Design and Analysis of Algorithms
(CSCE 423/823, Spring 2005)
URL: `http://www.cse.unl.edu/~vinod/823S05/index.html`
1st April, 2005

Homework 5
Due on 15th April, 2005

Name :

Student ID :

Course No : **423** **823**

Instructions

- Write your name and student ID on the space provided. Circle the course number in which you are registered. Return this sheet with your solutions.
- Clarity of presentation is utmost important. You should give a clear description of all your algorithms with an analysis of their time complexity.
- No collaboration with any one is permitted for CORE problems!. See the Instructor or the TA if you need additional guidance. You may discuss approaches to solving ADVANCED problems with your classmates. However, you must write up your own solutions in your own words and acknowledge any one with whom you collaborated.

Core Problems

1. (9 Points) Let $\mathcal{P}$ denote the set of decision problems solvable in polynomial time, and $\mathcal{NP}$ denote the set of decision problems which can be verified in polynomial time (non-deterministic polynomial time). Suppose P_1 and P_2 are decision problems and P_1 is polynomial time reducible to P_2. That is, $P_1 \leq_p P_2$. For each of the following give "Yes/No/Not necessary" answer and justify.

   (a) If P_1 is in $\mathcal{P}$, is P_2 in $\mathcal{P}$?
   (b) If P_2 is in $\mathcal{P}$, is P_1 in $\mathcal{P}$?
   (c) If P_1 is $\mathcal{NP}$-complete, is P_2 $\mathcal{NP}$-complete?
   (d) If P_2 is $\mathcal{NP}$-complete, is P_1 $\mathcal{NP}$-complete?
   (e) If P_2 is polynomially transformable to P_1, are P_1 and P_2 both $\mathcal{NP}$-complete?
   (f) If P_1 and P_2 are $\mathcal{NP}$-complete, is P_2 polynomially transformable to P_1?
   (g) If P_1 is in $\mathcal{NP}$, is P_2 $\mathcal{NP}$-complete?
   (h) If P_1 is not in $\mathcal{NP}$, is P_1 not in $\mathcal{P}$?
   (i) If P_2 is not in $\mathcal{P}$, is P_2 not in $\mathcal{NP}$?

2. (7 Points) Show that the following problem known as Set Partition problem is $\mathcal{NP}$-Complete.

   Input: A set of numbers, X.
   Question: Is it possible to partition X in to subsets Y and Z, such that $Y \bigcup Z = X$ and $Y \cap Z = \emptyset$ so that summation of elements in Y equals that of elements in Z; that is $\Sigma_{a \in Y} a = \Sigma_{a \in Z} a$?

3. (7 Points) An independent set of $G = (V, E)$ is a subset V' of V such that G has no edge between any pair of vertices in V'. Consider the decision problem known as INDEPENDENT SET problem.

   Input: A graph $G(V, E)$ and an integer $k > 0$
   Question: Does G have an independent set of size $\geq k$?

   Prove that INDEPENDENT SET Problem is NP-Complete. (Use Clique problem described in the lecture to do so).

4. (5 Points) The SUBGRAPH ISOMORPHISM problem takes two graphs G_1 and G_2 and asks whether G_1 is isomorphic to a subgraph of G_2. Show that the subgraph-isomorphism problem is NP-complete.

5. (7 Points) Let $G = (V, E)$ be a directed graph. A set $F \subseteq V$ is a feedback vertex set if every cycle of G contains a vertex in F. The FEEDBACK VERTEX SET problem is defined as follows:

   Instance: Directed graph $G = (V, E)$ and positive integer k.
   Question: Does $G = (V, E)$ have a feedback vertex set with at most k vertices?

   Use the VERTEX COVER problem described in the class to show that FEEDBACK VERTEX SET problem is NP-complete.

ADVANCED PROBLEMS

6. (10 Points; Problem 34.5.3 of the text; page 1017). The integer linear-programming problem is like the 0-1 integer-programming problem, except that the values of the vector x may be any integers rather than just 0 and 1. Assuming that 0-1 integer-programming problem is NP-Complete, show that the integer linear-programming problem is NP-Complete.

7. (5 Points) The class Co-$\mathcal{NP}$ is the class of decision problems whose *complement* problems are in $\mathcal{NP}$. For example, consider the $\mathcal{NP}$ problem SAT whose complement problem is the decision problem $\overline{\text{SAT}}$ for which the "Yes" instances are those Boolean formulae which are *not* satisfiable. Identical to the definition of $\mathcal{NP}$-completeness we can define Co-$\mathcal{NP}$-completeness.

   Show that L is complete for NP if and only if $\overline{L}$ is complete for Co-NP.