Randomized Algorithms

The Lovász Local Lemma

Joseph Chuang-Chieh Lin

Department of Computer Science & Engineering,
National Taiwan Ocean University

Spring 2026

Outline

1. The Lovász Local Lemma
2. Explicit Constructions Using the Local Lemma
3. Lovász Local Lemma: The General Case

Outline

1 The Lovász Local Lemma

2 Explicit Constructions Using the Local Lemma

3 Lovász Local Lemma: The General Case

Introduction

- One of the most elegant and useful tools in applying the probabilistic method is the *Lovász Local Lemma* (LLL).
- Let $E_1, E_2, \ldots, E_n$ be a set of **bad** events, we want to show that there is an event (or element in the sample space) that is NOT included in any of the bad events.

Mutually Independent (Recall)

Events $E_1, E_2, \ldots, E_n$ are mutually independent iff for any subset $I \subseteq [1, n]$,

$$\Pr\left[\bigcap_{i \in I} E_i\right] = \prod_{i \in I} \Pr[E_i].$$

- If $E_1, \ldots, E_n$ are mutually independent then so are $\overline{E}_1, \ldots, \overline{E}_n$. (Exercise)

Mutually Independent (Recall)

Events $E_1, E_2, \ldots, E_n$ are mutually independent iff for any subset $I \subseteq [1, n]$,

$$\Pr\left[\bigcap_{i \in I} E_i\right] = \prod_{i \in I} \Pr[E_i].$$

- If $E_1, \ldots, E_n$ are mutually independent then so are $\overline{E}_1, \ldots, \overline{E}_n$. (Exercise)
- If $\Pr(E_i) < 1$ for all i, then

$$\Pr\left[\bigcap_{i=1}^{n} \overline{E}_i\right] = \prod_{i=1}^{n} \Pr(\overline{E}_i) > 0.$$

Mutually Independent (Recall)

Events $E_1, E_2, \ldots, E_n$ are mutually independent iff for any subset $I \subseteq [1, n]$,

$$\Pr\left[\bigcap_{i \in I} E_i\right] = \prod_{i \in I} \Pr[E_i].$$

- If $E_1, \ldots, E_n$ are mutually independent then so are $\overline{E}_1, \ldots, \overline{E}_n$. (Exercise)
- If $\Pr(E_i) < 1$ for all i, then

$$\Pr\left[\bigcap_{i=1}^{n} \overline{E}_i\right] = \prod_{i=1}^{n} \Pr(\overline{E}_i) > 0.$$

- However, mutual independence is too much to ask for in many arguments.

LLL Comes to the Rescue

- The Lovász local lemma generalizes to the case where the n events are not mutually independent but the dependency is **limited**.

LLL Comes to the Rescue

- The Lovász local lemma generalizes to the case where the n events are not mutually independent but the dependency is **limited**.
- Specifically, we say that an event E_{n+1} is mutually independent of the events $E_1, E_2, \ldots, E_n$ if, for any subset $I \subseteq [1, n]$,

$$\Pr\left[E_{n+1} \,\middle|\, \bigcap_{j \in I} E_j\right] = \Pr[E_{n+1}]$$

Dependency graph

Dependency graph

A dependency graph for a set of events $E_1, \ldots, E_n$ is a graph $G = (V, E)$ such that $V = \{1, 2, \ldots, n\}$ and for $i = 1, 2, \ldots, n$, event E_i is mutually independent of the events $\{E_j \mid (i, j) \notin E\}$.

- We discuss first a special case, the symmetric version of the Lovász local lemma, which is more intuitive and sufficient for most algorithmic applications.

Lovász local lemma

Theorem [Lovász Local Lemma]

Let $E_1, E_2, \ldots, E_n$ be a set of events, And assume that the following hold:

1. for all i , $\Pr[E_i] \le p$;
2. the degree of the dependency graph given by $E_1, E_2, \ldots, E_n$ is bounded by d;
3. $4dp \le 1$.

Then

$$\Pr\left[\bigcap_{i=1}^{n} \overline{E}_i\right] > 0.$$

Lovász local lemma

Theorem [Lovász Local Lemma]

Let $E_1, E_2, \ldots, E_n$ be a set of events, And assume that the following hold:

1. for all i , $\Pr[E_i] \leq p$;
2. the degree of the dependency graph given by $E_1, E_2, \ldots, E_n$ is bounded by d;
3. $4dp \leq 1$.

Then

$$\Pr\left[\bigcap_{i=1}^{n} \overline{E}_i\right] > 0.$$

- Note that $p < 1$ for $d > 0$ (for $d = 0$ we refer to the mutually indepent case).

Proof of LLL (1/9)

- Let $S \subset \{1, 2, \dots, n\}$. We prove by induction on $s = 0, \dots, n-1$ that, if $|S| \leq s$, then for all $k \notin S$ we have

$$\Pr\left(E_k \;\middle|\; \bigcap_{j \in S} \overline{E}_j\right) \leq 2p.$$

Proof of LLL (1/9)

- Let $S \subset \{1, 2, \ldots, n\}$. We prove by induction on $s = 0, \ldots, n-1$ that, if $|S| \leq s$, then for all $k \notin S$ we have

$$\Pr\left(E_k \;\middle|\; \bigcap_{j \in S} \overline{E}_j\right) \leq 2p.$$

- For this expression to be well-defined when $S \neq \emptyset$, we need

$$\Pr\left[\bigcap_{j \in S} \overline{E}_j\right] > 0.$$

Proof of LLL (1/9)

- Let $S \subset \{1, 2, \ldots, n\}$. We prove by induction on $s = 0, \ldots, n-1$ that, if $|S| \leq s$, then for all $k \notin S$ we have

$$\Pr\left(E_k \,\middle|\, \bigcap_{j \in S} \overline{E}_j\right) \leq 2p.$$

- For this expression to be well-defined when $S \neq \emptyset$, we need

$$\Pr\left[\bigcap_{j \in S} \overline{E}_j\right] > 0.$$

*Note that $\Pr[A \mid B] = \Pr[A \cap B] / \Pr[B]$.

Proof of LLL (2/9)

- The base case $s = 0$ follows from the assumption that $\Pr[E_k] \le p$.
- The inductive step:
  - First, show that $\Pr\left[\bigcap_{j\in S} \overline{E}_j\right] > 0$.
    This is true when $s = 1$, because $\Pr[\overline{E}_j] \ge 1 - p > 0$.
    For $s > 1$, WLOG, $S = \{1, 2, \ldots, s\}$. Then

$$\begin{aligned}
\Pr\left[\bigcap_{i=1}^{s} \overline{E}_i\right] &= \prod_{i=1}^{s} \Pr\left[\overline{E}_i \;\middle|\; \bigcap_{j=1}^{i-1} \overline{E}_j\right] \\
&= \prod_{i=1}^{s} \left(1 - \Pr\left[E_i \;\middle|\; \bigcap_{j=1}^{i-1} \overline{E} j\right]\right) \\
&\ge \prod_{i=1}^{s} (1 - 2p) > 0.
\end{aligned}$$

Proof of LLL (2/9)

- The base case $s = 0$ follows from the assumption that $\Pr[E_k] \le p$.
- The inductive step:
  - First, show that $\Pr\left[\bigcap_{j \in S} \overline{E}_j\right] > 0$.

    This is true when $s = 1$, because $\Pr[\overline{E}_j] \ge 1 - p > 0$.
    For $s > 1$, WLOG, $S = \{1, 2, \ldots, s\}$. Then $(\because 4dp \le 1)$

$$
\begin{aligned}
\Pr\left[\bigcap_{i=1}^{s} \overline{E}_i\right] &= \prod_{i=1}^{s} \Pr\left[\overline{E}_i \,\middle|\, \bigcap_{j=1}^{i-1} \overline{E}_j\right] \\
&= \prod_{i=1}^{s} \left(1 - \Pr\left[E_i \,\middle|\, \bigcap_{j=1}^{i-1} \overline{E} j\right]\right) \\
&\ge \prod_{i=1}^{s} (1 - 2p) > 0.
\end{aligned}
$$

Proof of LLL (3/9)

- For the rest of the induction, let $S_1 = \{j \in S \mid (k, j) \in E\}$ and $S_2 = S \setminus S_1$.
  - If $S_2 = S$ then E_k is mutually independent of the events $\overline{E}_i, i \in S$, and

$$\Pr\left[E_k \;\middle|\; \bigcap_{j \in S} \overline{E}_j\right] = \Pr[E_k] \leq p.$$

Proof of LLL (3/9)

- For the rest of the induction, let $S_1 = \{j \in S \mid (k, j) \in E\}$ and $S_2 = S \setminus S_1$.
  - If $S_2 = S$ then E_k is mutually independent of the events $\overline{E}_i, i \in S$, and

$$\Pr\left[E_k \;\middle|\; \bigcap_{j \in S} \overline{E}_j\right] = \Pr[E_k] \leq p.$$

- We continue with the case $|S_2| < s$.

Proof of LLL (4/9)

- It will be helpful to introduce the following notation.
- Let F_S be defined by
$$F_S = \bigcap_{j \in S} \overline{E_j},$$
And similarly define F_{S_1} and F_{S_2}. Notice that $F_S = F_{S_1} \cap F_{S_2}$.

Proof of LLL (4/9)

- It will be helpful to introduce the following notation.
- Let F_S be defined by

$$F_S = \bigcap_{j \in S} \overline{E_j},$$

And similarly define F_{S_1} and F_{S_2}. Notice that $F_S = F_{S_1} \cap F_{S_2}$.

- By the definition of conditional probability,

$$\Pr[E_k \mid F_S] = \frac{\Pr[E_k \cap F_S]}{\Pr[F_S]}. \quad (*)$$

Proof of LLL (6/9)

$$\Pr[E_k \mid F_S] = \frac{\Pr[E_k \cap F_S]}{\Pr[F_S]}. \quad (*)$$

- Applying the definition of conditional probability to (*), we obtain

$$\Pr[E_k \cap F_S] = \Pr[E_k \cap F_{S_1} \cap F_{S_2}] = \Pr[E_k \cap F_{S_1} \mid F_{S_2}]\Pr[F_{S_2}].$$

- The denominator can be written as

$$\Pr[F_S] = \Pr[F_{S_1} \cap F_{S_2}] = \Pr[F_{S_1} \mid F_{S_2}]\Pr[F_{S_2}].$$

Proof of LLL (6/9)

$$\Pr[E_k \mid F_S] = \frac{\Pr[E_k \cap F_S]}{\Pr[F_S]}. \quad (*)$$

- Applying the definition of conditional probability to (*), we obtain

$$\Pr[E_k \cap F_S] = \Pr[E_k \cap F_{S_1} \cap F_{S_2}] = \Pr[E_k \cap F_{S_1} \mid F_{S_2}]\Pr[F_{S_2}].$$

- The denominator can be written as

$$\Pr[F_S] = \Pr[F_{S_1} \cap F_{S_2}] = \Pr[F_{S_1} \mid F_{S_2}]\Pr[F_{S_2}].$$

- Canceling the common factor (nonzero), yields

$$\Pr[E_k \mid F_S] = \frac{\Pr[E_k \cap F_{S_1} \mid F_{S_2}]}{\Pr[F_{S_1} \mid F_{S_2}]}. \quad (**)$$

Proof of LLL (7/9)

- Since the probability of an intersection of events is bounded by the probability of any one of the events and since E_k is independent of the events in S_2, we can bound the numerator of (6.5) by

$$\Pr[E_k \cap F_{S_1} \mid F_{S_2}] \leq \Pr[E_k \mid F_{S_2}] = \Pr[E_k] \leq p.$$

Because $|S_2| < |S| = s$, we can apply the induction hypothesis to

$$\Pr[E_i \mid F_{S_2}] = \Pr\left[E_i \,\middle|\, \bigcap_{j \in S_2} \overline{E_j}\right].$$

Proof of LLL (8/9)

- Using also the fact that $|S_1| \leq d$, we estabilish a lower bound on the denominator of (**) as follows:

$$
\begin{aligned}
\Pr[F_{S_1} \mid F_{S_2}] &= \Pr\left[\bigcap_{i \in S_1} \bar{E}_i \,\middle|\, \bigcap_{j \in S_2} \bar{E}_j\right] \\
&\geq 1 - \sum_{i \in S_1} \Pr\left[E_j \,\middle|\, \bigcap_{j \in S_2} \bar{E}_j\right] \\
&\geq 1 - \sum_{i \in S_1} 2p \\
&\geq 1 - 2pd \\
&\geq \frac{1}{2}.
\end{aligned}
$$

Proof of LLL (9/9)

- Using the upper bound for the numerator and the lower bound for the denominator, we prove the induciotn:
$$\Pr[E_k \mid F_S] = \frac{\Pr[E_k \cap F_{S_1} \mid F_{S_2}]}{\Pr[F_{S_1} \mid F_{S_2}]} \le \frac{p}{1/2} = 2p.$$

Proof of LLL (9/9)

- Using the upper bound for the numerator and the lower bound for the denominator, we prove the induciotn:

$$\Pr[E_k \mid F_S] = \frac{\Pr[E_k \cap F_{S_1} \mid F_{S_2}]}{\Pr[F_{S_1} \mid F_{S_2}]} \leq \frac{p}{1/2} = 2p.$$

- The theorem follows from

$$\begin{aligned}
\Pr\left[\bigcap_{i=1}^{n} \bar{E}_i\right] &= \prod_{i=1}^{n} \Pr\left[\bar{E}_i \,\middle|\, \bigcap_{j=1}^{i-1} \overline{E_j}\right] \\
&= \prod_{i=1}^{n} \left(1 - \Pr\left[E_i \,\middle|\, \bigcap_{j=1}^{i-1} \overline{E_j}\right]\right) \\
&\geq \prod_{i=1}^{n} (1 - 2p) > 0.
\end{aligned}$$

Application: Edge-Disjoint Paths

- Assume that n pairs of users need to communicate using edge-disjoint paths on a given network.
- Each pair $i = 1, 2, \ldots, n$ can choose a path from a collection F_i of m paths.
- **Goal:** Apply LLL to show that, if the possible paths do not share too many edges, then there is a way to choose n edge-disjoint paths connecting the n pairs.

Theorem 1

If any path in F_i shares edges with no more than k paths in F_j, where $i \neq j$ and $8nk/m \leq 1$, than there is a way to choose n edge-disjoint paths connecting the n pairs.

Proof of Theorem 1 (1/2)

- Consider the probability space defined by each pair choosing a path independently and uniformly at random from its set of m paths.
- $E_{i,j}$: the event that the paths chosen by pairs i and j **share at least one edge**.

Proof of Theorem 1 (1/2)

- Consider the probability space defined by each pair choosing a path independently and uniformly at random from its set of m paths.
- $E_{i,j}$: the event that the paths chosen by pairs i and j **share at least one edge**.
- Since a path in F_i shares edges with no more than k paths in F_j,

$$p = \Pr[E_{i,j}] \leq \frac{k}{m}.$$

Proof of Theorem 1 (2/2)

- Let d be the degree of the dependency graph. Since event $E_{i,j}$ is independent of all events $E_{i'j'}$ when $i' \notin \{i, j\}$, we have $d < 2n$. Since

$$4dp < \frac{8nk}{m} \leq 1,$$

all of the conditions of the LLL are satisfied, proving

$$\Pr\left[\bigcap_{i \neq j} \overline{E}_{i,j}\right] > 0.$$

Hence, there is a choice of paths such that the n paths are edge disjoint.

Application: Satisfiability

Theorem 2

If no variable in a k-SAT formula appears in more than $T = 2^k/4k$ clauses, then the formula has a satisfying assignment.

Proof:

- Consider the probability space defined by the event that "the ith clause is not satisfied by the random assignment."
- Define E_i: the event that the ith clause is not satisfied by the random assignment. Since each clause has k literals,

$$\Pr[E_i] = 2^{-k}.$$

Proof of Theorem 2 (2/3)

- The event E_i is mutually independent of all of the events related to clauses that do not share variables with clause i.
- Because each of the k variables in clause i can appear in no more than $T = 2^k/4k$ clauses, the degree of the dependency graph is bounded by $d \leq kT \leq 2^{k-2}$.

Proof of Theorem 2 (2/3)

- The event E_i is mutually independent of all of the events related to clauses that do not share variables with clause i.
- Because each of the k variables in clause i can appear in no more than $T = 2^k/4k$ clauses, the degree of the dependency graph is bounded by $d \leq kT \leq 2^{k-2}$.
- In this case,

$$4dp \leq 4 \cdot 2^{k-2} 2^{-k} \leq 1.$$

Proof of Theorem 2 (3/3)

- So, we can apply the LLL to conclude that

$$\Pr\left(\bigcap_{i=1}^{m} \bar{E}_i\right) > 0;$$

hence there is a satisfying assignment for the formula.

Outline

1. The Lovász Local Lemma
2. Explicit Constructions Using the Local Lemma
3. Lovász Local Lemma: The General Case

The issue of LLL

- The Lovász Local Lemma proves that a random element in an appropriately defined sample space has a nonzero probability of satisfying our requirement.
- However, this probability might be too small for an algorithm that is based on simple sampling.
- The number of objects that we need to sample before we find an element that satisfies our requirements might be exponential in the problem size.

- The Lovász Local Lemma can be used to derive efficient construction algorithms.

- The Lovász Local Lemma can be used to derive efficient construction algorithms.

Common two phases (break the problem into small subproblems)

1. a subset of the variables of the problem are assigned random values;
   - the random partial solution fixed in the first phase can be extended to a full solution of the problem.
   - the dependency graph H between events defined by the variables deferred to the second phase has, w.h.p., only small connected components.
2. the remaining variables are deferred to the second stage.

- The Lovász Local Lemma can be used to derive efficient construction algorithms.

Common two phases (break the problem into small subproblems)

1. a subset of the variables of the problem are assigned random values;
   - the random partial solution fixed in the first phase can be extended to a full solution of the problem.
   - the dependency graph H between events defined by the variables deferred to the second phase has, w.h.p., only small connected components.
2. the remaining variables are deferred to the second stage.

- Let's see some examples.

Example: A Satisfiability Algorithm (k-SAT)

- **Goal:** Design a polynomial time algorithm for k-SAT when k is a constant.

Example: A Satisfiability Algorithm (k-SAT)

- **Goal:** Design a polynomial time algorithm for k-SAT when k is a constant.
  - Note: k-SAT is NP-complete for $k \geq 3$.

Input Setting

- Consider a k-SAT formula $\mathcal{F}$, k is an even constant, such that each variable appears in no more than $T = 2^{\alpha k}$ clauses for some constant $\alpha > 0$.
- $x_1, x_2, \ldots, x_\ell$: the ℓ variables; $C_1, C_2, \ldots, C_m$: the m clauses of $\mathcal{F}$.

Example: A Satisfiability Algorithm (k-SAT)

- **Goal:** Design a polynomial time algorithm for k-SAT when k is a constant.
  - Note: k-SAT is NP-complete for $k \geq 3$.

Input Setting

- Consider a k-SAT formula $\mathcal{F}$, k is an even constant, such that each variable appears in no more than $T = 2^{\alpha k}$ clauses for some constant $\alpha > 0$.
- $x_1, x_2, \ldots, x_\ell$: the ℓ variables; $C_1, C_2, \ldots, C_m$: the m clauses of $\mathcal{F}$.

Dangerous clause

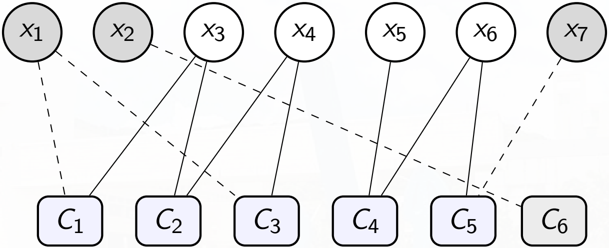
A clause C_i is dangerous if both of the following conditions hold:

- $k/2$ literals of the clause C_i have been fixed;
- C_i is not yet satisfied.

The Two Phases

- **Phase I:** Consider the variables $x_1, \ldots, x_\ell$ sequentially. If x_i is not in a dangerous clause, assign it independently and uniformly at random a value in $\{0, 1\}$.
  - *Surviving clause*: a clause which is not satisfied by variables ($< k/2$) fixed in Phase I.
  - *Deferred variable*: not assigned a value in Phase I.
- **Phase II:** Exhaustive search to assign values to deferred variables.

From partial assignment to surviving clauses



$C_1 = (x_1 \vee x_3)$, $C_2 = (\neg x_3 \vee x_4)$, $C_3 = (x_1 \vee \neg x_4)$,
$C_4 = (x_5 \vee x_6)$, $C_5 = (\neg x_6 \vee x_7)$, $C_6 = (x_2)$.

$$F = C_1 \wedge C_2 \wedge C_3 \wedge C_4 \wedge C_5 \wedge C_6.$$

- x_1, x_2, x_7: fixed; x_3, x_4, x_5, x_6: deferred.
- C_6 is satisfied by x_2 so disappears in H'.

The Lemma

- The partial solution computed in Phase I can be extended to a full satisfying assignment of $\mathcal{F}$.
- with high probability, the exhaustive search in Phase II is completed in time polynomial in m.

Lemma 1

There is an assignment of values to the deferred variables such that all the surviving clauses are satisfied.

Proof of Lemma 1 (1/2)

- Let $H = (V, E)$ be a graph on m nodes, where $V = \{1, 2, \ldots, m\}$ and let $(i, j) \in E$ if and only if $C_i \cap C_j \neq \emptyset$ (dependency graph).
- Let $H' = (V', E')$ be a subgraph of H such that
  (a) $i \in V'$ iff C_i is a surviving clause. and
  (b) $(i, j) \in E'$ iff C_i and C_j share a deferred variable.

Proof of Lemma 1 (1/2)

- Let $H = (V, E)$ be a graph on m nodes, where $V = \{1, 2, \ldots, m\}$ and let $(i, j) \in E$ if and only if $C_i \cap C_j \neq \emptyset$ (dependency graph).
- Let $H' = (V', E')$ be a subgraph of H such that
  (a) $i \in V'$ iff C_i is a surviving clause. and
  (b) $(i, j) \in E'$ iff C_i and C_j share a deferred variable.
- Consider the probability space defined by assigning a random value in $[0, 1]$ independently to each deferred variable.

Proof of Lemma 1 (1/2)

- Let $H = (V, E)$ be a graph on m nodes, where $V = \{1, 2, \ldots, m\}$ and let $(i, j) \in E$ if and only if $C_i \cap C_j \neq \emptyset$ (dependency graph).
- Let $H' = (V', E')$ be a subgraph of H such that
  (a) $i \in V'$ iff C_i is a surviving clause. and
  (b) $(i, j) \in E'$ iff C_i and C_j share a deferred variable.
- Consider the probability space defined by assigning a random value in $[0, 1]$ independently to each deferred variable.
- Let E_i, for $i = 1, 2, \ldots, m$, be the event that surviving clause C_i is NOT satisfied by this assignment (Phase I + II).

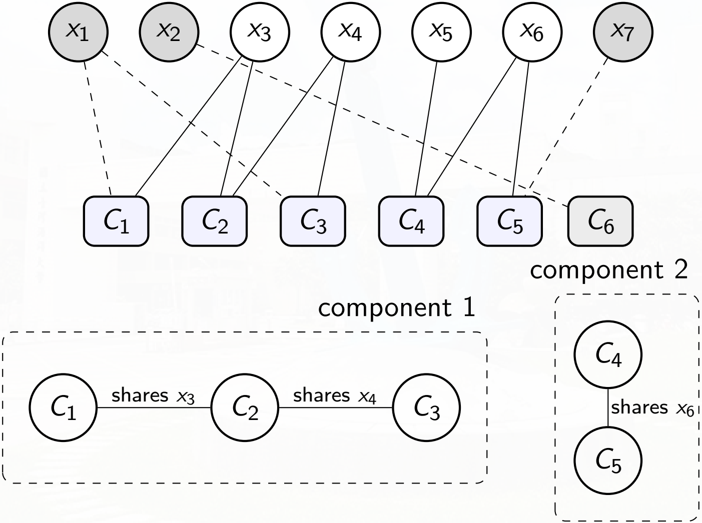
Proof of Lemma 1 (1/2)

- Let $H = (V, E)$ be a graph on m nodes, where $V = \{1, 2, \ldots, m\}$ and let $(i, j) \in E$ if and only if $C_i \cap C_j \neq \emptyset$ (dependency graph).
- Let $H' = (V', E')$ be a subgraph of H such that
  (a) $i \in V'$ iff C_i is a surviving clause. and
  (b) $(i, j) \in E'$ iff C_i and C_j share a deferred variable.
- Consider the probability space defined by assigning a random value in $[0, 1]$ independently to each deferred variable.
- Let E_i, for $i = 1, 2, \ldots, m$, be the event that surviving clause C_i is NOT satisfied by this assignment (Phase I + II).
- Associate E_i with node $i \in V'$. The graph H' is then the dependency graph for this set of events.

Proof of Lemma 1 (2/2)

- A surviving clause has at least $k/2$ deferred variables, so
$$p = \Pr[E_i] \leq 2^{-k/2}.$$
- A variable appears in no more than T clauses; therefore, the degree of the dependency graph is bounded by $d = kT \leq k2^{\alpha k}$.
- For any $k \geq 12$, there is a corresponding suitably small constant $\alpha > 0$ so that
$$4dp = 4k2^{\alpha k}2^{-k/2} \leq 1.$$
- Thus, by the Lovász Local Lemma, there exists an assignment for the deferred variables that (+ the assignment of variables in phase I) satisfies the formula.

Dependency graph and its connected components



Further Observation of the Dependency Graph

- The problem is divided into independent subformulas.
- These subformulas correspond to connected components of the dependency graph H'.

Further Observation of the Dependency Graph

- The problem is divided into independent subformulas.
- These subformulas correspond to connected components of the dependency graph H'.

⋆ What if the size of each connected component is **small**?

Further Observation of the Dependency Graph

- The problem is divided into independent subformulas.
- These subformulas correspond to connected components of the dependency graph H'.

$\star$ What if the size of each connected component is **small**?
$\Rightarrow$ the exhaustive search of all possible assignments can be done efficiently.

Further Observation of the Dependency Graph

- The problem is divided into independent subformulas.
- These subformulas correspond to connected components of the dependency graph H'.

$\star$ What if the size of each connected component is **small**?
$\Rightarrow$ the exhaustive search of all possible assignments can be done efficiently.

Lemma 2

All connected components in H' are of size $O(\log m)$ with probability $1 - o(1)$.

Proof of Lemma 2 (1/6)

- The probability that a given clause survives is the probability that either this clause or at least one of its direct neighbors is dangerous, which is bounded by

$$(d+1)2^{-k/2}, \text{ where } d = kT > 1.$$

Proof of Lemma 2 (1/6)

- The probability that a given clause survives is the probability that either this clause or at least one of its direct neighbors is dangerous, which is bounded by

$$(d+1)2^{-k/2}, \text{ where } d = kT > 1.$$

  - a clause if dangerous with probability $\leq 2^{-k/2}$.
  - union bound.

- We identify a subset K of the vertices in a component R such that the survival of the clauses represented by the vertices in K are **independent events**.

4-Tree of a connected component

4-Tree

A *4-tree* S of a connected component R in H is defined as follows:

1. S is a rooted tree;
2. any two nodes in S are at distance at least 4 in H;
3. there can be an edge in S only between two nodes with distance exactly 4 between them in H;
4. any node of R is either in S or is at distance 3 or less from a node is S.

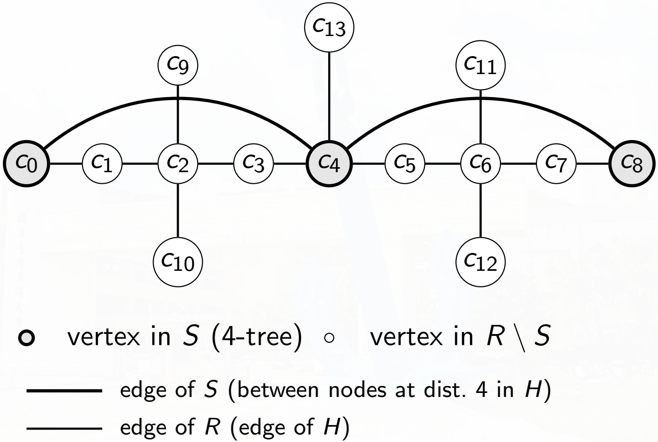


Fig.: A 4-tree $S = \{c_0, c_4, c_8\}$ inside a connected component R of H.

Intuitive Idea(s)

- A node u in a 4-tree survives and the event that another node v in a 4-tree survives are **independent**.
  - Any clause that could cause u to survive has distance ≥ 2 from any clause that could cause v to survive.
  - Clauses at distance 2 share no variables, and hence the events that they are dangerous are independent.

Intuitive Idea(s)

- A node u in a 4-tree survives and the event that another node v in a 4-tree survives are **independent**.
  - Any clause that could cause u to survive has distance ≥ 2 from any clause that could cause v to survive.
  - Clauses at distance 2 share no variables, and hence the events that they are dangerous are independent.
- We can take advantage of this independent to conclude that, for any 4-tree S, the probability that the nodes in the 4-tree survive is

$$\leq ((d+1)2^{-k/2})^{|S|}.$$

Proof of Lemma 2 (3/6)

- A maximal 4-tree S of a connected component R is the 4-tree with the largest possible number of vertices.

Proof of Lemma 2 (3/6)

- A maximal 4-tree S of a connected component R is the 4-tree with the largest possible number of vertices.
- Since the degree of the dependency graph is bounded by d, there are no more than $d + d(d-1) + d(d-1)(d-1) \leq d^3 - 1$ nodes at distance 3 or less from any given vertex.

Proof of Lemma 2 (3/6)

- A maximal 4-tree S of a connected component R is the 4-tree with the largest possible number of vertices.
- Since the degree of the dependency graph is bounded by d, there are no more than $d + d(d-1) + d(d-1)(d-1) \leq d^3 - 1$ nodes at distance 3 or less from any given vertex.
- **Claim:** A maximal 4-tree of R must have $\geq r/d^3$ vertices (note: $r = |V(R)|$).

Proof of Lemma 2 (3/6)

- A maximal 4-tree S of a connected component R is the 4-tree with the largest possible number of vertices.
- Since the degree of the dependency graph is bounded by d, there are no more than $d + d(d-1) + d(d-1)(d-1) \leq d^3 - 1$ nodes at distance 3 or less from any given vertex.
- **Claim:** A maximal 4-tree of R must have $\geq r/d^3$ vertices (note: $r = |V(R)|$).
  - When we consider the vertices of the maximal 4-tree S and all neighbors within distance ≤ 3 of these vertices, we obtain $< r$ vertices.

Proof of Lemma 2 (4/6): Maximality of S

- Hence, there must be a vertex of distance ≥ 4 from all vertices in S.

Proof of Lemma 2 (4/6): Maximality of S

- Hence, there must be a vertex of distance ≥ 4 from all vertices in S.
- If this vertex has distance exactly 4 from some vertex in S, then it can be added to S and thus S is not maximal ($\Rightarrow\!\Leftarrow$).
- If its distance > 4 from all vertices in S, consider any path that brings it closer to S; such a path must eventually pass through a vertex of distance at least 4 from all vertices in S and of distance 4 from some vertex in S ($\Rightarrow\!\Leftarrow$: maximality of S).

Proof of Lemma 2 (5/6): $1 - o(1)$ constraint

- Next, we show that there is no connected component R of size $r \geq c \lg m$ for some constant c in H'.
- We show that we show that there is NO 4-tree of H of size r/d^3 that survives with probability $1 - o(1)$.

Proof of Lemma 2 (5/6): $1 - o(1)$ constraint

- Next, we show that there is no connected component R of size $r \geq c \lg m$ for some constant c in H'.
- We show that we show that there is NO 4-tree of H of size r/d^3 that survives with probability $1 - o(1)$.
- Count the number of 4-trees of size $s = r/d^3$ in H.

Proof of Lemma 2 (5/6): $1 - o(1)$ constraint

- Next, we show that there is no connected component R of size $r \geq c \lg m$ for some constant c in H'.
- We show that we show that there is NO 4-tree of H of size r/d^3 that survives with probability $1 - o(1)$.
- Count the number of 4-trees of size $s = r/d^3$ in H. We can choose the root of the 4-tree in m ways.
- A tree with root v is **uniquely defined** by an Eulerian tour that starts and ends at v and traverses each edge of the tree twice, once in each direction.
- Since an edge of S represents a path of length 4 in H, at each vertex in the 4-tree the Eulerian path can continue in as many as d^4 different ways, and therefore the number of 4-trees of size $s = r/d^3$ in H is bounded by $m(d^4)^{2s} = md^{8r/d^3}$.

Proof of Lemma 2 (6/6)

- The probability that the nodes of each such 4-tree survive in H' is at most.

$$((d+1)2^{-k/2})^{s} = ((d+1)2^{-k/2})^{r/d^3}.$$

Hence the probability that H' has a connected component of size r is bounded by

$$md^{8r/d3}((d+1)2^{-k/2})^{r/d^3} \leq m2^{(rk/d^3)(8\alpha+2\alpha-1/2)} = o(1)$$

for $r \geq c\log_2 m$ and for a suitably large constant c and a sufficiently small constant $\alpha > 0$.

Solving k-SAT in expected polynomial time for small k

Thus, we have the following theorem.

Theorem 2

Consider a k-SAT formula with m clauses, where k is an even constant and each variable appears in up to $2^{\alpha k}$ clauses for a sufficiently small constant $\alpha > 0$. Then there is an algorithm that finds a satisfying assignment for the formula in expected time that is polynomial in m.

Solving k-SAT in expected polynomial time for small k

Thus, we have the following theorem.

Theorem 2

Consider a k-SAT formula with m clauses, where k is an even constant and each variable appears in up to $2^{\alpha k}$ clauses for a sufficiently small constant $\alpha > 0$. Then there is an algorithm that finds a satisfying assignment for the formula in expected time that is polynomial in m.

Proof:

- If the Phase I partitions the problem into subformulas involving only $O(k \log m)$ variables (w.p. $1 - o(1)$), then a solution can be found by solving each subformula exhaustively in time polynomial in m.

Solving k-SAT in expected polynomial time for small k

Thus, we have the following theorem.

Theorem 2

Consider a k-SAT formula with m clauses, where k is an even constant and each variable appears in up to $2^{\alpha k}$ clauses for a sufficiently small constant $\alpha > 0$. Then there is an algorithm that finds a satisfying assignment for the formula in expected time that is polynomial in m.

Proof:

- If the Phase I partitions the problem into subformulas involving only $O(k \log m)$ variables (w.p. $1 - o(1)$), then a solution can be found by solving each subformula exhaustively in time polynomial in m.
- Thus, we need only run phase I a constant number of times on average before obtaining a good partition.

Outline

1. The Lovász Local Lemma
2. Explicit Constructions Using the Local Lemma
3. Lovász Local Lemma: The General Case

The General LLL

For completeness we provide the general case of LLL below.

Theorem 2 [Lovász Local Lemma (The General Case)]

Let $E_1, E_2, \ldots, E_n$ be a set of events in an arbitrary probability space, and let $G = (V, E)$ be the dependency graph for these events. Assume there exist $x_1, x_2, \ldots, x_n \in [0, 1]$ such that, for all $1 \leq i \leq n$,

$$\Pr[E_i] \leq x_i \prod_{(i,j) \in E} (1 - x_j).$$

Then,

$$\Pr\left[\bigcap_{i=1}^{n} \overline{E}_i\right] \geq \prod_{i=1}^{n} (1 - x_i).$$

Discussions