Computational Complexity

Reading guide and Homework for **Week 02**

1 Reading Guide

In the previous two classes we have learned:

- The time complexity classes: P, E, EXP, and the *time hierarchy theorem* showing that these classes are different. Read §3.1.
- We learned about the class NP, and saw several examples. Read §2.1.
- We learned about exhaustive search algorithms, and why they are not efficient. We learned about Gödel's problem and Nash's problem. This is not in the book, but at the end of your reading, you should read the related section §2.7, and the "Chapter Notes and History" section that follows it.
- We learned about reductions, and NP-completeness. Read §2.2.
- We proved the Cook-Levin Theorem, that 3SAT is NP-complete. Read §2.3.
- We showed the NP-hardness of CLIQUE and other problems.

There were several details that we covered very quickly, and which **must** be looked at carefully, in order to properly understand what is going on. This kind of familiarity will be necessary to understand what's being said in class.

2 Exercise guide

Before doing the homework, you should train yourself with the following exercises from the book. Solutions will be provided for the exercises marked with an asterisk (see the webpage). You should still try to do them by yourself before looking at the solutions.

$*$ 2.2.

• 2.3.

$*$ 2.4

• 2.8

• 2.9

$*$ 2.10

$*$ 2.15

• 2.17

• 2.31

$*$ 2.19

$*$ 2.20

• 2.21

3 Homework

You should turn in solutions for the following exercises before the beginning of the next class. You can turn in solutions electronically (to `bruno.loff+homework@gmail.com`), or in paper, in which case we will scan them ourselves. If you given them in paper, please respect the following rule, which is meant to make scanning easy:

Solutions should be given in separate (not stapled) a4 sheets of paper, and your name and number should appear clearly on every page.

You should feel free to discuss the exercises with your colleagues, but when you write down the answer, you must do it alone, without any help. (

1. (Exercise 2.16) In the MAXCUT problem, we are given an undirected graph G and an integer K and we have to decide whether there is a subset of vertices S such that there are at least K edges in G having one endpoint in S and one endpoint outside of S. Prove that this problem is NP-complete.

2. Suppose that God gave you a magical computer device, called an *oracle*, which could solve CLIQUE. I.e., you could feed the device any n-vertex graph G and any number $1 \leq k \leq n$, and the device would tell you whether or not the graph G has a clique of size k. (We say that the oracle solves the *decision problem.*)

   Show that an algorithm can use such an oracle to find a largest-possible clique in any given graph. (This is called the *search problem.*)

   In other words, show that search reduces to decision for the CLIQUE problem.

   You should start by reading and understanding §2.5 of the book, but you should solve the above exercise directly, i.e., without appealing to the results proven in that section (which show the same thing for the SAT problem).

3. (extra credit) Now let A be any NP-complete problem. Suppose that $A = \{x \in \{0,1\}^n \mid n \in \mathbb{N}, \exists y \in \{0,1\}^{n^c} \quad M(x,y) = 1\}$, where M is a Turing machine running in polynomial time (i.e., M is a polytime verifier for A). Show that an oracle for solving A — i.e., which given x will answer whether $x \in A$ — can be used to find a y such that $M(x,y) = 1$, if any such y exists.

   Prove this without appealing to the argument appearing in the second part of Theorem 2.18 in §2.5. Hint: Construct a decision problem A' in NP which will help you find such a y, by extending a partial solution to a full solution.

4. (extra credit) We saw in class that CLIQUE is NP-complete. Given a 3CNF formula ψ with m clauses we constructed a graph G_ψ such that G_ψ has an m-clique if and only ψ is satisfiable. However, the number of m-cliques of G_ψ is not exactly equal to the number of satisfying assignments of ψ.

   Show that there is a polynomial-time reduction from 3SAT to CLIQUE where this property is preserved. I.e., given ψ a 3CNF, show how to produce in polynomial time a pair (G_ψ, k_ψ) such that G_ψ has exactly as many k_ψ-cliques as the number of satisfying assignments of ψ.