ESO207: Data Structures and Algorithms (Practice Problems – III)

Question 1: Another scheduling problem

There are n jobs. There are n PCs and one supercomputer. Each job consists of two stages: first it needs to be preprocessed on the supercomputer, and then it needs to be finished on one of the PCs. Let us say that job J_i needs p_i seconds of time on the supercomputer, followed by f_i seconds of time on a PC. Since there are n PCs available, the finishing of the jobs can be performed in parallel – all jobs can be processed at the same time on their respective PCs. However, the supercomputer can only work on a single job at a time. So we need to find out the order in which to feed the jobs to the supercomputer. Our aim is to minimize the completion time of the schedule which is defined as the earliest time at which all jobs will have finished processing on the PCs.

Design a greedy strategy/algorithm that finds the order in which the jobs should be scheduled on the supercomputer so that completion time achieved is as small as possible.

Question 2: Subsequence detection

Sequence A is said to be subsequence of another sequence B if there is a way to delete zero or more elements from A so that the resulting sequence turns out to be B. For example $\langle a, b, a, a, c, b, d\rangle$ is a subsequence of $\langle c, a, a, b, a, b, a, b, c, a, c, b, b, d\rangle$. Let S and $S?$ be two sequences of characters. Let m and n be respectively the lengths of sequences S and $S?$. You may assume that S and $S?$ are given in the form of arrays. Getting inspiration from greedy approach, design an $O(m+n)$ time algorithm to detect whether S is a subsequence of $S?$. Prove correctness of the algorithm as well.

Question 3: Efficient construction of committee

The manager of a large student union on campus comes to you with the following problem. She is in charge of a group of n students, each of whom is scheduled to work one shift during the week. There are different jobs associated with these shifts (tending the main desk, helping with package delivery, rebooting cranky information kiosks, etc.), but we can view each shift as a single contiguous interval of time. There can be multiple shifts going on at once. She is trying to choose a subset of these n students to form a supervising committee that she can meet once a week. She considers such a committee to be complete if for every student not on the committee, that studentÕs shift overlaps (at least partially) the shift of some student who is on the committee. In this way, each studentÕs performance can be observed by at least one person who is serving on the committee.

Give an efficient algorithm that takes the schedule of n shifts and produces a complete supervising committee containing as few students as possible. You must also prove the correctness of the algorithm.

Question 4: MST with changing edge weights

There is a connected graph $G = (V, E)$, with edge costs that are all distinct. You are given a minimum spanning tree T for G. Let v be a leaf node in T. Weight of one of the edges incident on v in the graph has decreased. This might lead to update T. You have to determine if T has to be updated, and in case, the MST is updated, you have to compute the new minimum spanning tree in $O(n)$ time only. (Note that there could be $O(n)$ edges incident on v in the original graph.)

Question 5: Exploring a simple hash function

Consider the hash function $h(i) = i \mod n$. Assume the universe size, $m > n^2$.

(a) Describe a set $S \subset U$ of size n for which the function h is too bad: all elements of S are hashed into the same location in the hash table.

(b) Describe a set $S \subset U$ of size n for which the function h is perfect: all elements of S are hashed into the different locations in the hash table.

Question 6: Longest Common Subsequence

A string $B = b_1 b_2 \dots b_k$ is said to be a subsequence of a string $A = a_1 a_2 \dots a_n$ if B can be obtained from A by removing zero or more characters from A.

Design a dynamic programming solution that takes as input two strings X and Y (need not have the same length) and outputs the longest common subsequence (LCS) between X and Y. If there are multiple such LCSes then your algorithm should output any one of them. What is the time complexity of your algorithm?

Question 7: Coin Minimization Problem

You are given unlimited supply of coins of denomination 1, 5 and 7. Given a positive integer V design a dynamic programming solution to give change for an amount of V using the minimum number of coins possible. What is the time complexity of your algorithm?