i} \prod_{B \in U} \mu(B) \right)^{-1}. \tag{2}$$

Then the expected number of resamplings performed by algorithm MT is $O(\varepsilon^{-1} n \log(n/\varepsilon))$.

1.1 Our Results

We prove that the improvements of [10] and [5] stated earlier can be combined, yielding the strongest form of the LLL known to us.

As in the setting of Moser and Tardos [9], 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 the variables in $\mathcal{P}$ and let $G = G(\mathcal{A})$ be the the dependency graph of the events. Recall that Γ_i denotes the inclusive neighborhood of (the vertex corresponding to) each event A_i in G.

▶ **Definition 3.** For each Γ_i, let I_i consist of all subsets of Γ_i that are independent in $G(\mathcal{A})$. Let also $S_i = I_i \cup \{I \cup \{A_i\} : I \in I_i\}$, i.e., S_i contains every independent set of Γ_i along with its version enhanced by the addition of A_i.

▶ **Theorem 4.** *If there exists $\mu : \mathcal{A} \to (0, +\infty)$ and $\varepsilon > 0$ such that*

$$\forall A_i \in \mathcal{A} : \Pr[A_i] \leq (1-\varepsilon)\mu(A_i) \cdot \left(\sum_{U \in I_i} \prod_{B \in U} \mu(B) \right)^{-1} , \tag{3}$$

then the expected number of resamplings performed by MT is $O\left(\varepsilon^{-1} n \log\left(\sum_{A_i \in \mathcal{A}} \mu(A_i)\right)\right)$.

By slightly strengthening the condition in (3) the bound on the running time can be replaced by an expression that, in most applications, is independent of the number of events.

▶ **Theorem 5.** *In the setting of Theorem 4, assume that (3) holds if the summation is extended over all sets $U \in S_i$ (rather than over all sets $U \in I_i$). Then the expected number of resamplings performed by MT is bounded by $O(\varepsilon^{-1} n \log(Z/\varepsilon))$, where*

$$Z = Z(\mu) = \sum_{A_i \in \mathcal{A}} \frac{\mu(A_i)}{1 + \mu(A_i)} .$$

Clearly, the bound on the number of resamplings of Theorem 5 is no greater than the bound in Theorem 4, since each term in the sum is now divided by $1+\mu(A_i)$. It was further shown in [5], that for most applications $Z = O(n \log n)$, implying that the expected number of resamplings in the conclusion of Theorem 5 is $O(\varepsilon^{-1} n \log(n/\varepsilon))$.

Indeed, our approach shows that all results from [5], e.g., the results regarding the case $\varepsilon = 0$, hold under the relaxed conditions in which one sums only over independent sets (or over the sets S_i if one wants to replace $\sum \mu(A_i)$ with $Z(\mu)/\varepsilon$ in the running time). As the proof technique is identical for all cases, we only show here the proofs of Theorems 4 and 5.

2 Witness Trees

2.1 Definition

A witness tree W is a finite, rooted tree where each vertex is labelled by an event. The analysis of the algorithm in [9] was made by mapping each prefix of the sequence of resampled events to a witness tree. Specifically, let L be the sequence of resampled events and let $L(t) = [A_{i_1}, A_{i_2}, ..., A_{i_t}]$ be the first t resampled events. The witness tree $W_L(t)$ is a *labelled* tree constructed as follows.

- Initialize $W_L(t)$ to a single node r (the root) with label A_{i_t} and depth $d(r) = 0$.
- For s from $t-1$ down to 1 do:
  1. Seek the deepest node of $W_L(t)$ whose event (label) is adjacent to A_{i_s} in $G(\mathcal{A})$.
  2. If you find such a node b, then update $W_L(t)$ by adding a node v with label A_{i_s} as a child of b and let $d(v) = d(b) + 1$.
  3. If no such b exists, do nothing.

▶ Remark. If $t_1 \neq t_2$ then $W_L(t_1) \neq W_L(t_2)$. (To see this observe that the witness tree associated with the k-th resampling of A_i will be the only witness tree with A_i as the root and exactly k nodes with label A_i.)

Moser and Tardos [9] provided an upper bound for the expected number of resamplings by studying the size of the witness trees generated by executions of their algorithm. Specifically, they proved the following upper bound on the probability that a specific witness tree W is ever generated for a given dependency graph G.

▶ **Lemma 6** ([9]). *For any witness tree W, let $M = M(W)$ be the multiset containing the labels of the vertices of W. Let X_W be the indicator random variable that W occurs in L.* $\Pr[X_W = 1] \leq \prod_{i \in M} \Pr[A_i]$.

The above lemma is used in [9] to bound the expected number of resamplings as follows. Let T be the random variable equal to the number of resamplings and let W_i be the set of all possible witness tress with root A_i. Since $T = |L|$ and each fixed witness tree occurs at most once in L, we get

$$\mathbb{E}[T] = \mathbb{E}[|L|] = \mathbb{E}\left[\sum_{A_i \in \mathcal{A}} \sum_{W \in W_i} X_W\right] = \sum_{A_i \in \mathcal{A}} \sum_{W \in W_i} \mathbb{E}[X_W] = \sum_{A_i \in \mathcal{A}} \sum_{W \in W_i} \Pr[X_W = 1] \ . \quad (4)$$

2.2 Random generation via branching process

To bound the sum in (4), Moser and Tardos [9] defined the set of *proper* witness trees to consist of all rooted, labelled trees in which the children of each node have distinct labels. Note that the set of proper witness trees contains all witness trees that can be generated by algorithm MT, since there cannot be two nodes with the same label at the same depth of the tree (the node that was added later must have gone at a strictly larger depth, if nothing else as a child of the earlier node).

To bound the total probability of all proper witness trees with root A_i, they defined the Galton-Watson branching process below and proved that the probability the algorithm generates any particular such witness tree W, i.e., $\Pr[X_W = 1]$, is bounded by a constant C times the probability that W is generated by the branching process. Therefore, each of the m events A_i is expected to appear at most C times in L and, thus, the expected total number of resamplings is bounded by Cm.

- Let $g(A_i) = \mu(A_i)(1 + \mu(A_i))^{-1}$.
- Create a single node r (the root) with label A_i and depth $d(r) = 0$.
- Let $d = 0$.
- **Repeat**
  1. For each node A at depth d consider its neighboring events in the dependency graph. For each such event B, with probability $g(B)$ add to A a child node with label B.
  2. $d \leftarrow d + 1$.
- **until** there is no node at depth d.

The bound on $\Pr[X_W = 1]$ comes by first applying Lemma 6, then substituting the bounds on $\Pr[A_i]$ from the LLL conditions, and finally relating the resulting expression to the exact expression for the probability that W is generated by the branching process.

2.3 The improvement of Pegden

Observe that since witness trees for the MT algorithm are grown by adding each event to the deepest possible level, the nodes at every level are labelled by events that form an independent set in the dependency graph G. Using this observation, in [10], Pegden defined witness trees in which *sibling* nodes must have non-adjacent events as *strongly proper* and

modified the Galton Watson process above as follows. To generate the progeny of each node we generate progeny-samples as before, i.e., via an independent trial for each neighboring event, but continue to reject until we get a progeny that forms an independent set in G. For this modified branching process, Pegden proved the following.

▶ **Lemma 7** ([10]). *For any strongly proper witness tree W with root labelled A_i, the probability that the branching process described above produces W is equal to*

$$p_W \equiv \mu(A_i)^{-1} \prod_{A_j \in W} \mu(A_j) \left(\sum_{U \in I_j} \prod_{B \in U} \mu(B) \right)^{-1} . \tag{5}$$

Moser and Tardos [9] had proved a similar lemma, where the summation is over all subsets of vertices in Γ_j instead of just the independent sets $U \in I_j$.

3 Proof of Theorem 4

We first state without proof a lemma that gives an upper bound on the expectation of a random variable given an exponential tail bound.

▶ **Lemma 8.** *Let X be an integer-valued random variable such that for some $\varepsilon > 0$, $C > 1$, and all $i > 0$, $\Pr[X \geq i] \leq C(1-\varepsilon)^i$. Then $\mathbb{E}[X] \leq \varepsilon^{-1}(1 + \varepsilon + \log C)$.*

Our plan is to bound the expected number of resamplings of the most frequent event. We will do this by bounding the expected size of the largest witness tree and using the fact that if an event A is resampled r times, then the witness tree associated with its last resampling will contain at least r nodes, as it will have exactly r nodes with label A. So, an upper bound on the expected size of the largest witness tree gives an upper bound on the expected number of resamplings of the most frequent event. Multiplying by n then gives an upper bound on the expected total number of resamplings.

For any integer k, let $Y(k)$ be the random variable equal to the number of witness trees that occur and have size at least k. Also, let $W_i^s(k)$ denote the set of strongly proper witness trees with root A_i and size at least k. Since the only witness trees that can possibly occur are strongly proper,

$$Y(k) = \sum_{A_i \in \mathcal{A}} \sum_{W \in W_i^s(k)} X_W . \tag{6}$$

By Markov's inequality, the probability that there is at least one witness tree of size at least k is bounded by the expected number of trees of size at least k that occur. Therefore, the probability that the largest witness tree in L has size at least k is bounded by

$$\sum_{A_i \in \mathcal{A}} \sum_{W \in W_i^s(k)} \Pr[X_W = 1] .$$

Under the conditions in (3), for every witness tree W of size at least k, Lemma 6 implies

$$\begin{aligned} \Pr[X_W = 1] &\leq \prod_{A_j \in W} (1-\varepsilon) \cdot \mu(A_j) \left(\sum_{U \in I_j} \prod_{B \in U} \mu(B) \right)^{-1} \\ &\leq (1-\varepsilon)^k \prod_{A_j \in W} \mu(A_j) \left(\sum_{U \in I_j} \prod_{B \in U} \mu(B) \right)^{-1} , \end{aligned} \tag{7}$$

where the second inequality follows from the fact that W has size at least k.

Substituting (7) into (6) we get (8) below and, by Lemma 7 we get (9).

$$\mathbb{E}[Y(k)] \leq (1-\varepsilon)^k \sum_{A_i \in \mathcal{A}} \sum_{W \in W_i^s(k)} \prod_{A_j \in W} \mu(A_j) \left(\sum_{U \in I_j} \prod_{B \in U} \mu(B) \right)^{-1} \tag{8}$$

$$= (1-\varepsilon)^k \sum_{A_i \in \mathcal{A}} \mu(A_i) \sum_{W \in W_i^s(k)} p_W \tag{9}$$

$$\leq (1-\varepsilon)^k \sum_{A_i \in \mathcal{A}} \mu(A_i) \ . \tag{10}$$

Thus,

$$\Pr\left[\max_{W \in L} |W| \geq k\right] \leq \mathbb{E}[Y(k)] \leq (1-\varepsilon)^k \sum_{A_i \in \mathcal{A}} \mu(A_i) \ .$$

Lemma 8 thus implies that the expected size of the largest witness tree is bounded by

$$\varepsilon^{-1} \left(1 + \varepsilon + \log \left(\sum_{A_i \in \mathcal{A}} \mu(A_i) \right) \right) \tag{11}$$

which, as mentioned, implies that the expected number of resamplings of the MT algorithm is $O\left(\varepsilon^{-1} n \log\left(\sum_{A_i \in \mathcal{A}} \mu(A_i)\right)\right)$.

4 Proof of Theorem 5

Recall that our assumption is that there exists a function $\mu : \mathcal{A} \to (0, +\infty)$ such that

$$\forall A_i \in \mathcal{A} : \Pr[A_i] \leq (1-\varepsilon)\mu(A_i) \cdot \left(\sum_{U \in S_i} \prod_{B \in U} \mu(B) \right)^{-1} \ . \tag{12}$$

The idea is to prove that for every function μ satisfying (12), there is another function μ' satisfying (12) with slack somewhat less than ε, but all of whose values are bounded by some constant depending on ε. Observe that if μ' satisfies (12) with any slack, then μ' also satisfies the condition of Theorem 4 with the same slack since in that condition we only sum over $I_i \subseteq S_i$. Thus armed with μ' we first apply Theorem 4 to get a bound on the expected number of resamplings in terms of $\sum \mu'(A_i)$ and then exploit the boundedness of μ' to get a bound on the number of resamplings that depends only on $Z(\mu'), n$ and ε.

Recall that each set S_i contains all subsets of Γ_i that form independent sets in G, i.e., $I_i \subseteq S_i$, along with each element of I_i augmented by A_i. Therefore,

$$\sum_{U \in S_i} \prod_{B \in U} \mu(B) = (1 + \mu(A_i)) \sum_{U \in I_i \setminus \{A_i\}} \prod_{B \in U} \mu(u) \ . \tag{13}$$

Substituting (13) into (12) we get that for each event A_i,

$$\Pr[A_i] \leq (1-\varepsilon) \cdot \frac{\mu(A_i)}{1+\mu(A_i)} \left(\sum_{U \in I_i \setminus \{A_i\}} \prod_{B \in U} \mu(u) \right)^{-1} \ . \tag{14}$$

▶ **Lemma 9.** *If (12) holds for $\mu : \mathcal{A} \to (0, +\infty)$, then there is $\mu' : \mathcal{A} \to (0, 2/\varepsilon - 1)$ for which (12) holds with ε replaced by $\varepsilon/2$.*

Proof. The function $g(x) = x/(1+x)$ with domain $(0, +\infty)$ is strictly increasing and its range is $(0, 1)$. So, for every i, we can find $\mu'(A_i) < \mu(A_i)$ such that

$$\frac{\mu'(A_i)}{1+\mu'(A_i)} = (1-\varepsilon/2)\frac{\mu(A_i)}{1+\mu(A_i)} \tag{15}$$

$$< 1-\varepsilon/2 \ . \tag{16}$$

This choice implies that μ' is bounded, as desired, since starting with (16) we get

$$\frac{\mu'(A_i)}{1+\mu'(A_i)} < 1-\varepsilon/2 \implies \mu'(A_i) < 1+\mu'(A_i) - \frac{\varepsilon}{2}(1+\mu'(A_i)) \implies \mu'(A_i) < \frac{2}{\varepsilon} - 1 \ . \tag{17}$$

Now, starting with (14) and using that $(1-\varepsilon/2)^2 > 1-\varepsilon$ for any $\varepsilon > 0$ we get (18) below. Substituring (15) in (18) yields (19). Using that $0 < \mu' < \mu$ yields (20). Reorganizing the sum in (20) as in the derivation of (13) yields (21), i.e., the conclusion of the lemma.

$$\Pr[A_i] < (1-\varepsilon/2)^2 \cdot \frac{\mu(A_i)}{1+\mu(A_i)} \left(\sum_{U \in I_i \setminus \{A_i\}} \prod_{B \in U} \mu(u) \right)^{-1} \tag{18}$$

$$= (1-\varepsilon/2) \cdot \frac{\mu'(A_i)}{1+\mu'(A_i)} \cdot \left(\sum_{U \in I_i \setminus \{A_i\}} \prod_{B \in U} \mu(u) \right)^{-1} \tag{19}$$

$$< (1-\varepsilon/2) \cdot \frac{\mu'(A_i)}{1+\mu'(A_i)} \cdot \left(\sum_{U \in I_i \setminus \{A_i\}} \prod_{B \in U} \mu'(u) \right)^{-1} \tag{20}$$

$$= (1-\varepsilon/2) \cdot \mu'(A_i) \cdot \left(\sum_{U \in S_i} \prod_{B \in U} \mu'(B) \right)^{-1} \ . \tag{21}$$

◀

By Lemma 9, we know that (21) holds for all i. Since $\mu' > 0$ and $I_i \subseteq S_i$, this immediately implies that for all A_i,

$$\Pr[A_i] \leq (1-\varepsilon/2)\mu'(A_i) \cdot \left(\sum_{U \in I_i} \prod_{B \in U} \mu'(B) \right)^{-1} \ ,$$

which is precisely the condition of Theorem 4. Applying the theorem yields that the expected number of resamplings is $O\left(\varepsilon^{-1} n \log\left(\sum_{A_i \in \mathcal{A}} \mu'(A_i)\right)\right)$. But by (17)

$$\sum_{A_i \in \mathcal{A}} \mu'(A_i) \leq (1 + \max_i \mu'(A_i)) \sum_{A_i \in \mathcal{A}} \frac{\mu'(A_i)}{1+\mu'(A_i)} < (2/\varepsilon) Z(\mu') \ . \tag{22}$$

References

1 Noga Alon. A parallel algorithmic version of the Local Lemma. In *FOCS*, pages 586–593. IEEE Computer Society, 1991.

2 József Beck. An algorithmic approach to the Lovász Local Lemma. I. *Random Struct. Algorithms*, 2(4):343–366, 1991.

3 Rodrigo Bissacot, Roberto Fernández, Aldo Procacci, and Benedetto Scoppola. An improvement of the Lovász Local Lemma via cluster expansion. *Combinatorics, Probability & Computing*, 20(5):709–719, 2011.

4 Paul Erdös and Laszlo Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In A. Hajnal et al., editors, *Infinite and Finite Sets*, volume 11 of *Colloq. Math. Soc. Janos Bolyai*, pages 609–627. North-Holland, 1975.

5 Bernhard Haeupler, Barna Saha, and Aravind Srinivasan. New constructive aspects of the Lovász Local Lemma. In *FOCS*, pages 397–406. IEEE Computer Society, 2010.

6 Kashyap Babu Rao Kolipaka and Mario Szegedy. Moser and Tardos meet Lovász. In *STOC*, pages 235–244. ACM, 2011.

7 Michael Molloy and Bruce A. Reed. Further algorithmic aspects of the Local Lemma. In *STOC*, pages 524–529. ACM, 1998.

8 Robin A. Moser. A constructive proof of the Lovász Local Lemma. In *STOC*, pages 343–350. ACM, 2009.

9 Robin A. Moser and Gábor Tardos. A constructive proof of the general Lovász Local Lemma. *J. ACM*, 57(2), 2010.

10 Wesley Pegden. An improvement of the Moser-Tardos algorithmic local lemma. *CoRR*, abs/1102.2853, 2011.

11 Aravind Srinivasan. Improved algorithmic versions of the Lovász Local Lemma. In *SODA*, pages 611–620. SIAM, 2008.