An algorithmic proof of the Lovasz Local Lemma via resampling oracles

Nick Harvey
University of British Columbia

Jan Vondrak
IBM Almaden
Simultaneously avoiding many events

**Lovasz Local Lemma:** [Erdos-Lovasz ‘75]

Let $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ be events in a probability space. Suppose

- $\mathcal{E}_i$ is jointly independent of all but $d$ events
- $\mathbb{P}[\mathcal{E}_i] \leq 1/ed$ for all $i$

Then $\mathbb{P}[\bigcap_i \overline{\mathcal{E}_i}] > 0$ (typically exponentially small).

---

**Beyond what nature intended:**

Finding a very rare object.
Simultaneously avoiding many events

Lovasz Local Lemma: [Erdos-Lovasz ‘75]

Let $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ be events in a probability space. Suppose

- $\mathcal{E}_i$ is jointly independent of all but $d$ events
- $\mathbb{P}[\mathcal{E}_i] \leq 1/ed$ for all $i$

Then $\mathbb{P}[\bigcap_i \overline{\mathcal{E}_i}] > 0$ (typically exponentially small).

Some applications:

- Near-Ramanujan expanders via 2-lifts [Bilu-Linial]
- $O(\text{congestion} + \text{dilation})$ packet routing [Leighton-Maggs-Rao]
- $O(1)$-approx for Santa Claus problem [Feige], [Haeupler-Saha-Srinivasan]
Example: 2-coloring hypergraphs

- **Given:** System of sets of size $k$, each intersecting $\leq \frac{2^{k-1}}{e}$ sets.
- **Goal:** Color vertices red/blue so that each set has both colors.
- **Random approach:** color each vertex independently red/blue.

- Let $E_i$ be event $i^{th}$ set is all-red or all-blue. Note $P[E_i] = 2^{1-k}$.
- $E_i$ is independent of all but $\frac{2^{k-1}}{e} = d$ sets.
- Since $P[E_i] \leq \frac{1}{ed}$, LLL implies $P[\cap_i \overline{E_i}] > 0$.
- So, there is a coloring where no set is all-red or all-blue.
Algorithmic LLL

• Under the original distribution $\mathbb{P}$, avoiding $E_1, E_2, \ldots, E_n$ is exponentially unlikely.

• Can we find another distribution (i.e. a randomized algorithm) in which it is likely to avoid $E_1, E_2, \ldots, E_n$?

• Yes: Hypergraph case, weaker parameters: [Beck ’91], [Alon ’91], [Molloy-Reed ’98], [Czumaj-Scheideler ’00], [Srinivasan ’08]

Beyond what nature intended:
Amplifying probability of rare event.
Moser-Tardos Variable Model
(dependency based on shared variables)

- Independent random variables \( Y_1, \ldots, Y_m \)
- "Bad" events \( E_1, E_2, \ldots, E_n \)
- \( E_i \) depends on variables \( \text{var}(E_i) \)
- A dependency graph \( G \):
  \( i \sim j \) if \( \text{var}(E_i) \cap \text{var}(E_j) \neq \emptyset \).

Moser-Tardos Algorithm [Moser-Tardos '08]

While some event \( E_i \) occurs
Resample all variables in \( \text{var}(E_i) \)

Theorem: Suppose max-degree < \( d \) and \( \mathbb{P}[E_i] \leq 1/ed \). The algorithm finds a point in \( \bigcap_i \overline{E_i} \) after \( O(n) \) resampling operations, in expectation.
Algorithmic Improvements

- Many extensions of Moser-Tardos:
  - deterministic LLL algorithm [Chandrasekaran-Goyal-Haupler ’10]
  - exponentially many events [Haupler-Saha-Srinivasan ’10]
  - better conditions on probabilities [Kolipaka-Szegedy ’11], [Harris ’14]
- Require variable model (dependencies based on $Y_1,\ldots,Y_m$)

$$(\overline{x}_1 \lor x_2 \lor \overline{x}_3) \land (x_4 \lor \overline{x}_5 \lor x_6) \ldots$$

$k$-SAT

Hypergraph coloring
Algorithmic Improvements

• Original LLL works for arbitrary probability spaces
  — Permutations [Erdos-Spencer ‘91]
  — Hamilton cycles [Albert-Frieze-Reed ’95]
  — Spanning trees [Lu-Mohr-Szekely ‘13]

• Algorithms beyond variable model
  — Permutations [Harris-Srinivasan ‘14]
  — Abstract “flaw-correction” framework
    [Achlioptas-Iliopoulos ‘14], [Kolmogorov ‘15]
Algorithmic Local Lemma for general probability spaces?

• For the LLL in any probability space, can we design a randomized algorithm to quickly find \( \omega \in \bigcap_i \overline{E}_i \) ?

How can algorithm “move about” in general probability space?

• Flaw-correcting actions [Achlioptas-Iliopoulos ‘14]
• Resampling oracles [This paper]
Resampling Oracles

- Consider probability space $\Omega$, measure $\mathbb{P}$, events $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ and dependency relation denoted $\sim$.

- A **resampling oracle** for $\mathcal{E}_i$ is a random function $r_i : \Omega \rightarrow \Omega$
  - Removes conditioning on $\mathcal{E}_i$:
    If $X$ has measure $\mathbb{P}$ cond. on $\mathcal{E}_i$, then $r_i(X)$ has measure $\mathbb{P}$.
  - Does not cause non-neighbor events:
    If $\mathcal{E}_k \sim \mathcal{E}_i$ and $X \notin \mathcal{E}_k$, then $r_i(X) \notin \mathcal{E}_k$. 
Resampling Oracles

- Consider probability space $\Omega$, measure $\mathbb{P}$, events $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ and dependency relation denoted $\sim$.

- A **resampling oracle** for $\mathcal{E}_i$ is a random function $r_i : \Omega \rightarrow \Omega$
  - Removes conditioning on $\mathcal{E}_i$:
    - If $X$ has measure $\mathbb{P}$ cond. on $\mathcal{E}_i$, then $r_i(X)$ has measure $\mathbb{P}$.
  - Does not cause non-neighbor events:
    - If $\mathcal{E}_k \sim \mathcal{E}_i$ and $X \notin \mathcal{E}_k$, then $r_i(X) \notin \mathcal{E}_k$.

**Example**: Resampling Oracle for Hypergraph Coloring
Our Main Result:
An Algorithmic LLL in a General Setting

- Arbitrary probability space $\Omega$
- Events $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$
- An arbitrary graph $G$
- A resampling oracle for each $\mathcal{E}_i$, with respect to $G$

**Theorem:** [H.-Vondrak '15] Suppose max-degree $< d$ and $P[\mathcal{E}_i] \leq 1/ed$. Our algorithm finds a point in $\cap_i \overline{\mathcal{E}}_i$ after $O(n^2)$ resampling operations, with high probability.

- Holds much more generally: Lovasz’s conditions, Shearer’s conditions...
Our Main Result:
An Algorithmic LLL in a General Setting

- **Theorem:** [H.-Vondrak ’15] Suppose max-degree < \( d \) and \( \Pr[\mathcal{E}_i] \leq 1/ed \). Our algorithm finds a point in \( \bigcap_i \mathcal{E}_i \) after \( O(n^2) \) resampling operations, with high probability.

- Holds much more generally: Lovasz’s conditions, Shearer’s conditions...

- We design efficient resampling oracles for essentially every known application of the LLL (and generalizations)
Algorithmic LLL via Resampling Oracles

MIS Resample

Draw $\omega$ from $\mathbb{P}$

Repeat

$I \leftarrow \emptyset$

While there is $i \not\in \Gamma^+(I)$ s.t. $E_i$ occurs in $\omega$

Pick smallest such $i$

$\omega \leftarrow r_i(\omega)$

$I \leftarrow I \cup \{i\}$

End

Until $I = \emptyset$

Output $\omega$

Similar to Moser & Tardos’ parallel algorithm
Resampling Spanning Trees in $K_n$

- Let $T$ be a uniformly random spanning tree in $K_n$.
- For edge set $A$, let $\mathcal{E}_A = \{A \subseteq T\}$.
- **Dependency Graph:** Make $\mathcal{E}_A$ a neighbor of $\mathcal{E}_B$, unless $A$ and $B$ are vertex-disjoint.
- **Resampling oracle $r_A$:**
  - If $T$ uniform conditioned on $\mathcal{E}_A$, want $r_A(T)$ uniform.
  - But, should not disturb edges that are vtx-disjoint from $A$. 
• $\mathcal{E}_A = \{ A \subseteq T \}$.

• **Resampling oracle** $r_A(T)$:
  
  — If $T$ uniform conditioned on $\mathcal{E}_A$, want $r_A(T)$ uniform.
  
  — But, should not disturb edges that are vtx-disjoint from $A$.

  — Contract edges of $T$ vtx-disjoint from $A$
  
  — Delete edges adjacent to $A$
  
  — Let $r_A(T)$ be a uniformly random spanning tree in resulting (multi)-graph.

• **Lemma**: $r_A(T)$ is uniformly random.
Summary

• Our algorithmic proof of LLL works for any probability space and any events, under usual LLL conditions, so long as you can design resampling oracles.

• Efficiency is similar to Moser-Tardos, but quadratically worse

• Analysis is similar to Moser-Tardos, perhaps simpler — no “log” or branching processes

• Works under Lovasz’s condition, cluster expansion condition, Shearer’s condition (no slack needed!)