CSDS 310: Algorithms Fall 2024

Assignment 4: Greedy Algorithms

Instructor: Orhan Özgüner Due: November 20 before 11:59 PM

Problem 1

We started activity selection problem in class. Prove that the following greedy choices do not lead to optimal solutions for the activity selection problem:

(a) Select the activity with the earliest starting time.

(b) Select the activity with least duration.

(c) Select the activity that overlaps the fewest other remaining activities.

Problem 2

You are given two sets A And B, each containing n positive integers. You can choose to reorder each set however you like. After reordering let a_i be the i^{th} element of set A and b_i the i^{th} element of set B. You then receive a payout of

$$\prod_{i=0}^{n} a_i^{b_i} \tag{1}$$

Give an algorithm that maximizes your profit.

Problem 3

We have n activities. Each activity requires t_i time to complete and has deadline d_i. We would like to schedule the activities to minimize the maximum delay in completing any activity; that is, we would like to assign starting times s_i to all activities so that $\max_{1 \le i \le n}\{\Delta_i\}$ is minimized, where $\Delta_i = f_i - d_i$ is the delay for activity i and $f_i = s_i + t_i$ is the finishing time for activity i. Note that we can only perform one activity at a given time (if activity i starts at time s_i, the next scheduled activity has to start at time f_i).

For example, if $t =< 10, 5, 6, 2 >$ and $d =< 11, 6, 12, 20 >$, then the optimal solution is to schedule the activities in the order $< 2, 1, 3, 4 >$ to obtain starting/finishing times $s/f =< 5/15, 0/5, 15/21, 21/23 >$ and achieve a maximum delay of 9 (for the third activity).

Give an algorithm that minimizes the maximum delay.

Problem 4: (Bonus question: 10 pts)

We have infinite supply of integer coin denominations of $c_1 = 1 < c_2 < ... < c_k$ to make change for a given an integer amount n. For this purpose, we would like to find the minimum number of coins that add up to n. An obvious greedy choice for this problem is to use the largest coin that has value less than or equal to n (e.g., if $c_k \leq n$, then return c_k, and solve the problem for $n - c_k$).

(a) Prove that, if the coin denominations are arbitrary, this greedy choice is not guaranteed to lead to an optimal solution. (Just prove the greedy choice, no pseudocode or run time)

(b) Prove that, if the coin denominations are powers of 2, i.e., $c_i = 2^{i-1}$ for $1 \leq i \leq k$, then this greedy choice is guaranteed to lead to an optimal solution. (Just prove the greedy choice, no pseudocode or run time)