Undergraduate Complexity Theory
Lecture 24: Oracle TMs & P^{NP}

Marcythm

July 26, 2022

1 Lecture Notes

recall: $\Sigma_1\mathsf{P} = \mathsf{NP}, \Pi_1\mathsf{P} = \mathsf{coNP}, \Sigma_0\mathsf{P} = \Pi_0\mathsf{P} = \mathsf{P}, \mathsf{PH} = \bigcup_{k\in N} \Sigma_k\mathsf{P} = \bigcup_{k\in N} \Pi_k\mathsf{P}$ (different notations)

Theorem 1.1. $\mathsf{NP} = \mathsf{P} \implies \mathsf{PH} = \mathsf{P}$.

Proposition 1.2. $\mathsf{MC} \in \Pi_2\mathsf{P}$*, where* MC *means* $\mathsf{MINIMUM\text{-}CIRCUIT}$.

Proof. Given C, $\langle C\rangle \in \mathsf{MC} \iff \forall\langle C'\rangle, |C'| < |C| : \exists y, |y| = \#$ inputs to $C, C' : C(y) \neq C'(y)$. The innermost statement $C(y) \neq C'(y)$ is what the verifier V checks. $\square$

Corollary 1.3. $\mathsf{NP} = \mathsf{P}(\iff \mathsf{SAT} \in \mathsf{P}) \implies \mathsf{MC} \in \mathsf{P}$.

Proof. For fixed C, C' what left to decide? "$\exists y : C(y) \neq C'(y)$", which is known as DCF (Different Circuit Functionality problem) in hw4: $\mathsf{DCF} \in \mathsf{NP}$. By assumption $\mathsf{DCF} \in \mathsf{P}$, so exists poly-time deterministic TM A deciding DCF. Now $\langle C\rangle \in \mathsf{MC} \iff \forall\langle C'\rangle, |C'| < |C| : \langle C, C'\rangle \in \mathsf{DCF} \iff \forall\langle C'\rangle : A(\langle C, C'\rangle)$ accepts. Reverse the accept/reject condition, we get $\mathsf{MC} \in \mathsf{coNP} \implies \overline{\mathsf{MC}} \in \mathsf{NP} \implies \overline{\mathsf{MC}} \in \mathsf{P} \implies \mathsf{MC} \in \mathsf{P}$. $\square$

Suppose we can solve SAT efficiently, we can solve a bunch of problems ($\mathsf{3COL}$, UNSAT, $\mathsf{HAMPATH}$, etc.) efficiently, also …$\mathsf{MIN\text{-}CIRCUIT}$?

ketpoint: Having a black box solving a problem is not as good as having the code for the problems.

To formalize "pesudocode" with an unimplemented `SolveSAT()` function:

Definition 1.4. A SAT-oracle TM is a TM with an extra power: an r/w "oracle tape", and an extra instruction "ORACLE". When operating ORACLE instruction, if oracle tape contains y, it's replaced by "1" if $y \in \mathsf{SAT}$, and "0" if $y \notin \mathsf{SAT}$, with a cost of only 1 step.

Definition 1.5. More generally, we can define a B-oracle TM for every language B.

Definition 1.6. $\mathsf{P}^{\mathsf{SAT}} = \{L : L$ solvable in poly-time by a SAT-oracle TM$\}$.

e.g. $\mathsf{NP} \subseteq \mathsf{P}^{\mathsf{SAT}}, \mathsf{coNP} \subseteq \mathsf{P}^{\mathsf{SAT}}, \mathsf{CHROMATIC4} \in \mathsf{P}^{\mathsf{SAT}}$, but $\mathsf{MINIMUM\text{-}CIRCUIT}$ may not in $\mathsf{P}^{\mathsf{SAT}}$.

$\mathsf{P}^{B} \subseteq \mathsf{P}^{\mathsf{SAT}}$ if $B \in \mathsf{NP}$, and $\mathsf{P}^{B} = \mathsf{P}^{\mathsf{SAT}}$ if B is NP-complete (also for coNP-complete).

Notation 1.7. $\mathsf{P}^{\mathsf{NP}} = \mathsf{P}^{\mathsf{SAT}}$.

Definition 1.8 (Turing reduction (Cook reduction))**.** $A \leq_T^P B$ if $A \in \mathsf{P}^B$.

Theorem 1.9. $\mathsf{P}^{\mathsf{NP}} \subseteq \Sigma_2\mathsf{P}$.

Proof. Assume $L \in \mathsf{P}^{\mathsf{NP}}$ solved by poly-time SAT-oracle TM A, need poly-time verifier $V(x, u_1, u_2)$ s.t. $x \in L \iff A(x)$ acc $\iff \exists u_1 \forall u_2 V(x, u_1, u_2)$ acc. Here

$$u_1 = \langle \text{answers (and satisfying assignments if satisfiable) to } A(x)\text{'s oracle queries}\rangle$$

$$u_2 = \langle \text{some assignments for those unsatisfiable } \phi\text{s in } u_1\rangle$$

$V(x, u_1, u_2)$ checks those satisfying assignments in u_1, and checks all the assignments given in u_2 cannot satisfy the corresponding ϕs, then simulates $A(x)$. $\square$

Corollary 1.10. $\mathsf{co\text{-}P}^{\mathsf{NP}} = \mathsf{P}^{\mathsf{NP}}$*, so* $\mathsf{P}^{\mathsf{NP}} \subseteq \Pi_2\mathsf{P}$.

2 Reading

2.1 sipser 6.3 (Turing Reducibility)

definition of “decidable relative”, Turing reducible

2.2 sipser 9.2 (Relativization)

definition of oracle Turing Machine