Seminar Theoretical Computer science. April 9th, 2018.

Exercise 1. What is the error in the following fallacious argument that $\mathsf{P} \neq \mathsf{NP}$? Assume that $\mathsf{P} = \mathsf{NP}$, then SAT $\in \mathsf{P}$ and so for some k, SAT $\in \mathsf{TIME}(n^k)$. Therefore NP-completeness of SAT implies $\mathsf{P} = \mathsf{NP} \subseteq \mathrm{TIME}(n^k)$, But by the time hierarchy theorem, $\mathsf{TIME}(n^{k+1})$ contains a language not in $\mathsf{TIME}(n^k)$, which contradicts $\mathsf{P} \subseteq \mathsf{TIME}(n^k)$. Therefore $\mathsf{P} \neq \mathsf{NP}$.

Exercise 2. Show that $\mathsf{PSPACE}(n^3) \not\subseteq \mathsf{NPSPACE}(n)$.

Exercise 3. Let A^{c} be the complement of language A. Show that $\mathsf{P}^A = \mathsf{P}^{A^{\mathsf{c}}}$ and $\mathsf{NP}^A = \mathsf{NP}^{A^{\mathsf{c}}}$.

Exercise 4. Define the USAT problem to be

$$\{\langle\phi\rangle : \phi \text{ is a Boolean formula that has a single satisfying assignment}\}$$

Show that USAT $\in \mathsf{P}^{\mathrm{SAT}}$.

Exercise 5. Show that if $\mathsf{P}^{\mathrm{SAT}} = \mathsf{NP}$ then $\mathsf{NP} = \mathrm{co}\mathsf{NP}$.

Exercise 6. Let MIN-FORMULA be the language defined in the previous seminar. Show that the complement of this language is in $\mathsf{NP}^{\mathrm{SAT}}$.

Exercise 7. Show that $\mathsf{P}^{\mathrm{TQBF}} = \mathsf{NP}^{\mathrm{TQBF}}$.

Exercise 8. Imagine you are given two oracles and one of them is the set TQBF. You don't know which one. Design an algorithm that can access the two oracles A and B and that decides TQBF in polynomial time.

Solutions

Exercise 2. By the hierarchy theorems we have that $\mathsf{PSPACE}(n^3) \not\subseteq \mathsf{PSPACE}(n^2)$
and by Savitch theorem $\mathsf{NPSPACE}(n) \subseteq \mathsf{PSPACE}(n^2)$.

Exercise 3. This follows almost directly from the definition of oracle computations: the oracle machines can simply negate the answers from their queries.

Exercise 4. In fact, USAT $\leq_p$ SAT. Indeed: $\phi \in$ USAT if and only if the formula with variables x and y given by

$$\phi(x) \ \wedge \ \phi(y) \ \wedge \ \text{“}x \neq y\text{”}$$

is satisfiable. To rewrite the last part using $\wedge$, $\vee$ and negation operators, note that

$$x \neq y \iff \bigvee_{i=1}^{m} x_i \neq y_i \iff \bigvee_{i=1}^{m} \big((x_i \wedge \bar{y}_i) \vee (\bar{x}_i \wedge y_i)\big).$$

Exercise 5. Note that $A \in \mathsf{P} \Leftrightarrow A^{\mathsf{c}} \in P$, because in a decider we can flip accept and reject states. For the same reason this is true for P^B. Hence,

$$A \in \mathsf{NP} \iff A \in \mathsf{P}^{\mathrm{SAT}} \iff A^{\mathsf{c}} \in \mathsf{P}^{\mathrm{SAT}} \iff A^{\mathsf{c}} \in \mathsf{NP} \iff A \in \mathrm{co}\mathsf{NP}.$$

Exercise 6. We need to show that the following problem is in $\mathsf{NP}^{\mathrm{SAT}}$.

The problem NOT MIN-FORMULA:

Input: A Boolean formula ϕ.
Question: Does there exist a shorther equivalent formula ψ?

The non-deterministic machine guesses the formula ψ and then uses the SAT-oracle to check whether $\psi \equiv \phi$, i.e., whether the formula $(\phi \wedge \overline{\psi}) \vee (\overline{\phi} \wedge \psi)$ is satisfiable.

Exercise 7. This is proven in Sipser's book Theorem 9.20 p378.

Exercise 8. Given a quantified Boolean formula

$$\psi = \exists y_1 \forall y_2 \exists y_3 \forall y_4 \dots Q_m y_m \left[\phi(y_1, y_2, \dots, y_m)\right].$$

the algorithm asks both oracles whether ψ is true. If both oracles give the same answer, then the algorithm accepts or rejects accordingly. Otherwise, we need to discover which of the oracles correclty decides TQBF. The idea is that we let both oracles play the formula game against each other. See Sipser's book for a description of this game. The correct oracle will win the game. We explain in detail how this done. Recall that the players need to create an assignment $x_1, \dots, x_m$. The $\exists$-player uses the oracle that believes the formula is true. The $\forall$-player uses the other oracle.

Each time the exists player needs to assign the next variable x_i, he will ask his oracle both for $b = 0$ and $b = 1$ whether the formula

$$\forall y_i \exists y_{i+1} \dots Q_m y_m \phi(x_1, x_2, \dots, x_{i-1}, b, y_{i+1}, \dots, y_m)$$

is true or false. If for both values, it answered "false", then ψ is rejected; in this case we say that the oracle replied *inconsistently*. Otherwise, let $x_i = b$ for a value b for which the answer was "true".

If the $\forall$-player needs to assign a next variable x_i, she asks her oracle both for $b = 0$ and $b = 1$ whether

$$\forall y_i \exists y_{i+1} \dots Q_m y_m \phi(x_1, x_2, \dots, x_{i-1}, b, y_{i+1}, \dots, y_m)$$

is true or false. If for both values it answered "true", then the formula ψ is accepted (again, we say that the oracle replied *inconsistently*). Otherwise, she sets $x_i = b$ for a value b for which the answer was "false".

After $x_1, \dots, x_m$ are assigned, we evaluate $\phi(x_1, \dots, x_m)$. We accept ψ if the value equals 1 and otherwise we reject.

Why does this algorithm work? If the oracles both give the same answer on input ψ, then the algorithm is correct. Otherwise, precisely one oracle claims that ψ is true.

In case ψ is true, the $\exists$-player uses the correct oracle (representing TQBF). By construction, he plays a winning strategy, and thus never replies inconsistently. Either the $\forall$-player plays inconsistently or the game terminates with $\phi(x_1, \dots, x_m) = 1$. In both cases ψ is accepted.

Now assume that ψ is false. Thus the $\forall$-player uses the correct oracle. Either the $\exists$-player plays inconsistently or the game terminates with $\phi(x_1, \dots, x_m) = 0$. In both cases ψ is rejected. Hence, our algorithm decides TQBF.