Homework #9: Complexity Theory
Due Date: Friday, 30 November 2012

Guidelines
Please make sure you adhere to the policies on collaboration and academic honesty as outlined in the syllabus.

Reading
Chapter 9.1, Chapter 34

Problems

Problem A (25 points)
“Work entirely on your own.”

Assume that you are given a collection of n items that belong to an underlying total order, and that the only information you can gather is by performing a comparison “e_j < e_k?” for arbitrary elements e_j and e_k. Show that you can determine the second largest of the elements using at most n + ⌈lg n⌉ − 2 comparisons.

Problem B (25 points)
“Work entirely on your own.”

Chapter 34 gives a construction for reducing an instance of the 3-CNFSAT problem to an instance of the SUBSET-SUM problem. Describe the precise set S of numbers and the target value t that corresponds to the 3-CNF formula

\[ \phi = (x_1 \lor \neg x_2 \lor x_3) \land (\neg x_2 \lor \neg x_3 \lor x_4) \land (\neg x_1 \lor x_2 \lor \neg x_4). \]

Describe which subset corresponds to the satisfying assignment of

\[ x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0. \]
Problem C (25 points)

“Work entirely on your own.”

A boolean formula is in *discunctive normal form* (DNF) if it consists of a disjunction (“or”) of terms, each of which is the conjunction (“and”) of one or more literals. For example, the formula

\[(\neg x_1 \land x_2 \land \neg x_3) \lor (x_2 \land x_3) \lor (x_1 \land \neg x_2 \land \neg x_3)\]

is in disjunctive normal form. The DNF-SAT problem asks whether such a formula is satisfiable.

1. Shows that DNF-SAT is in P.

2. Show that any CNF formula with at most three literals per clause can be converted to a DNF formula by repeated application of the distributive law. For example, \((x_1 \lor x_2 \lor \neg x_3) \land (\neg x_1 \lor \neg x_2)\) is equivalent to \((x_1 \land \neg x_2) \lor (x_2 \land \neg x_1) \lor (\neg x_3 \land \neg x_1) \lor (\neg x_3 \land \neg x_2)\).

3. What is the error in the following argument that \(P = NP\)?

   If given an instance of 3-CNF-SAT, which is an NP-hard problem, we can convert the formula to DNF, based on part (2), and then apply the polynomial algorithm from part (1). We therefore have a polynomial-time algorithm for an NP-hard problem, and thus \(P = NP\).

Problem D (25 points)

“You may discuss ideas with other students.”

Problem 34-3 parts (d–f) only.

Problem E (EXTRA CREDIT – 10 points)

“You may discuss ideas with other students.”

Prove that any algorithm for finding the second largest of \(n\) elements (as in Problem A) requires \(n + \lceil \lg n \rceil - 2\) comparisons in the worst case.