P, NP, PH, PSPACE

Rutam

Note Title 2/8/2010

A deterministic TM is said to be in SPACE$(s(n))$ if it uses space $O(s(n))$ on inputs of length n. |||ly it is in TIME$(t(n))$ if it uses time $O(t(n))$ on such inputs.

A language L is polynomial-time decidable if $\exists K$ and a TM M to decide L s.t.

$M \in \text{TIME}(n^K)$

Note that K is independent of n.

e.g. PATH, i.e. does there exist a path between s and t in a given graph has a polynomial-time decider.

Median

Min/Max weight spanning tree.

P is the class of languages with polynomial time TMs.

$$P = \bigcup_{k} \text{TIME}(n^k)$$

Do all decidable languages belong to P?

<u>HAM PATH</u> $\exists$ path from s to t that visits every vertex in G exactly once?

<u>SAT</u> Given a boolean formula, $\exists$ a setting of its variables that makes the formula true?

$$f = (x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_3 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee x_2)$$

No polytime algorithms known for these problems.

They can be solved (decided) by polynomial-time NONDETERMINISTIC TMs.

"guess" the path

"guess" the assigment

Recall that a NTM accepts iff __any__ one of its computation paths accepts. The path amounts to a verification of the __YES__ answer.

We have NSPACE($s(n)$) and NTIME($t(n)$)

__NP__ is the class of languages that can be decided by polynomial-time NTMs.

$$NP = \bigcup_{k} NTIME(n^k)$$

Alternatively, NP is the class of languages with the property that membership ("YES") can be verified in polynomial-time using a polynomial-sized certificate.

e.g. __SAT__ : if F is satisfiable, a valid assignment is the certificate

__HAMPATH__ : if G has a __HAM__ path then the sequence of vertices visited is the certificate.

Clearly $P \subseteq NP$

From Savitch's theorem,

$$NPSPACE = PSPACE$$

since the space requirement only squares.

Also $NTIME(t(n)) \subseteq DTIME(2^{O(t(n))})$

EXPTIME := Languages that can be decided in exponential time.

$$P \subseteq NP \subseteq PSPACE \subseteq EXP$$

Amazingly, we do not know if these containments are strict, i.e., $\exists$ a language L

$L \in EXP$ and $L \notin PSPACE$

or $L \in PSPACE$ and $L \notin NP$

or $L \in NP$ and $L \notin P$

We know that $P \subsetneq EXP$ from the time hierarchy theorem.

$L \in NP \iff \exists NTM\ M$ s.t.

$L = \{x \mid \exists$ accepting path in M on input $x\}$

The class of languages that are complements of languages in NP is called CoNP

$L \in CoNP \iff \exists NTM\ M$ s.t.

$L = \{x \mid$ <u>Every</u> valid computation path of M is an accepting for $x\}$

$L \in CoNP \iff \bar{L} \in NP \iff$

$\bar{L} = \{x \mid x \notin L\}$ $\qquad L = \{x \mid x$ is not accepted by a TM for L on any path $\}$

— L is rejected on every path —

How to verify membership in a co-NP language?

Short (polynomial-size) certificate that $x \notin L$,

e.g. G does <u>not</u> have a HAM PATH?

F does not ———— satisfying assignment?

$\text{SAT} : \{ F \mid \exists x : F(x) = 1 \}$

$\overline{\text{SAT}} : \{ F \mid \forall x : F(x) = 0 \}$

$\Sigma_2 \text{SAT} : \{ F \mid \exists x \forall y \ F(x,y) = 1 \}$

$\Pi_2 \text{SAT} : \{ F \mid \forall x \exists y \ F(x,y) = 0 \}$

$\vdots$

$\Sigma_i \text{SAT} : \{ F \mid \underbrace{\exists x_1 \forall x_2 \ldots}_{i \text{ quantifiers}} \ F(x_1, \ldots) = 1 \}$

$\Pi_i \text{SAT} : \{ F \mid \forall x_1 \ldots \ldots F(\quad) = 0 \}$

Alternating turing Machines that can at each node of computation accept if any one path emanating from the node accepts or if all paths accept.

$$PH = \bigcup_i \underbrace{\Sigma_i \bigcup_k \text{Time}(n^k)} = \bigcup_i \Pi_i \bigcup_k \text{TIME}(n^k)$$

$$PH \subseteq PSPACE.$$

How hard are problems in PSPACE?
We don't know, but we can define the hardest problems.

A language L is said to be PSPACE-complete if

(a) $L \in$ PSPACE

(b) $\forall B \in$ PSPACE

$\exists$ polynomial-time reduction $B \to L$

i.e. using L as an oracle / procedure and polynomial additional time, B can be solved in PSPACE

"L is at least as hard as any problem in PSPACE".

(If only (b) holds, L is PSPACE-hard).

Do there exist COMPLETE languages for PSPACE?

TQBF: True Quantified Boolean Formula

$F = \forall x_1 x_2 \exists x_3 \forall x_4 \ldots \quad F(x_1, x_2 \ldots x_n)$

$\text{TQBF} = \{ F : F \text{ is true} \}$

Thm TQBF is PSPACE-complete.

Pf. TQBF $\in$ PSPACE

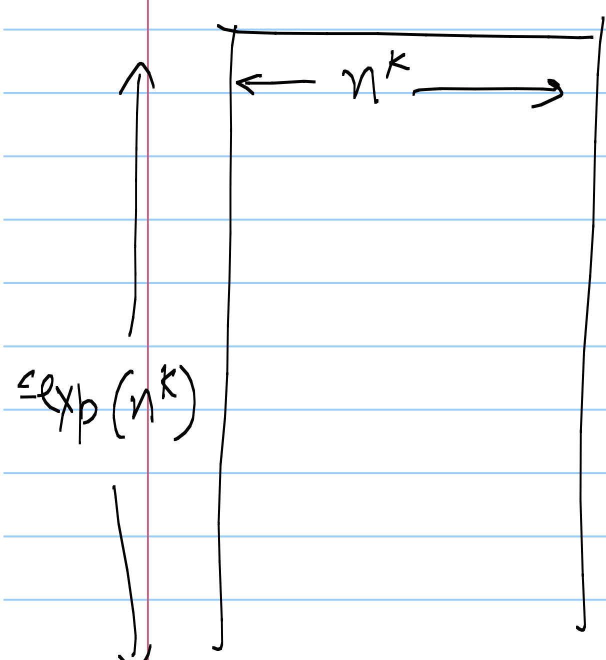
For this we just give an ATM that matches the quantifiers of a given formula and the depth of the tree is the # variables.

Any problem $B \in$ PSPACE has a reduction

$$B \longrightarrow L$$

Since $B \in$ PSPACE $\exists$ TM M that decides B

Examine the computation tableau of M on input w



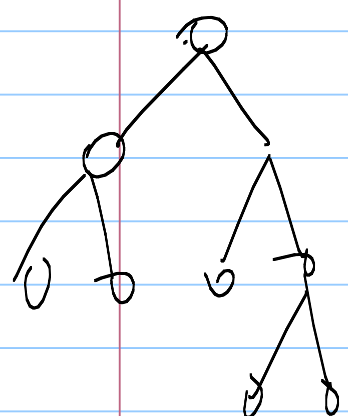
we can write a boolean formula F_B to check that the computation is valid, the start is valid and the end state is ACCEPT.

$w \in B \iff F_B$ is true

But F_B has exponential size!

$F_{start, accept, t}$: Formula that checks tableau from start to accept using at most t steps.

$$F_{s,a,t} = \exists u \left(\phi_{s,u,\lceil \frac{t}{2} \rceil} \wedge \phi_{u,a,\lfloor \frac{t}{2} \rfloor} \right)$$



if $t = 1$ or 0, we can write an explicit formula.

Does this work? No! still exponential size.

We can use universal quantifiers.

$$F_{s,a,t} = \exists u \; \forall (x,y) \in \{(s,u),(u,a)\} \quad F_{x,y,\lceil \frac{t}{2} \rceil}$$

$$\forall x \in \{y,z\} F \equiv \quad \forall x \; (x=y \vee x=z) \Rightarrow F$$

$$\forall x \quad \neg (x=y \vee x=z) \vee F$$

$$\forall x \quad \neg \big((x \to y) \wedge (y \to x) \vee \dots \big)$$

$$\forall x \quad \neg \; \big((\neg x \vee y) \wedge (\neg y \vee x) \vee \dots \big)$$

Every boolean $F \to$ AND/OR/NOT. Now $size(F) = O(n^{2k})$.