Lecture 7

Sep 24

Prover-Delayer games

$F :=$ unsat CNF formula

Prover Goal: Find a falsified clause
Delayer Goal: Stop the Prover!

Together maintain a "state" S (partial assignment to F)

In each round

- Prover picks a variable x (not in S)
- Delayer responds with $b \in \{0,1\}$, S updated
  $S = S \cup \{x=b\}$
- Prover can then choose to forget a subset of S.

Game ends when S falsifies some clause of F.

"Runs of game" $\equiv$ "Resolution proofs"

Prover strategies $\equiv$ upper bounds (i.e. actual resolution pfs)

Delayer strategies $\equiv$ lower bounds

Thm Consider the modified PD game: where now the Delayer can choose not to respond with a bit and let the Prover pick. Whenever this happens Delayer scores a point.

Sep 24

If there is a strategy for the Delayer where they always score $\geq r$ points then

$$S^T_{Res}(F) \geq 2^r.$$

Pf Let π be a tree-like resolution proof. We use π to make a Prover strategy.

The Prover strategy works from "top-down", by taking a walk starting at $\perp$ and ending at a leaf (i.e. an axiom).

When the Prover is at a clause C in π, it asks Delayer for the value of x, the variable that was resolved on to deduce C.

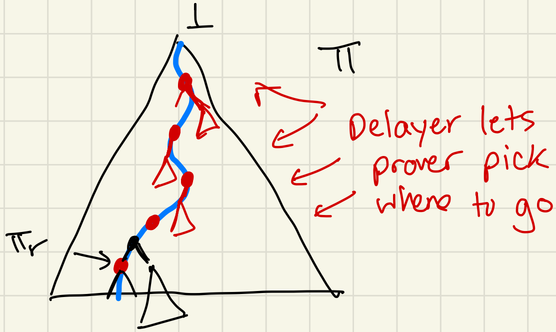
Let $C = A \vee B$, was obtained by resolution step

$$\frac{A \vee x \quad B \vee \bar{x}}{A \vee B = C}$$

If Delayer responds with 1, Prover goes to $B \vee \bar{x}$ (adds x to the state).

———————————— 0, Prover goes to $A \vee x$ (adds $\bar{x}$ to state).

If the Delayer passes (wins a point): then Prover chooses whatever value leads to the smaller subtree.



Let

$\Pi_1, \Pi_2, \ldots, \Pi_r$ be the

r subtrees at these points in time.

$$1 \le |\Pi_r| \le \frac{1}{2}|\Pi_{r-1}| \le \frac{1}{4}|\Pi_{r-2}| \le \cdots \le \frac{1}{2^{r-1}}|\Pi_1| \le \frac{1}{2^r}|\Pi|$$

so $|\Pi| \ge 2^r$. $\square$

<u>Thm</u> Resolution complexity measures can be completely characterized by PD games.

$$D_{Res}(F) := \min\{d \in \mathbb{Z} : \exists \text{ Prover } P \ \forall \text{ Delayers } D \ P \text{ ends the game in } \le d \text{ rounds}\}$$

$$w_{Res}(F) := \min\{w \in \mathbb{Z} : \exists \text{ Prover } P \ \forall \text{ Delayers } D \ P \text{ ends game using states of size } \le w\}$$

$$S_{Res}(F) := \min\{S \in \mathbb{Z} : \exists \text{ Prover } P \text{ and set of } \le S \text{ states } \Pi \text{ s.t. } \forall \text{ Delayer } P \text{ end the game only using states in } \Pi\}$$

Proof Let π be any resolution proof, we design a Prover strategy using π. Sep 24

The Prover strategy works from "top-down", by taking a walk starting at $\perp$ and ending at a leaf (i.e. an axiom).

Along this walk we maintain the following invariant:

Invariant The state S stored by the Prover is exactly $\neg C$ where C is the clause currently visited by the walk.

Initially: $S = \emptyset$, and $C = \perp$, so invariant is satisfied.

When the Prover is at a clause C in π, it asks Delayer for the value of x, the variable that was resolved on to deduce C.

Let $C = A \vee B$, was obtained by resolution step

$$\frac{A \vee x \quad B \vee \bar{x}}{A \vee B = C}$$

If Delayer responds with 1, Prover goes to $B \vee \bar{x}$, x is added to S, and all vars in $A \setminus B$ are forgotten. Now $S = x \wedge \neg B$.

(so invariant is restored)

If Delayer responds with 0, do the symmetric move.

If C is an Axiom of F, then invariant $\Rightarrow$ S falsifies C, so game ends.

The depth, size, width are maintained.

($\Leftarrow$) <u>Exercise</u>

Let P be a prover strategy, Π be states visited by the prover over all possible Delayers.

Show: For every state $S \in \Pi$, there is a resolution + weakening proof of $\neg S$ from F.

(Hint: Structure Π as a DAG, sort in topological order and do induction.)

$\square$

Algorithms for SAT

Let F be a K-CNF formula, not necessarily unsatisfiable

Goal: Solve SAT on F! Either

- Find a satisfying assignment for F, or
- Prove that F is unsatisfiable

Totally naive method: try every assignment to F and evaluate.

Runs in time: $2^n \cdot \text{poly}(|F|)$

all assign

time to evaluate

Exponential Time Hypothesis [Impagliazzo-Paturi 03?]

There is $\alpha > 0$ s.t. 3-SAT can not be solved in $2^{\alpha n}$-time.

Strong Exponential Time Hypothesis [IP 03]

There is $\underline{\underline{\text{no}}}$ $\varepsilon > 0$ s.t. $\forall K$, K-SAT can be solved in

$$(2-\varepsilon)^n$$

time.

ETH $\Rightarrow$ $2^{o(n)}$ time is impossible for 3-SAT

SETH $\Rightarrow$ for SAT (ie no bound on width of clauses) then cannot do better than $2^{n-o(n)}$

<u>Thm</u> [PPZ 98, PPSZ 01, Hertli 10]

For every K there is a (randomized) algorithm (the PPSZ algorithm) that solves K-SAT in time

$$2^{(1-\frac{c_K}{K})n} \qquad c_K > 0, \; c_K \to \frac{\pi^2}{6} \approx 1.44 \text{ as } K \to \infty$$

$$2^{(1-\frac{O(1)}{K})n}$$

The best algs for SAT (theoretically: PPSZ, practice: Conflict-Driven-clause-learning CDCL) have the same underlying algorithm schema:

Proceed from the "top-down": choose variables x and choose assignments to those variables

- if lucky, find a satisfying assignment, done!

- if unlucky, they falsify a clause from F.
  The algorithms backtrack and proceed elsewhere.