CSE 2012- Design and Analysis of Algorithms
Practice Problem Sheet (Greedy Algorithms)
Activity Selection Problem

Practice makes you Perfect

Activity Selection Problem :
Let $S = \{a_1, a_2, ..., a_n\}$ be the set of n proposed activities that uses the same resource (say, auditorium), which can serve only one activity at a time. Each activity has a start-time s_i and a finish-time f_i , $0 \leq s_i < \infty$. Any tow activities a_i, a_j are compatible if $s_i \geq f_j$ or $s_j \geq f_i$. The task is to select a maximum-size subset of mutually compatible activities from S.

1. In the 'Activity selection Problem' discussed in the class, we selected the activities that finish first. For this purpose, we even sorted the given activities in the order of their finish time. Instead, design a greedy algorithm for activity selection problem, by choosing the activity that starts last. Analyse the algorithm with the running-time and the time-complexity

2. Let $S = \{a_1, a_2, ..., a_n\}$ be the set of activities. Each activity a_i has the start-time(s_i), finish-time (f_i) and a value (v_i). Every a_i is associated with a triplet $< s_i, f_i, v_i >$. Any two activities $a_i =< s_i, f_i, v_i >$, $a_j =< s_j, f_j, v_j >$ are compatible if $s_i \geq f_j$ or $s_j \geq f_i$. Design an algorithm to choose a set A of compatible activities such that $\sum_{a_k \in A} v_k$ is maximum. Analyse the algorithm with the running-time and the time-complexity

3. Suppose we have a set S of activities (with the start-time and the finish-time, as in the activity selection problem) that has to be scheduled among a large number of elcture halls, where any activity can take place in any lecture hall. The task is to schedule all the activities using as few lecture halls. We call this problem as 'Extended activity selection problem'. Given the set of activities and the detials of the lecture halls (say, $l_1, l_2, ...l_m$), Design a greedy algorithm to determine which activity should use which lecture hall. Analyse the algorithm with the running-time and the time-complexity

4. Design a dynamic programming algorithm for the activity selection problem by using the recurrence relation developed for the greedy approach (which was discussed in the class) and compare the dynamic programming algorithm with that of the greedy algorithm.

5. There are n items in a store. The price of i-th item is Rs.v_i and the weight of the i-th item is w_i Kg. Here w_i and w_i are integers. One wishes to take as many items as one could, from the store in a knapsack. Task is here to store the items in the knapsack such that total value of all the the items selected, is maximum and the total weights of all the items chosen is less than or equal to W, where W capacity of the knapsack. Thisproblem is called $0-1$ knapsack problem. Design a greedy algorithm for the $0-1$ knapsack problem. Analyse the algorithm with the running-time and the time-complexity.