Monday, May 14, 2018 7:49 AM

Players take turns selecting an edge out of current node, to any unvisited node. Player loses if there is no valid edge to pick.

Natural complete problems for PSPACE: games.

GEOGRAPHY = { (G, s) : G is an undirected graph & player 1 can win from starting at s }

Theorem: GEOGRAPHY is PSPACE complete.

in PSPACE: $\Phi_i(v_1, .., v_i)$ $s v_1 v_2 \cdots v_i$ is a loosing path.

(i even)

expressible as a poly-size Boolean formula.

$(s, v_1)\ (v_1 v_2) \cdots (v_{i-1}, v_i)$ are all edges.

$\{ s, v_1, v_2, .., v_{i-1} \}$ are all distinct.

$(v_i = s) \vee (v_i = 1) \cdots \vee (v_i = v_{i-1})$ v_i must be a repeat.

$$\exists v_1 \forall v_2 \cdots \forall v_n \bigvee_{i \text{ even}} \Phi_i$$

Next QSAT & GEOG

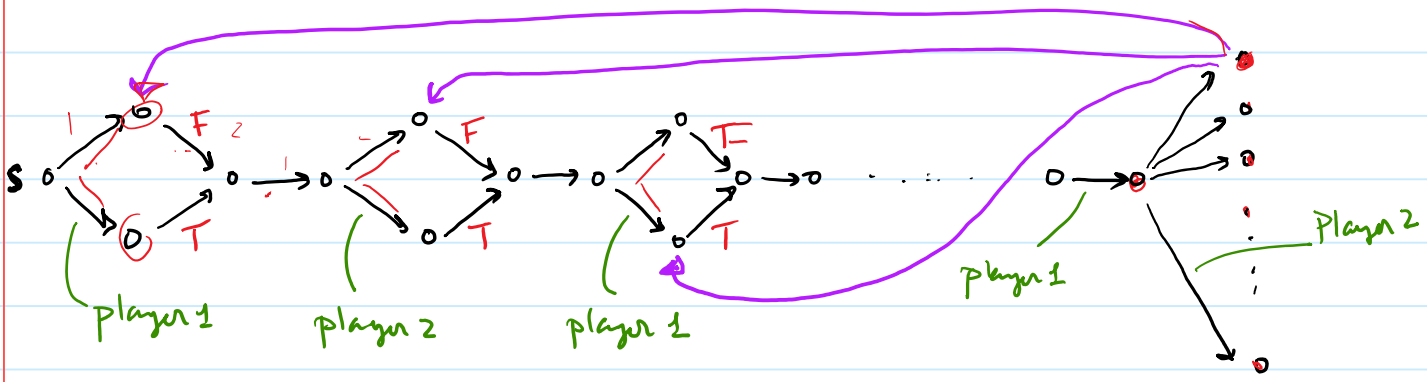
Player 1 trying to make clauses true.
Player 2 trying to make a clause false.

① Boolean Formula $\Rightarrow$ (G, s).

Wed May 16, 2018 - page 2

Monday, May 14, 2018 7:49 AM

Clause $(x_1 \vee x_2 \vee \neg x_3)$



A node for each clause.

Player 2 tries to pick an unsatisfied clause.
Given a clause, Player 1 tries to find a literal inside the chosen clause that is true.

Wednesday, May 16, 2018 8:28 AM

Karp-Lipton theorem.

We know that if P=NP then SAT has poly-sized circuits. What about the converse of that statement?

The converse holds if we use uniform circuit families.

Also, if SAT has poly-size circuits (non-uniform) then the PH collapses to the 2nd level.

Theorem: If $NP \subseteq P/poly \Rightarrow PH = \Sigma_2$.

It suffices to show that

$SAT \in P/poly \Rightarrow \Pi_2 \subseteq \Sigma_2$.

↳ Will show a Π_2-complete problem that can be done in Σ_2.

$\Rightarrow$ Π_2-complete:

$$\{ \phi \mid \forall u \in \{0,1\}^n \ \exists v \in \{0,1\}^n \ \ \phi(u,v) = 1. \}$$

Fixing $u = u^*$ results in an instance of SAT

$$\{ \phi \mid \exists v \in \{0,1\}^n \ \ \phi(u^*, v) = 1 \}$$

$NP \subseteq P/poly \Rightarrow$ there is a poly-sized circuit family that solves SAT.

Wednesday, May 16, 2018 8:28 AM

For every boolean formula ϕ + $u \in \{0,1\}^n$

$C_m(\phi, u) = 1$ iff $\exists v \ \phi(u,v) = 1$.

$m = |(\phi, u)|$

C_m solves the decision problem for SAT. This can be converted to a circuit that finds the solution v if it exists.
If C_m is faulty then the fault will be discovered.

Pick an encoding so that the size of ϕ_n doesn't change if some of the inputs are hard-coded.
$\phi_n(t_1 \cdots t_i \ v_{i+1}, \ldots v_n)$
$t_1 \cdots t_i$ hard-coded to 0 or 1.

use $C_m(\phi, u, t)$

For $i = 1 \ldots n$.
Is $\phi(t_1, \ldots, t_{i-1}, 0, v_{i+1}, \ldots, v_n)$ satisfiable?
Yes $\Rightarrow t_i \leftarrow 0$
No $\Rightarrow t_i \leftarrow 1$.
Output $\phi(t_1, \ldots, t_n)$

$\Rightarrow$ this algorithm can be converted to a circuit C_m'.
If output = 1 then ϕ is definitely satisfiable even if C_m is faulty
If ϕ_n is not satisfiable output = 0 regardless of C_m.
If C_m is not faulty output of C_m' is correct.

Wednesday, May 16, 2018 8:28 AM

If NP $\subseteq$ P/poly then circuit family $\{C'_n\}$ exists.

C_n can be guessed using the $\exists$ quantifier using $q(n)$ bits for some poly $q(n)$.

$\exists\, w \in \{0,1\}^{q(n)} \quad \forall\, u \in \{0,1\}^n \quad C'_n(\phi, u) = 1.$

string w is a description of C_n then that is used to create C'_n.

If $\forall u\, \exists v\, \phi(u,v)$ is TRUE

then there is some w that is the correct circuit C'_n.

for every u, $\exists v\, \phi(u,v) = 1 \iff C'_n(\phi, u) = 1$.

$\exists w\, \forall u\, C'_n(\phi, u) = 1.$ is TRUE.

If $\forall u\, \exists v\, \phi(u,v)$ is FALSE

then $\exists\, u = u^*$ s.t. $\forall v\, \phi(u^*, v) = 0$.

$\phi(u^*, v)$ is not satisfiable.

Regardless of the guess of $C_n \Rightarrow$ C'_n will output 0 on input $\phi(u^*, v)$.

$\exists w\, \forall u\, C'_n(\phi, u) = 1$ is False. $\in \Sigma_2$.

Wednesday, May 16, 2018 8:28 AM

Theorem: $BPP \subseteq \Sigma_2 \cap \Pi_2$

We don't even know if $BPP \neq EXP$. but we expect that $\Sigma_2 \cap \Pi_2$ is much weaker than EXP.

It's enough to show that $BPP \subseteq \Sigma_2$ since BPP is closed under complement.

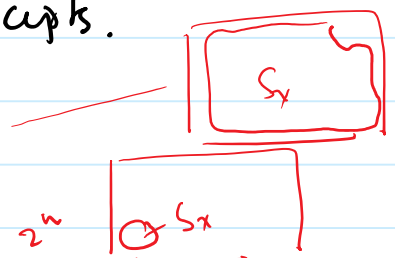
$$L \in BPP \Rightarrow \overline{L} \in BPP \Rightarrow \overline{L} \in \Sigma_2 \Rightarrow L \in \Pi_2.$$

Use error reduction on input of length n. use $m = \text{poly}(n)$ random bits to get:

$$x \in L \Rightarrow \Pr_r[M(x,r) \text{ accepts}] \geq 1 - 2^{-n}$$
$$x \notin L \Rightarrow \Pr_r[M(x,r) \text{ accepts}] \leq 2^{-n}.$$

For $x \in \{0,1\}^n$ let S_x = set of strings r for which $M(x,r)$ accepts.

$$x \in L \Rightarrow |S_x| \geq 2^m(1-2^{-n})$$
$$x \notin L \Rightarrow |S_x| \leq 2^m \cdot 2^{-n}$$



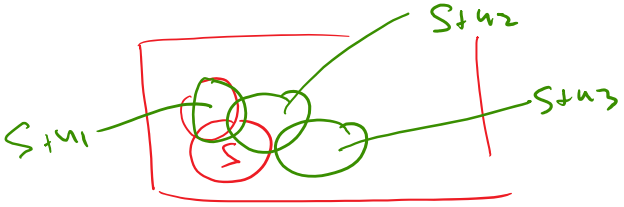
For $S \subseteq \{0,1\}^m$ $u \in \{0,1\}^m$ define $S+u = \{x+u \mid x \in S\}$

$+$ = bit-wise X-OR

$$\begin{array}{r} 10110 \\ 01101 \\ \hline 11011 \end{array}$$

$x + u + u = x$.

Let $k = \lceil \frac{m}{n} \rceil + 1$



Wednesday, May 16, 2018 8:28 AM

Lemma 1: for every $S \subseteq \{0,1\}^m$ s.t. $|S| \le 2^{m-n}$
For every choice of $u_1 \cdots u_k \in \{0,1\}^m$

$$\bigcup_{i=1}^{k} (S+u_i) \neq \{0,1\}^m \qquad k = \left\lceil \frac{m}{n} \right\rceil + 1 \cdot \text{poly}(n)$$

Proof: $|S+u_i| = |S| = 2^{m-n}$

$$\left|\bigcup_{i=1}^{k} (S+u_i)\right| \le k|S| = k\,2^{m-n} = 2^m \left(\frac{\lceil \frac{m}{n} \rceil + 1}{2^n}\right) < 2^m.$$

Lemma 2: For every $S \subseteq \{0,1\}^m$ s.t. $|S| > (1-2^{-n})2^m$
$\exists\, u_1, \ldots, u_k$ s.t.

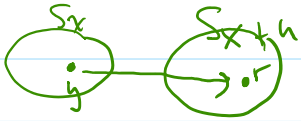
$$\bigcup_{i=1}^{k} (S+u_i) = \{0,1\}^m$$

Proof of Theorem from Lemma:

$x \notin L \Rightarrow |S_x| \le 2^{m-n} \Rightarrow \forall u_1 \cdots u_k \; \exists r \quad r \notin \bigcup_{i=1}^{k} S_x + u_i$

$\neg\, \exists u_1 \cdots u_k \; \forall r \quad r \in \bigcup_{i=1}^{k} S_x + u_i$

$x \in L \Rightarrow |S_x| > (1-2^{-n})2^m \Rightarrow \exists u_1 \cdots u_k \; \forall r \quad r \in \bigcup_{i=1}^{k} S_x + u_i$

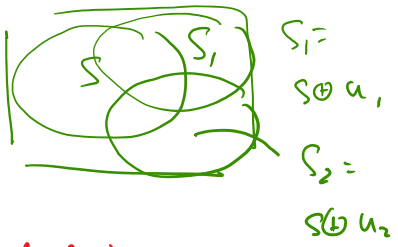


$x \in L \Leftrightarrow \exists u_1 \cdots u_k \; \forall r \quad r \in \bigcup_{i=1}^{k} S_x + u_i$

$\Leftrightarrow \exists u_1 \cdots u_k \; \forall r \quad \bigvee_{i=1}^{k} M(x, r+u_i)$ accepts

→ poly-time procedure expressible as Boolean formula.

Note: $M(x, r+u_i)$ accepts $\Leftrightarrow r+u_i \in S_x \Leftrightarrow r \in S_x + u_i$



Wednesday, May 16, 2018 8:28 AM

$k = \lceil \frac{m}{n} \rceil + 1$ $\quad n < m < \text{poly}(n)$.

Lemma 2: For every $S \subseteq \{0,1\}^m$ s.t. $|S| > (1-2^{-n})2^m$
$\exists u_1, \ldots, u_k$ s.t.
$$\bigcup_{i=1}^{k} (S + u_i) = \{0,1\}^m$$

Proof: Probabilistic Method.
Pick $u_1 \ldots u_k$ at random.
Will show: $\Pr\left[\bigcup_{i=1}^{k} S_{x+u_i} = \{0,1\}^n\right] > 0$

So there exists $u_1 \ldots u_k$ for which ↳ holds.

Prob $\exists$ bad r which is not in $\bigcup_{i=1}^{k} S_{x+u_i} = 1 - \Pr[\bigcup_{i=1}^{k} S_{x+u_i} = \{0,1\}^n]$
↳ will show < 1.

For fixed r: $r \in S_x + u_i \iff u_i \in S_{x+r}$ ($y \neq S_x$)
$y \oplus u_i = r \iff u_i = y \oplus r$.
$u_i \oplus y = r$.
$u_i \oplus y \oplus y = r \oplus y$
$u_i = r \oplus y$

$$\text{Prob}[u_i \notin S_{x+r}] = 1 - \frac{(1-2^{-n})2^m}{2^m} = 2^{-n}.$$

Prob [r not in any S_{x+u_i}] $\iff$ Prob [all $u_i \notin S_{x+r}$] $\quad k = \lceil \frac{m}{n} \rceil + 1$
$\leq (2^{-n})^k < 2^{-m}$

$$\text{Prob}[\text{fixed } r \notin \bigcup_{i=1}^{k} S_{x+u_i}] < 2^{-m}$$

$$\text{Prob}[\exists r \notin \bigcup_{i=1}^{k} S_{x+u_i}] < 2^{-m} \cdot 2^m \leq 1. \quad \#$$

$$\text{Prob}\left[\exists r \notin \bigcup_{i=1}^{k} S_{x+u_i}\right] < 2^{-m} \cdot 2^{m} \leq 1.$$

$$\Pr\left[\bigvee_{r} r \notin \bigcup S_{x+u_i}\right] < \sum_{r} \Pr\left[r \notin S_{x+u_i}\right]$$