Design and Analysis of Algorithms
Class Test 2

Time: 1 hr 30 min | November 8, 2012 | Marks: 30

Instructions:

1. While designing an algorithm, optimize it as much as you can.
2. Write all the algorithms clearly. Correct but unclear algorithms will carry only partial credit.
3. Write a *precise, clear* proof of correctness in at most 5 sentences each, and analyze the time and space complexity.
4. Whenever you are asked to justify a statement, the justification should not contain more than 3 sentences.

Questions:

1. State whether the following are true/false. Justify. **6 marks**
   (a) If there is a polynomial-time algorithm for an NP-hard problem, then there is a polynomial-time algorithm for all the problems in NP .
   (b) If Y is an NP-complete problem, $X \in$NP and $X \leq_P Y$, then X is NP-complete .
   (c) If $X \in$NP , then there is a polynomial-time verifiable certificate for all instances of X.
2. Recall that a *subset-sum* problem instance consists of a set $S = \{a_1, \ldots, a_n\}$ of positive integers, along with a target integer t. The goal is to determine whether there exists $S' \subseteq S$ such that $\sum_{i:a_i \in S'} a_i = t$.
   (a) Give an $O(nt)$ time dynamic programming algorithm for this problem. **4 marks**
   (b) State the conditions under which the $O(nt)$ time algorithm is considered to be a polynomial-time algorithm. Justify. **2 marks**
3. An instance of the *0/1 knapsack* problem consists of a set of items $\{i_1, \ldots, i_n\}$, with weights $\{w_1, \ldots, w_n\}$, and values $\{p_1, \ldots, p_n\}$ respectively. There is a knapsack of capacity T. You need to choose items to be put in the knapsack so as to get the maximum value without violating the knapsack capacity. (You can assume that all the numbers are positive integers.)
   (a) State the decision version of this problem. **1 mark**
   (b) Show that the decision version of this problem is NP-complete . **3 marks**
   (c) Can you modify the above NP-complete ness proof to show that the fractional knapsack problem is also NP-complete ? (Recall that in the fractional knapsack problem, you are allowed to pick fractions of items.) If yes, give the required modifications. If not, state the reason. **2 marks**
4. Suppose you are given a polynomial-time algorithm A which takes a Boolean formula ϕ as input and determines whether it is satisfiable. Give a polynomial-time algorithm that takes a Boolean formula ϕ as input, makes calls to A as a subroutine, and returns a satisfying assignment for ϕ. **4 marks**

5. Consider the bipartite matching problem. In the description of Hopcroft-Karp algorithm, we define an augmenting path P with respect to a matching M to be *a path that starts and ends at unmatched vertices, and alternates between unmatched and matched edges.*

   (a) Can an augmenting path P be non-simple? In other words, if the path P is allowed to be non simple, is $M \oplus P$ a matching in the given graph G? Justify your answer. **2 marks**

   (b) Is $M \oplus P$ a matching in G if G is non-bipartite and P is non-simple? **2 marks**

   (c) Let M_i be a matching obtained after i iterations of Hopcroft-Karp algorithm on a bipartite graph G. Let M^* be the matching when the algorithm terminates. State whether the following are true/false, and justify. **4 marks**

      i. $E(M_i) \subseteq E(M^*)$, where $E(M)$ is defined as the set of edges in a matching M.

      ii. $V(M_i) \subseteq V(M^*)$, where $V(M)$ is defined as the set of vertices that are matched in M.