22/10/2021

Defn :- $AP = \bigcup_{c>0} ATIME(n^c)$.

Thm :- $AP = PSPACE$

Proof :- $PSPACE \subseteq AP$

Recall TQBF is PSPACE-complete

$\exists x_1 \; \forall x_2 \; \exists x_3 \cdots Q_n x_n \; \varphi(x_1, \ldots, x_n)$

guess the variables using $\exists$ state and $\forall$ state depending on the quantification of the variable.

$AP \subseteq PSPACE$?

Use the algorith similar to the one that shows $TQBF \in PSPACE$.

Defn :- $\Sigma_2 TIME(T(n))$ to be the set of languages accepted by a $T(n)$-time ATM M

whose initial state is labelled $\exists$ and on every input and every computation path no. of alternations $\leq i-1$.

Claim: $\Sigma_i^P = \bigcup_{c>0} \Sigma_i TIME(n^c)$

$\exists \underbrace{\exists \forall}_{\# \text{ alteration } = 1} \in \Sigma_2$

$\underline{\exists\exists\exists\exists \forall\forall\forall\forall} \Rightarrow$ # of alternation $= 1$

How would you capture Π_i^P

the only difference is that the start state should be labeled with $\forall$.

$\exists x_1 \in \{0,1\} \exists x_2 \in \{0,1\} \exists x_3 \in \{0,1\}$

$\exists (x_1, x_2, x_3) \in \{0,1\}^3$

FACTS:- (1) APSPACE $=$ EXP $:= \bigcup_{c>0} \mathrm{DTIME}(2^{n^c})$

(2) ALOG $=$ P

PH — $\exists\forall\exists$ SAT

— ATM

— Oracle Machines.

Defn: (Oracle Turing Machines)

Start with a DTM M

now add to it a special read-write tape, "ORACLE" tape.

three special states q_{query}, q_{Yes}, q_{No}

Let $O \subseteq \{0,1\}^*$ be some language

M does usual computation

At some point M moves into the state q_{query} and then let $y \in \{0,1\}^*$ be the string on ORACLE tape.

is $y \in O$?

And then depending on the answer to the above question

the machine M moves to either q_{yes} or q_{NO}. and then continue with its computation.

<u>Notation</u>:- a machine M with Oracle O is denoted by M^O and its output on input x is denoted by $M^O(x)$.

<u>EXAMPLE</u>:- (1) Given a 3-CNF formula φ, How much time

would you take to figure out whether $\varphi \in SAT$ or not.

Given access to a SAT oracle.

(2) $\varphi \in UNSAT$?

Given access to SAT oracle.

<u>Algo</u>:- give φ to the SAT-oracle.

if oracle says yes.
then you reject.

if oracle says no.
then you accept.

<u>NOTATION</u>:- P^{SAT}

$SAT \in P^{SAT}$

$UNSAT \in P^{SAT}$

EXACT − IS := $\{(G, k) \mid$ the largest IS in G is of size k $\}$

$\in \Sigma_2^P$

IS := $\{(G, k) \mid$ G has a IS of size at least k $\}$

$IS \leq_p SAT$

two queries with k and $k+1$.

$\underbrace{\text{yes} \qquad \text{No}}_{\text{overall yes}}$

Notation :- P^{SAT}, P^L where $L \in NP$

P^L simulate using P^{SAT}. $P^{SAT} := P^{NP}$

Characterization of Polynomial Hierarchy using Oracle machines.

Thm :- $\Sigma_i^P = NP^{\Sigma_{i-1}^P} \quad \forall i \geq 1$

$\Pi_i^P = coNP^{\Sigma_{i-1}^P} \quad \forall i \geq 1$

$\Sigma_1^P = NP = NP^{\Sigma_0^P} = NP^P$

$P^P = P$

$\Sigma_2^P = NP^{\Sigma_1^P} = NP^{NP}$

Proof:- $\Sigma_2^P = NP^{NP} = NP^{SAT}$

$\boxed{\text{Exercise:- } P^{NP} \subseteq \Sigma_2^P}$

easy direction: $\Sigma_2^P \subseteq NP^{SAT}$

$L \in \Sigma_2^P$. Then $\exists$ poly-time det. verifier V and polynomial q

s.t.

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)}$$
$$\forall u_2 \in \{0,1\}^{q(|x|)}$$
$$V(x, u_1, u_2) = 1.$$

Define $L' := \{ (x, u_1) \mid \forall u_2 \in \{0,1\}^{q(|x|)} \; V(x, u_1, u_2) = 1 \}$

$(x', u') \overset{?}{\in} L'$

$(x, u_1) \in L'$
$\Updownarrow$
$x \in L$

$L' \in coNP$

$\Rightarrow \overline{L'} \in NP$

NP^{SAT} - algorithm :

input x :

Step 1 :- non-deterministically guesses u_1

now you have fixed x, u_1.

Step 2 :- Make an Oracle query.

Does $(x, u_1) \in \overline{L'}$?

$L' \in coNP$

redue (x, u_1) to an instance of UNSAT

and then make the query to

the SAT Oracle.

if Oracle returns sat then Reject

if Oracle returns UNSAT then Accept.

<u>Harder direction:</u> $NP^{SAT} \subseteq \Sigma_2^P$

NP^{SAT} machine can make polynomial no. of 'dependent' queries.

<u>Σ_2^P algorithm.</u> $\exists \dots \forall \dots V(\cdot)$

poly time

$\Downarrow$ guess the non-deterministic choices

NP-machine

$\downarrow$ runs.

$\to$ query

guesses an answer to this query.

$\downarrow$ keep continue in this fashion.

Let's say there are m-non-deterministic choices that the NP machine makes

Let's say there are $q_1, \dots, q_k$.

and their answers are $a_1, \dots, a_k$.

$\exists c_1, \ldots, c_m \ \exists a_1, \ldots, a_k \ \exists u_1, u_3, u_6 \ldots$

s.t. $\quad \forall v_2, v_4, v_5 \ldots$

Can be done in PTIME

(N accepts x using choices $c_1, \ldots, c_m$ and answers $a_1, \ldots, a_k$ **AND**

if $a_i = 1$ then check $q_i(u_i) = 1$

if $a_i = 0$ then check $q_i(v_i) = 0$

if $a_i = 1$, it means q_i is satisfiable.

and to certify that give an assignment u_i

if $a_i = 0$, it means q_i is unsatisfiable

Suppose $a_2 = 0$

if $a_i = 1$. then $q_i(u_i) = 1$

if $a_i = 0$ then $q_i(v_i) = 0$

if x is never accepted by NP^{SAT} machine.

$\forall$ non-deterministic choice.

$\forall$ ~~answers~~ to the queries ($\forall$ queries)

N doesn't accept x.