22C:131 Homework 2

Due: Thursday, 10/18

Notes: (a) Undergraduate students are required to solve the first 5 problems in the homework. Graduate students are required to solve all problems. The problem numbers, definition numbers, and Theorem numbers refer to the textbook, by Arora and Barak. (b) It is possible that solutions to some of these problems are available to you via other theory of computation books, on-line lecture notes, wikipedia, etc. If you use any such sources, please acknowledge these in your homework *and* present your solutions in your own words. You will benefit most from the homework, if you sincerely attempt each problem on your own first, before seeking other sources. (c) It is okay to discuss these problems with your classmates. Just make sure that you take no written material away from these discussions *and* (as in (b)) you present your solutions in your own words. When discussing homework with classmates please be aware of guidelines on "Academic Dishonesty" as mentioned in the course syllabus.

1. Let $L \subseteq \{0,1\}^*$ be a language. Let L^+ denote the language consisting of all strings x such that x is obtained by concatenating a finite number strings belonging to L. (In other words, for $n \geq 1$, let $L^n = \{x_1 \cdot x_2 \cdot \ldots \cdot x_n \mid x_1, x_2, \ldots, x_i \in L\}$, where "$\cdot$" denotes the concatenation operator. Then define $L^+ = \cup_{n \geq 1} L^n$.) Show that if $L \in NP$ then $L^+ \in NP$.
2. Show that `HALT` is NP-hard.
3. Show that if $P = NP$ there is a polynomial time algorithm that takes an undirected graph $G = (V, E)$ as input and returns a largest independent set in G as output. Theorem 2.18 provides an indirect way of solving this problem, using a reduction to `SAT`. Here I want you to solve this problem directly, without reliance on `SAT`, by using the fact that if $P = NP$, then `INDSET` has a polynomial time algorithm.
4. Problem 2.14 on *polynomial-time Cook reductions* (Page 65).
5. Definition 2.19 and 2.20 provide alternate ways of defining the class $coNP$. Show that these two definitions define the same set of languages.
6. Problem 2.30 on *Berman's Theorem* (Page 66). The textbook contains a "hint" for the problem. Let me elaborate on the hint by saying that the downward self-reducibility argument in the proof of Theorem 2.18 essentially implies the following simple algorithm for solving `3SAT`. Substitute $x_1 = TRUE$ in the given formula to get a new formula with only $n-1$ variables and then substitute $x_1 = FALSE$ in the given formula to get a new formula with only $n-1$ variables. Now we have two formulas to test, but both have $n-1$ variables each. We continue in this manner until the formulas become small enough to test satisfiability of each formula by brute force. The problem with this algorithm is that we end up producing exponentially many formulas. Now if there were an NP-complete unary language L, the fact that `3SAT` $\leq_P L$ could be used to significantly prune the number of formulas we need to test.
7. Problem 2.33 (Page 67).