COMS 4236: Introduction to Computational Complexity, Spring 2018

Problem Set 4, due Thursday March 29, 11:59pm on Courseworks

Please follow the homework submission guidelines posted on Courseworks.

Problem 1. [15 points] Recall the *Subset Sum* problem:
Input: A collection S of positive integers $s_1, s_2, \cdots, s_m$ and another positive integer t.
Question: Is there a subcollection of S whose sum is equal to t?
As we said in class, the Subset Sum problem can be solved in pseudopolynomial time, but the problem is NP-complete if the numbers are represented in binary as usual.
Use the NP-completeness of the Subset Sum problem to show the NP-completeness of the following *Partition Problem*:
Input: A collection S of positive integers $s_1, s_2, \cdots, s_m$
Question: Is there a partition of the collection S into two subcollections S_1, S_2 (where $S_1 \cap S_2 = \emptyset$, $S_1 \cup S_2 = S$) whose sums are equal: $\sum_{s_i \in S_1} s_i = \sum_{s_i \in S_2} s_i$?

Problem 2. [20 points] Let $G=(V,E)$ be a directed graph (without any self-loops). A *kernel* of G is a subset K of V such that (1) there is no edge (u,v) with both nodes u,v in K (i.e. K is an independent set of nodes), and (2) for every node $v \notin K$ there is a node $u \in K$ such that $(u,v) \in E$, i.e. every node is either itself in K or has an incoming edge from some node in K. Note that some graphs do not have any kernel.

a. Show that if u,v are two nodes of G such that the graph has both edges (u,v) and (v,u) and there are no other edges entering the nodes u,v, then any kernel of G must include exactly one of the two nodes u,v.
b. Show that if three nodes u,v,w form a cycle $u \to v \to w \to u$ in G, then any kernel of G must contain some node x (distinct from u,v,w) that has an edge to at least one of the three nodes u,v,w.
c. Show that it is NP-complete to determine whether a given directed graph has a kernel. (*Hint:* You can reduce from 3SAT. You can use parts a and b for you variable and clause gadgets respectively.)

Problem 3. [25 points] In the *Steiner Tree* problem, we are given a set N={1,…,n} of n cities, a subset M⊆N of *mandatory* cities (the rest are optional), and the pairwise distances $d(i,j)>0$, $1 \le i,j \le n$ between the cities, which are assumed to be positive integers and symmetric (i.e. $d(i,j)=d(j,i)$ for all i,j). The problem is to find a connected graph H=(V,E) that includes all the mandatory cities (and any number of optional cities), i.e. M⊆V, and which has minimum total distance $d(H) = \sum \{d(i,j) \mid (i,j) \in E\}$.
1. Show that the optimal graph is a tree (i.e. has no cycles).

2. Formulate the decision version of the Steiner Tree problem.
3. Suppose that we are given a subroutine that solves the decision version in polynomial time. Give a polynomial time algorithm that uses this subroutine to solve the optimization problem, i.e. which returns a Steiner tree with minimum total distance.
(*Hint:* First compute the value of the optimal Steiner tree; keep in mind that the distances are given in binary. Then compute the optimal Steiner tree itself.)
4. Show that the decision version of the Steiner tree problem is NP-complete even if all the distances are 1 or ∞.
(*Hint*: You can reduce if you want from the Node Cover or from the Set Cover problem.)

Problem 4. [20 points] A Boolean formula (expression) is in *Disjunctive Normal Form* (DNF) if it is the disjunction (the OR) of a set of conjunctions (AND's) of literals. For example the formula $(x_1 \wedge \overline{x}_2) \vee (\overline{x}_2 \wedge \overline{x}_3 \wedge x_4) \vee (\overline{x}_1 \wedge x_3 \wedge \overline{x}_4)$ is in DNF.
(a) Show that the Satisfiability problem for Boolean formulas in DNF is in P.
(b) A Boolean formula is called a *tautology* if every truth assignment satisfies it.
Show that the problem DNF-TAUT={ DNF formula F | F is a tautology } is coNP-complete.

Problem 5. [20 points] Consider optimization problems Π with solutions that are polynomially bounded and polynomially verifiable, and with polynomially computable integer values. More specifically, instances of Π and solutions are represented, as usual, as strings over some alphabet. There is a polynomially balanced and polynomially verifiable binary relation S that relates instances to solutions, i.e. $S(x,y)$ holds for strings x,y iff y is a solution for the instance x of Π (recall that "S is polynomially balanced" means that $S(x,y)$ implies that $|y| \leq p(|x|)$ for some polynomial p, and "S is polynomially verifiable" means that there is a polynomial-time algorithm which given x,y as input determines whether $S(x,y)$ holds or not). There is a polynomially computable integer-valued function $f(x,y)$ that gives the value of solution y for the instance x.

Let Π_1 be a minimization problem and Π_2 a maximization problem as above, with the same set of instances; the two problems have their own sets of solutions specified by relations S_1, S_2, and their own value functions f_1, f_2. We say that Π_1 and Π_2 are *dual* of each other if for every instance x they have equal optimal values, i.e. min{ $f_1(x,y)$ | y is a solution of instance x of Π_1 } = max{ $f_2(x,y)$ | y is a solution of instance x of Π_2 }.
Suppose that Π_1, Π_2 are *dual* problems as above.
1. Show that the following *Optimality Testing problem* is in NP$\cap$coNP for both Π_1, Π_2:
Input: Instance x, solution y.
Question: Is y an optimal solution for x?
2. Show that the decision version of both optimization problems is in NP$\cap$coNP.

(*Note:* Duality is an important property of many optimization problems. Examples include Linear Programming, Max Flow-Min Cut and a number of others.)