

Homework 10 – Due Friday, April 27, 2007

Reminder

- Collaboration is permitted, but you must write the solutions by yourself without assistance, and be ready to explain them orally to a member of the course staff if asked. You must also identify your collaborators. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.
- To facilitate grading, please write down your solution to each problem on a separate sheet of paper. Make sure to include all identifying information and your collaborators on each sheet. Your solutions to different problems will be graded separately, possibly by different people, and returned to you independently of each other.

Reading and Exercises Section 16.1 of CLRS. Do problem 16-4(a).

Problems to be handed in

1. **(Short Answers)** For each of the following give a short answer (at most a few sentences.)
 - (a) Show that if all the weights in a graph are distinct (that is, no two edges have exactly the same weight), then there is only one minimum spanning tree. *Hint:* Use Theorem 23.1 in CLRS to show that the output of Prim’s algorithm is the only possible MST.
 - (b) Suppose we modify a weighted graph G with positive distinct edge weights $\{w(e) : e \in E\}$ to create a graph G' which has the same vertices and edges, but where the weights have been squared, that is, we set $w'(e) = w(e)^2$. For each of the following statements, either give a short proof of the statement, or a counter-example which shows that it is false:
 - i. The minimum spanning tree of G' is the same as the minimum spanning tree of G .
 - ii. Shortest paths between all pairs of vertices are the same in G and G' .
 - (c) Do Exercise 16-1.2 from CLRS. (*Note:* what they call the “Activity Selection” problem is identical to the “Interval Scheduling” problem discussed in the lecture of Friday, April 20.)
 - (d) In the lecture of Monday, April 23, we discuss the *Maximum Lateness* scheduling problem. Give examples of inputs for this problem where the following greedy algorithms return a wrong (that is, non-optimal) solution:
 - i. Schedule the jobs in increasing order of duration (from shortest to longest).
 - ii. Schedule the jobs in increasing order of *slack time*. The slack time of a job is $d_i - t_i$, the difference between its deadline and the time it takes to do the job.
2. **(Scheduling in many rooms)** You are asked to help Penn State with scheduling classes. For each lecture i , you are given a start time $S(i)$ and a finish time $F(i)$. Your goal is to schedule *all* lectures, using as few rooms as possible. Each room can accommodate one lecture at a time. You do not need to schedule breaks between the lectures.

- (a) Define the *depth* d of the input to be the maximum number of lectures that are simultaneously scheduled at any point in time. Prove that at least d rooms are needed.
 - (b) Give an efficient greedy algorithm that solves the scheduling problem by assigning each lecture to one of d rooms labeled $1, 2, \dots, d$.
 - (c) Argue that your algorithm finds an optimal schedule. (Explain why each lecture gets assigned a room, and no conflicting lectures are ever assigned to the same room. Why is the algorithm using the optimal number of rooms?)
 - (d) Analyze the running time of your algorithm.
3. **(Hike Planning)** You are planning a multi-day hike along a trail that is L miles long. You can hike up to d miles a day. Campgrounds are located at distances x_1, x_2, \dots, x_n from the start of the trail. We say that a set of campgrounds is *valid* if the distance between each adjacent pair is at most d miles, the first one is at most d miles from the start of the trail, and the last one is at most d miles from the end of the trail (in other words, you can make it across the whole trail, stopping only at these campgrounds). Assume that the full set of campgrounds is valid.
- (a) Give a greedy algorithm for finding a minimum valid set of campgrounds.
 - (b) Use the “greedy stays ahead” strategy to prove that your algorithm produces an optimal solution.
 - (c) Analyze the running time of your algorithm.
4. **(Giving Change)** Recall the cashier’s problem discussed in class: give a specified amount of change using the smallest possible number of coins.
- (a) Give pseudocode for the greedy algorithm given in class that takes an integer amount a and coin denominations d_1, \dots, d_n , and returns a set of coins whose value is a .
 - (b) Suppose that coins come in denominations of 1, 5, 25 and 100. Use the “exchange” strategy to prove that the greedy algorithm finds the smallest possible set of coins.
 - (c) Now suppose that the coins come in denominations of 1, 5, 10 and 25 and 100 (as in the US). Prove that the greedy algorithm still returns the smallest possible set of coins (an exchange argument also works here but it is a touch more complicated).
- 5*. **(Extra credit)** We consider a variant of the scheduling task from Problem 4 from Homework 9. Suppose that you run a photocopy business with a single large machine. You’re faced with a list of n jobs to perform, and you want to order them so as to make your customers as happy as possible. Each job comes a duration t_i (the time it takes to complete the job) and a positive weight w_i representing the importance of the customer who ordered the job. Let C_i be the time at which job i is completed (so C_i is the sum of t_i together with the durations of all the jobs run before job i). You want to minimize the weighted sum of the finishing times,

$$\sum_{i=1}^n w_i C_i.$$

- (a) (Warm-up, do not hand in) Show that ordering the jobs by either increasing duration or decreasing importance does not necessarily minimize the weighted sum.
- (b) Give a greedy algorithm that outputs the ordering with the minimum weighted sum and prove that it is correct.