Lovász Local Lemma [Erdös & Lovász 1975]

**Scribe**: Avi Cohen

\[ \mathcal{A} = \{ A_1, \ldots, A_n \} \]

1. \( \Pr[A_i] \leq p \), \( \forall A_i \)
2. Each event depends on at most \( d \) other events.

**LLL**: \( e \cdot p \cdot (d+1) < 1 \), then \( \Pr[\bigwedge \overline{A_i}] > 0 \)

LLL can be seen as generalizing a local property to a global one:

“If there is positive probability that no bad event occurs in each neighborhood, then there is positive prob. no bad event happens globally.”

**Example 1**: (2-Coloring hypergraphs)

\( e_1, \ldots, e_m \)

\( k \)-uniform hypergraph

Every edge intersects \( d \) other edges.

**LLL**: If \( e(d+1) < 2^{k-1} \), 3 2-coloring of the vertices of the hypergraph s.t. no edge is monochromatic.

**Example 2**: \( k \)-CNF

\( n \) vars, \( m \) clauses.

- Clause has \( k \) vars.
- Every var in \( \leq \frac{2^k}{k^e} \) clauses.

\( (x_1 \lor x_2 \lor x_3) \land (x_2 \lor x_3 \lor x_4) \ldots \)

**LLL**: 3 satisfying assignment.
Constructive LLL (Moser & Tardos 2010)

\[ \mathcal{X} = \{x_1, \ldots, x_m\}: \text{indep. discrete r.v.'s} \]

\[ \mathcal{A} = \{A_1, \ldots, A_n\}: \text{events} \]

\[ \forall \mathcal{A}_i \subseteq \mathcal{X} \]

\[ A_i \text{ and } A_j \text{ are dep. if } \mathcal{A}(A_i) \cap \mathcal{A}(A_j) \neq \emptyset. \]

**Dep graph** \[ G_{\mathcal{A}}: (\mathcal{A}, E) \]

\[ E = \{ (A_i, A_j) | \mathcal{A}(A_i) \cap \mathcal{A}(A_j) \neq \emptyset \} \]

\[ \max \text{-deg} \leq d \]

\[ \Pr[A_i] \leq P \]

\[ e \cdot P(d+1) < 1-\varepsilon \]

\[ \Rightarrow \text{By LLL, we know a construction exists, but how do we find it?} \]

**MT Algorithm**

1) Initialize \( X_i \)'s with random assignment.

2) While \( \exists \text{bad event} \): Pick one bad event (arbitrarily) and resample all its variables.

**Witness tree**

**Execution**: \( A_{i_1}, A_{i_2}, \ldots \mid \ldots, A_{i_k} \)

**Witness tree** \( T_j \): (\( j \)'th resampling step)

Root \( A_{i_j} \)

For every \( A \in \{A_{i_1}, \ldots, A_{i_k}\} \):

Add \( A \) as the child of its deepest neighbor in \( T_j \).
Lemma: \( \forall i < j \) \( T_i \neq T_j \).

\[ \text{Pf: If the two trees have the same root then } T_i \text{ must be a (strict) sub-tree of } T_j. \]

Fix tree \( T \). \( |T| = s \).

What is the probability that \( T \) occurs in \( MT \) execution?

**Lemma:** \( \Pr [T \text{ occurs}] \leq p^s \).

**Pf:**

**Obs:** If \( A_i \) and \( A_j \) are in the same level, then they are independent.

\[ \Rightarrow \Pr [L_k] \leq p^{1_L} \]

The fact that \( A_j \) is a descendent of \( A_i \) does not increase the probability of \( A_i \), because we use fresh randomness.

\[ \Rightarrow \Pr [L_i \mid L_k, \ldots, L_{i-1}] = \Pr [L_i] \leq p^{1_L}. \]
Lemma: \( E[\text{# trees occurring in MT execution}] = O(n) \)

\[ \text{Pf: } T_s: \text{ collection of all trees of size } s. \]

\[
\text{# trees of size } s \leq n \cdot (S(d+1))^{S-1}
\]

Can represent each tree by a binary string of length \( S(d+1) \) with exactly \( S-1 \) 1 bits.

\[
\sum_{S=1}^{\infty} n \cdot (S(d+1))^{S-1} \cdot P^S
\]

\[
= n \sum_{S=1}^{\infty} (S(d+1)e)^S \cdot P^S = O(n)
\]

\( \square \)

Thm: W.h.p. there is no tree of size \( S = c \cdot \log_{1/\epsilon} n \)

\[ \text{Pf: } E[\text{# trees of size } S] = \left| T_s \right| \cdot P^S \leq (eP(d+1))^S = \frac{1}{n^c} \]

\( \square \)
Distributed LLL

Communication graph: dep. graph

\[ \mathcal{A} = \{A_1, \ldots, A_n\} \quad \mathcal{X} = \{X_1, \ldots, X_m\} \]

Node \( v_i \): event \( A_i \)

\[ \Pr[A_i] \leq p \]

LLL: \( p \cdot e \cdot (d+1) \leq 1 - \epsilon \)

Goal: Find good assignment.

Example: Defective Coloring

* Graph \( G = (V, E) \), \( \max\text{-deg} = \Delta \), \( f \geq 60 \ln \Delta \)

* Vertex has \( \lceil \frac{2\Delta}{f} \rceil \) colors.

Goal: each vertex has at most \( f \) neighbors with the same color.

- bad event: \( > f \) neighbors with same color.

- Dep. graph: vertices that share a neighbor (i.e., distance = 2).

Thm 1: there is a distributed LLL algorithm that runs in

\[ O(T_{\text{MIS}} \cdot \log \frac{1}{\epsilon^{\alpha(n)}} n) \] rounds, w.h.p.

Thm 2: [Chung, Pettie, Su]: LLL in \( O(\log \frac{1}{\epsilon^{\alpha(n)}} n) \).
Distributed LLL

1) Initialize with random assignment.

2) While bad event:
   \[ B := \text{bad event} \]
   \[ M = \text{MIS}(G[B]) \]
   Resample all vars in M.

Want to show: Step 2 repeats \(O(\log \frac{\log \alpha}{\log \beta} n)\) iterations.

**Lemma:** W.h.p. \(O(\log \frac{\log \alpha}{\log \beta})\) resampling phases.

\[
\begin{align*}
M_1, M_2, \ldots, M_k \\
\overline{y_1, \ldots, y_k}
\end{align*}
\]

**Obs:** \(u \in M_i\) it holds that \(T_u\) has depth \(\geq i-1\).

**Pf:** \(i=1\): easy.

**step:** \(N^+(u) \cap M_{i-1} \neq \emptyset\), why? \(\rightarrow\)

- If \(u\) is bad in \(i-1\) \(\checkmark\)
- If \(u\) is good in \(i-1\), we must have resampled a neighbor.

\(\Rightarrow\) we are done by induction.

The obs completes the proof of the lemma (using arguments from centralized alg).
Improved Distributed LLL

1) Initialize with random assignment.

2) While bad event:

   \( B := \text{bad event} \)

   \( I = \{ u \mid u \in B, \text{ID}(u) < \text{ID}(v), \forall v \in N(u) \cap B \} \)

   Resample all vars in \( I \).

Remark: I can be much smaller than MIS. Consider the path graph.

\[ 1 \rightarrow 2 \rightarrow 3 \rightarrow \ldots \rightarrow n \]

2-Witness tree \( I_1, \ldots, I_l \)

\[ \text{step } j: \]

Root \( A_{i,j} \)

For every \( A \in \{ A_{i,j-1}, \ldots, A_{1,1} \} \):

Connect \( A \) to its deepest node in \( N^+u \) in the tree.

- 2-Witness tree of size \( S \): \( p^S \)
- # trees of size \( S \): \( n \cdot \left( \frac{S(d^2 + 1)}{S-1} \right) \approx n \cdot (e(d^2 + 1))^S \)
- w.h.p. no 2-witness tree of size \( S = O \left( \log \frac{1}{p e d^2} n \right) \)
Lemma: \( u \in I_j \) has 2-witness tree of depth \( \geq j - 1 \)

Pf: \( N^{+2}(u) \cap I_{j-1} \)

* \( u \) is good in \( j-1 \)

* \( u \) is bad in \( j-1 \)

\( u \in I_{j-1} \)

\( v < u \), \( v \) is bad \( v \in N(u) \)

\( v \in I_{j-1} \)

\( v \) is good in \( j \) but bad in \( j-1 \)

\( \Rightarrow N^{+2}(u) \cap I_{j-1} \neq \emptyset \)

\( \square \)