CSC 363 - Summer 2005
Assignment 4

due on Tuesday, August 9th, at 6pm

Problem 1 [15p]

Recall that we defined **coNP** as the class of languages L such that the complement of L is in **NP**. Formally, $\mathbf{coNP} = \{L | \overline{L} \in \mathbf{NP}\}$.

(a) [5p] We can define the class **coNP-hard** in a manner similar to that in which we defined the class **NP-hard**. So, we say that language L is **coNP-hard** if every language $L' \in \mathbf{coNP}$ is polytime reducible to L, i.e. $L' \leq_p L$. Furthermore, we say L is **coNP-complete** if L is both **coNP** and **coNP-hard**.

Show that, for every language L, L is **coNP-hard** iff $\overline{L}$ is **NP-hard**.

(b) [10p] Show that if there exists some **NP-complete** language that is in **coNP**, then $\mathbf{NP} = \mathbf{coNP}$.

Problem 2 [10p]

This is part of problem 7.23 on p.296 of Sipser v2. Let CNF_k be the language consisting of encodings of satisfiable CNF formulas, in which every individual variable appears at most k times. Note that we are counting both positive and negative appearances. For example, $f = (x_1 \vee x_1 \vee x_2 \vee \overline{x}_1) \wedge (x_1 \vee x_3) \wedge \overline{x}_2$ is a satisfiable CNF formula in which x_1 has 4 appearances, x_2 has 2 appearances and x_3 has one appearance. So $\langle f \rangle \in \text{CNF}_4$, but $\langle f \rangle \notin \text{CNF}_3$. Also note that the size of the clauses does not play any role in deciding membership to CNF_k.

Prove that CNF_3 is **NP-complete**.

Problem 3 [20p]

In the Simple Knapsack problem, we are presented with m weights, $w_1, \ldots, w_m$, and a bound W, all in binary notation. We need to select a subset of weights $S \subseteq \{1, \ldots, m\}$, such that the sum of those weights, $\sum_{i \in S} w_i$ is maximum, while not exceeding W. In the decision version, we are also given a bound T, and we have to accept iff there is a subset with total weight at least T and not more than W.

Formally,

$$\text{SKD} = \{\langle w_1, \ldots, w_m, T, W \rangle \mid \exists S \subseteq \{1, \ldots, m\} \text{ such that } T \leq \sum_{i \in S} w_i \leq W\}.$$

It is easy to show that SKD is **NP-complete** using a reduction from SUBSET-SUM. You may use this fact without proof.

(a) [5p] Show that, if $\mathbf{P} = \mathbf{NP}$, then there exists a polynomial time algorithm, which, on input $\langle w_1, \ldots, w_m, W \rangle$, outputs a value T, such that T is the maximum sum achievable with these weights, that is at most W. In other words, $T = \max(\sum_{i \in S} w_i \mid S \subseteq \{1, \ldots, m\}$ and $\sum_{i \in S} w_i \leq W)$.

Note: For this part, assume that all the values mentioned (weights, bounds) are positive integer numbers.

(b) [15p] We have seen that for problems like CLIQUE or VC, if the target parameter is a constant or the maximum value minus a constant, the problem has polynomial time solutions; and if the target parameter is half the maximum value, the problem is still **NP-complete**. We investigate whether this is the case for SKD.

For each of the following languages, prove either that it has a polynomial time algorithm, or that it is **NP-complete**:

$$\textsc{Half-SKD} = \{\langle w_1, \ldots, w_m, W\rangle \mid \exists S \subseteq \{1, \ldots, m\} \text{ such that } \frac{W}{2} \leq \sum_{i \in S} w_i \leq W\}$$

$$\textsc{k-SKD} = \{\langle w_1, \ldots, w_m, W\rangle \mid \exists S \subseteq \{1, \ldots, m\} \text{ such that } W - k \leq \sum_{i \in S} w_i \leq W\}$$

Note: For this part, assume that all the values mentioned (weights, bounds) are positive *rational* numbers. You may assume we represent the rational number p/q (where p, q are positive integers) as $\langle p, q\rangle$. In other words, the length of the encoding of p/q is in the order of the sum of the lengths of the encodings of p and of q: $|\langle p, q\rangle| = O(|\langle p\rangle| + |\langle q\rangle|)$. In a nutshell, what I'm saying is that you can perform *division* as long as both numerators and denominators are reasonably sized.