Algorithms (CS:3330:0001 or 22C:031:001)
Homework 4

The homework is based on our discussion of greedy algorithms (Section 4.1 and 4.2) and has four questions, each of which counts for the same number of points. In questions 1, 2, and 4, I have not asked for a justification for the correctness of the algorithm you propose. Nevertheless, please think about why you believe your algorithms works correctly on every input. This is a key aspect of working with greedy algorithms.

1. Suppose that we are given a pattern S', which is a sequence of m items, and a much bigger data sequence S consisting of n items. A given item can occur multiple times in both the pattern and the data sequence. We say that S' is a *subsequence* of S if there is a way to delete certain items from S so that the remaining items, in order, are equal to the sequence S'. For example the sequence of items $\langle Y, E, Y, O \rangle$ is a subsequence of the sequence $\langle A, Y, E, Y, Y, O \rangle$, but is not a subsequence of $\langle A, Y, E, A, O, Y \rangle$.

   Give an efficient (polynomial time) algorithm to determine if a given sequence S' is a subsequence of given sequence S.

2. We are given a set of points $P = \{p_1, p_2, \ldots, p_n\}$ on the real line and a set I of intervals on the real line. For example,

   $$P = \{2, 3, 5, 8, 11, 14\} \text{ and } I = \{[1, 3], [2, 6], [3, 12], [7, 14], [14, 15].$$

   A subset $J \subseteq I$ of intervals is said to *cover* P if each point in P belongs to at least one interval in J. For example, $\{[1, 3], [3, 12]\}$ does not cover P since the point $14 \in P$ is not contained in either interval. On the other hand, $\{[1, 3], [3, 12], [14, 15]\}$ covers P. Describe a polynomial time algorithm that takes as input P and I and returns a minimum size subset of I that covers P.

   The subset $\{[1, 2], [3, 12], [14, 15]\}$ has size 3. The minimum size cover in the example is $\{[2, 6], [7, 14]\}$ with size 2.

3. In the process of finding a correct greedy algorithm for the interval scheduling problem (Section 4.1), we also showed that several other rather natural looking greedy algorithms do not always work. However, some nice things can be said about some of these algorithms.

   Let us revisit one of these greedy algorithms: Pick the interval I that overlaps with the least number of intervals, remove I and all intervals that overlap with I, and repeat this process till no more intervals remain.

   For this greedy algorithm, argue that the number of intervals picked is at least as large as half the size of the optimal solution.

   Hint: Let us recall the analysis of the greedy algorithm that picks the interval with the earliest finish time, rephrasing it slightly so that it suggests a direction for the argument in the present question. Suppose that an optimal solution consists of the intervals $o_1, o_2, \ldots, o_m$. The analysis works by claiming that after the first iteration, when the greedy algorithm has picked its first interval and removed incompatible intervals, *at most one* of the intervals in the optimal solution has been discarded, so *at least* $m-1$ of the intervals in the optimal solution are still in the set of intervals that have not been discarded. After the first two iterations, *at most two* of the intervals in the optimal solution have been discarded, so *at least* $m-2$ of the intervals in the optimal solution are still in the set of intervals that have not been discarded.

For the algorithm considered in this problem, suppose the first interval we pick is I. How many intervals in an optimal solution can be discarded when we remove I and all intervals that overlap with I?

4. A company is interviewing n candidates on a particular day. Each candidate needs to first have a face-to-face meeting with a panel, and then take a written exam. Since the candidates are from different backgrounds, the company has planned meetings and exams of different lengths for the candidates. Let m_i and e_i denote the lengths, in minutes, of the face-to-face meeting and the written exam, respectively, for candidate i.

   (We are assuming that the candidates are $\{1, 2, \ldots, n\}$.)

   Since there is only one panel, the face-to-face meetings cannot overlap. However, the written exams can be taken simultaneously by any number of candidates. So the company decides to schedule the face-to-face meetings in a certain order starting at 8:00 am. Each candidate proceeds to take her written exam immediately after her meeting.

   Describe a polynomial time algorithm that takes as input the m_i and the e_i values for each of the n candidates and computes an ordering of the candidates (for the face-to-face meetings) that results in the entire process completing as early as possible. (The process is said to complete when every candidate has finished their written exam.)

The homework is due in class on Monday, October 20. If you can't make it to class, drop it in my mailbox in the MacLean Hall mailroom.

Let's review some policy from the first day handout. "For homework problems, collaboration is allowed, assuming each of you has first spent some time (about 30 minutes) working on the problem yourself. However, no written transcript (electronic or otherwise) of the collaborative discussion should be taken from the discussion by any participant. Furthermore, discussing ideas is okay but viewing solutions of others is not. It will be assumed that each of you is capable of orally explaining the solution that you turn in, so do not turn in something you don't understand. Students are responsible for understanding this policy; if you have questions, ask for clarification."

Please note that although the correctness of your solution counts for a significant fraction of your score, the quality of your writing and a demonstration that you understood the question and made a serious attempt at solving it also count for a significant fraction.