Exercises Complexity Theory

Lecture 4

May 12, 2025

Only the exercises where points are given can be handed in.
(The maximum number of points per exercise is written in the margin.)

To be handed in on **May 19, 2025**, in Brightspace under Assignment 4, **deadline: 11:00.**

Exercise 1. Prove the following statements.

(10) (a) If $A \in \mathbf{P}$ and $B \in \mathbf{P}$, then $A \cap B \in \mathbf{P}$.

(10) (b) If $A \in \mathbf{P}$ and $B \in \mathbf{P}$, then $A \setminus B \in \mathbf{P}$.

(10) (c) If $A \in \mathbf{NP}$ and $B \in \mathbf{NP}$, then $A \cdot B \in \mathbf{NP}$.

(d) We have seen that if $A \in \mathbf{P}$, then $\overline{A} \in \mathbf{P}$ in the lectures. Why can't we apply a similar proof strategy to show that $A \in \mathbf{NP}$ implies $\overline{A} \in \mathbf{NP}$

Exercise 2. Prove the following statements.

(10) (a) If $B \in \mathbf{NP}$ and $f : \{0,1\}^* \to \{0,1\}^*$ is computable in polynomial time, then $\{a \in \{0,1\}^* \mid f(a) \in B\} \in \mathbf{NP}$.

(10) (b) If $A \in \mathbf{P}$ and $A \in \mathbf{NPH}$, then $\mathbf{P} = \mathbf{NP}$.

Exercise 3. Do the following hold? Give a proof or a counterexample.

(a) If $A \subseteq B \leq_P C$, then $A \leq_P C$.

(b) If $A \leq_P B \subseteq C$, then $A \leq_P C$.

Exercise 4. Put the following propositional formulas in CNF and determine if they are satisfiable.

(5) (a) $\varphi_1 = (p \to \neg q) \wedge (p \wedge (q \vee \neg r)) \wedge (\neg p \to q) \wedge (\neg(q \wedge p) \vee r)$.

(5) (b) $\varphi_3 := (p \wedge \neg q) \vee (q \wedge r)$.

(5) (c) $\varphi_2 := ((p \to q) \wedge (q \to r)) \to (p \to r)$.

Exercise 5. We consider boolean logic with three connectives $\vee, \wedge, \to$, negation $\neg$ and a constant $\bot$. We know that SAT is in **NPH**. For each of the following problem, determine whether it is in **P** or in **NPH**. Prove your answer.

(10) (a) The SAT problem for formulas restricted to $\vee, \bot$

(10) (b) The SAT problem for formulas restricted to $\to, \neg$.

(15) **Exercise** 6. Suppose we have a polynomial algorithm for SAT, that is: an f that computes for a boolean formula φ whether φ is satisfiable or not, in polynomial time in the size of φ.

Give an algorithm g that computes, in polynomial time, a satisfying assignment for a formula φ (in case φ is satisfiable, and return 0 otherwise).