Randomness in Computing

CS 537

LECTURE 22

Last time

- Probabilistic method
  - Conditional Expectation Inequality
  - Lovasz Local Lemma

Today

- Algorithmic Lovasz Local Lemma

Exercise

Consider an algorithm $\mathcal{A}$ for problem $\boldsymbol{\mathcal{P}}$ that, on inputs of length n, uses $R(n)$ random bits, runs in time $T(n)$, and produces the correct YES/NO answer for the given input with probability $> 1/2$.

Give a deterministic algorithm for $\boldsymbol{\mathcal{P}}$ and analyze its running time.

The running time of your algorithm is

A. $O\big(T(n)\big)$

B. $O\big(R(n) \cdot T(n)\big)$

C. $2^{O(R(n))} \cdot T(n)$

D. $R(n) \cdot 2^{O(T(n))}$

E. $2^{O(R(n)+T(n))}$

F. Larger than all of the above.

Lovasz Local Lemma (LLL)

- Event E is mutually independent from the events $E_1, \dots, E_n$ if, for any subset $I \subseteq [n]$,
$$\Pr[E \mid \bigcap_{j \in I} E_j] = \Pr[E].$$

- A dependency graph for events $B_1, \dots, B_n$ is a graph with vertex set $[n]$ and edge set E, s.t. $\forall i \in [n]$, event B_i is mutually independent of all events $\{B_j \mid (i,j) \notin E\}$.

Lovasz Local Lemma

Let $B_1, \dots, B_n$ be events over a common sample space s.t.

1. max degree of the dependency graph of $B_1, \dots, B_n$ is at most $\boldsymbol{d-1}$
2. $\forall i \in [n], \Pr[B_i] \leq \boldsymbol{p}$

If $\boldsymbol{epd \leq 1}$ then $\Pr\left[\bigcap_{i \in [n]} \bar{B}_i\right] > 0$

Different meaning of d than in the book (to correspond to algorithmic LLL).

Application of LLL

Theorem

If $e\left(\binom{k}{2}\binom{n-2}{k-2}+1\right)2^{1-\binom{k}{2}} \leq 1$ then edges of K_n can be colored with 2 colors so that there is no monochromatic K_k.

Proof:

Canonical special case of LLL: *k*SAT

- Notation: $n =$ number of variables, $m =$ number of clauses

Observation: If $m < 2^k$, then the formula is satisfiable.

Proof:

- Pick a uniformly random assignment.
- Let B_i be the event that clause i is violated.

Statement of LLL for kSAT

- Dependency graph: Vertices correspond to clauses

  edge (i, j) iff clauses i and j share a variable

  If clause i contains x and clause j contains $\bar{x}$, it counts as sharing a variable.

  $\deg(i) =$ number of clauses sharing a variable with clause i

- Let $d = 1 + \max_i \deg(i)$

Algorithmic Lovasz Local Lemma for kSAT

If $\boldsymbol{d \leq 2^{k-3} = \frac{2^k}{8}}$ for some kCNF formula ϕ, then ϕ is satisfiable.

Moreover, a satisfying assignment can be found in $O(m^2 \log m)$ time with probability at least $1 - 2^{-m}$.

Moser-Tardos Algorithm for LLL

Input: a kCNF formula with clauses $C_1, \ldots, C_m$
on n variables and with $d \leq 2^{k-3}$

Global variable

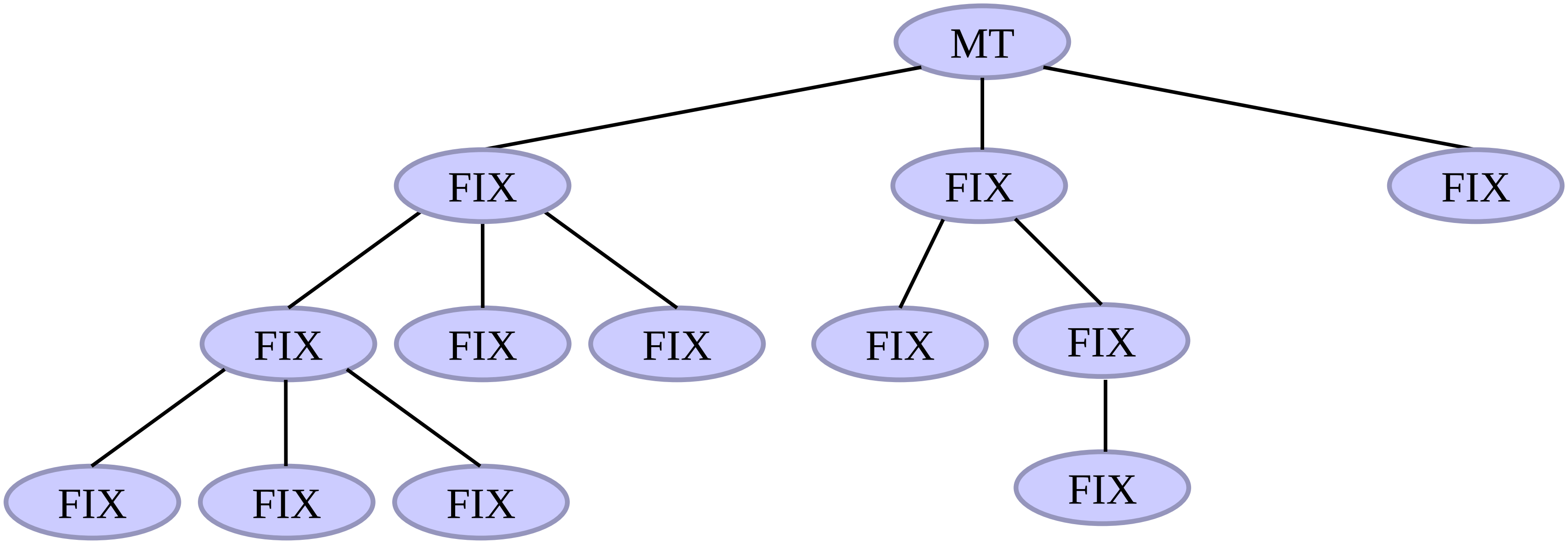
1. Let R be a random assignment where each variable is assigned 0 or 1 uniformly and independently.
2. While some clause C is violated by R, run $\mathsf{FIX}(C)$
3. **Return** R.

$\mathsf{FIX}(C)$

1. Pick new values for k variables in C uniformly and independently and update R.
2. While some clause D that shares a variable with C is violated by R, run $\mathsf{FIX}(D)$

D could be C if we chose the same values as before

Correctness of Moser-Tardos



Observation

If FIX(C) terminates, then it terminates with an assignment in which C and all clauses sharing a variable with C are satisfied.

Correctness of Moser-Tardos

Lemma (Correctness)

A call to FIX that terminates does not change any clauses of the formula from satisfied to violated.

Proof: Suppose for contradiction that some call FIX(C) terminated and changed an assignment to clause D from satisfied to violated, and consider such bad call that terminated first.

- D can't share a variable with C by Observation.
- Then randomly reassigning variables of C does not affect variables of D
- All calls to FIX that the current call made terminated before this call did and, by assumption that this is the first bad call to terminate, could not have spoiled D.

Theorem (Correctness)

If Moser-Tardos terminates, it outputs a satisfying assignment.

Run time of Moser-Tardos

- Assume: $m \geq 2^k$ (o.w. trivial by other means)

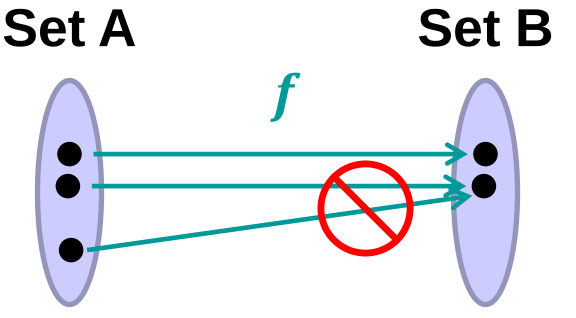
Theorem (Run time)

If $\boldsymbol{d \leq 2^{k-3}}$ then Moser-Tardos terminates after $O(m \log m)$ resampling steps with probability at least $1 - 2^{-m}$.

- Proof idea: Clever way to ``compress'' random bits if the algorithm runs for too long.

Observation 2

If a function $f: A \to B$ is injective (i.e., invertible on its range $f(A)$) then $|B| \geq |A|$.



Function f_T

- Suppose we stop Moser-Tardos after T resampling steps.

Randomness used:

$\boldsymbol{n}$ **bits** for initial assignment

$\boldsymbol{k}$ **bits** for each resampling step

Total: $\boldsymbol{n} + \boldsymbol{Tk}$ **bits**

- Let A be the set of all choices for $n + Tk$ bits

$$f_T\big((x_0, y_0)\big) = (x_T, z_T)$$

initial assignment

Tk bits for reassignment

assignment after T resampling steps

transcript after T resampling steps

Transcript

- Each call to FIX gets recorded as follows:

  If FIX(C) is called by the algorithm

  1 index of the clause C on which FIX is called

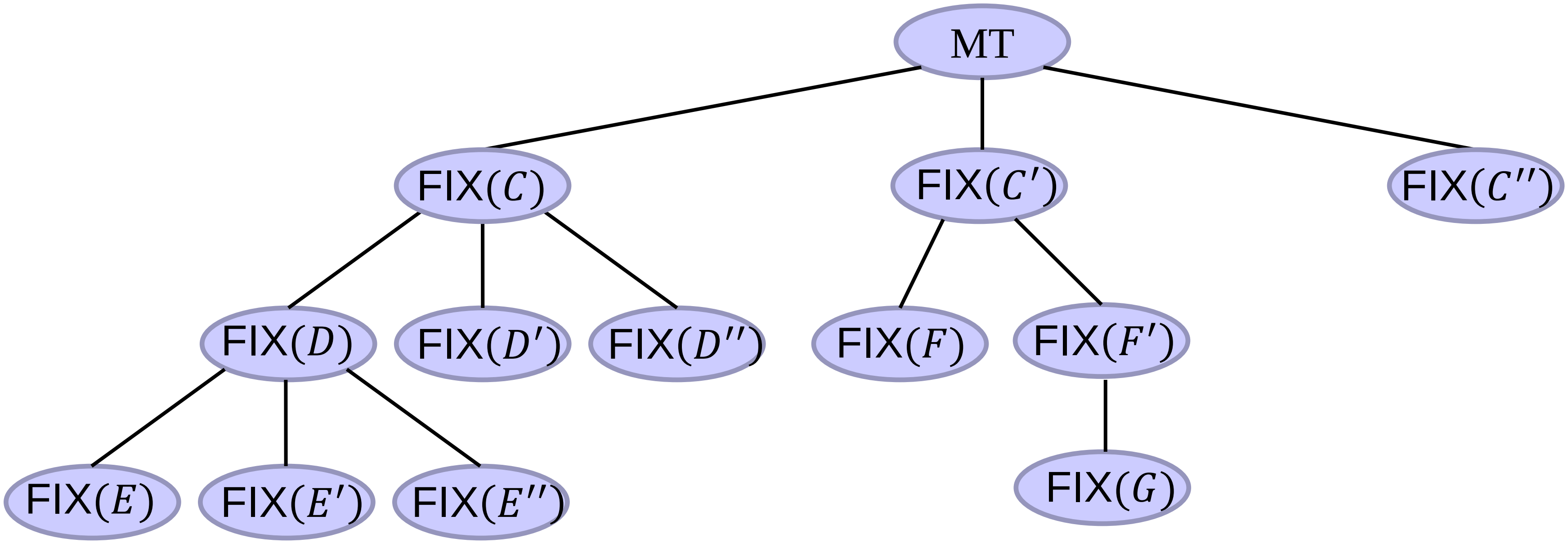
  If FIX(D) is a recursive call made by FIX(C)

  1 ``index'' of the clause D among all clauses that overlap with clause C

- When a call to FIX returns,

  0 is written on the transcript

Transcript: example



Run time of Moser-Tardos

Lemma 1

Function f_T is invertible on all inputs (x_0, y_0) for which Moser-Tardos does not terminate within T steps when run with randomness (x_0, y_0).

Lemma 2

Length of transcript z_T is at most $\boldsymbol{m(\lceil \log_2 m \rceil + 2) + T \cdot (k-1)}$.

Proof of Theorem

First, consider T such that Moser-Tardos never terminates within T resampling steps.

- There is a valid transcript z_T for every choice of the random $n + Tk$ bits needed to run Moser-Tardos

Proof of Theorem (continued)

Now, consider T such that Moser-Tardos fails to terminate w.p. $\geq \frac{1}{2^m}$ within T resampling steps.

- Then f_T is invertible on the set of size $\geq 2^{n+Tk-m}$

Proof of Lemma 1

Lemma 1

Function f_T is invertible on all inputs (x_0, y_0) for which Moser-Tardos does not terminate within T steps when run with randomness (x_0, y_0).

Proof:

- The recursion tree is uniquely defined by z_T
- FIX is only called on violated clauses, and each clause has a unique violating assignment.

Algorithmic LLL for kSAT

Algorithmic Lovasz Local Lemma for kSAT

If $\boldsymbol{d \leq 2^{k-3} = \frac{2^k}{8}}$ for some kCNF formula ϕ, then ϕ is satisfiable.

Moreover, a satisfying assignment can be found in $O(m^2 \log m)$ time with probability at least $1 - 2^{-m}$.