Department of Mathematics
II Semester 2006-2006
MAL 342 Analysis and Design of Algorithms
Major Test Weightage 50%
Date 9.5.07 Time 6 P.M. – 8 P.M.

Q1. (i) Is it true that every prefix code is a Huffman code for appropriate values of the frequencies of the symbols? If yes prove it. If no, produce a counterexample. [2]

(ii) Let $f(n) = n^3 + 3n^2 + 5n + 6$. Use definition of big oh to show that $f(n) = O(n^3)$. [2]

(iii) Is it true that any two maximal independent sets in a matroid M have same cardinality? Justify. [2]

(iv) Show that every MST of a weighted connected graph G can be obtained by Kruskal's algorithm. [2]

(v) Does the MST problem exhibit Greedy choice property? Justify. [2]

Q2. Design a Greedy algorithm to find the maximum weight independent set in a weighted Matroid. Prove the correctness of your algorithm. What is the complexity of your algorithm? [2+4+2]

Q3 Design a dynamic programming algorithm for the all pair shortest path problem in a weighted graph such that the weight of an edge may be negative but the graph is free from negative cycles. Prove the correctness of your algorithm. Analyze the time complexity of your algorithm. [4+2+2]

Q4. Let S be a set of n elements. For any $x \in S$, the rank of x, r(x), is $|\{y \in S \mid y \leq x\}|$. Given S in the form of an array A and two numbers m_1 and m_2, find S_1, where $S_1 = \{y \mid m_1 \leq r(y) \leq m_2\}$ in O (n) time. [6]

Q5. Given a graph G with n vertices and m edges in adjacency list representation, design a linear time algorithm to test whether G has a cycle of odd length. [4]

Q6. Design a 2 approximation algorithm for minimum cardinality maximal matching problem. (A matching in a graph G=(V,E) is a subset E' of E such that no two edges in E' are adjacent.) [4]

Q7. Prove that 3-SAT is NP-Complete. [5]

Q8. A unit-time task is a job, such as a program to be run on a computer, that requires exactly one unit of time to complete. Given a finite set S of unit-time tasks, a Schedule for S is a permutation of S specifying the order in which these tasks are to be performed. The first task in the schedule begins at time 0 and finishes at time 1, the second task begins at time 1 and finishes at time 2, and so on. The problem of scheduling unit-time tasks with deadlines and penalties for a single processor has the following inputs:

1. A set $S=\{a_1,a_2,\ldots,a_n\}$ of n unit time tasks;
2. A set of n integer deadlines $d_1,d_2,\ldots,d_n$, such that each d_i satisfies $1 \leq i \leq n$ and task a_i is supposed to finish by time d_i; and
3. A set of n nonnegative weights or penalties $w_1,w_2,\ldots w_n$, such that we incur a penalty of w_i if task a_i is not finished by time d_i and we incur no penalty if a task finishes by its deadline.

The unit-time task scheduling problem is to find a schedule that minimizes penalty incurred for missed deadlines.

Model this problem as a maximum weight independent set problem in some appropriate weighted Matriod [5]