CSE Ph.D. Qualifying Exam, Spring 2022
Algorithms

Instructions:

Please answer three of the following four questions. All questions are graded on a scale of 10. If you answer all four, all answers will be graded and the three lowest scores will be used in computing your total.

Questions:

1. **Greedy**

   You are given n distinct points and one line l on the plane and some constant $r > 0$. Each of the n points is within distance at most r of line l (as measured along the perpendicular). You are to place disks of radius r centered along line l such that every one of the n points lies within at least one disk. Devise a greedy algorithm that runs in $O(n \log n)$ time and uses a minimum number of disks to cover all n points; prove its optimality. You can use the "greedy stays ahead" argument.

2. **Dynamic Programming**

   Let A be the set of all integers in the range of 1 to n. For each pair of numbers in A, denoted as (i, j), where $1 \leq i \leq j \leq n$, a cost function $C[i, j] > 0$ is defined and given. The task is to find an increasing sequence of cutpoints $i_1, i_2, \ldots, i_k \in \{1, 2, \ldots, n-1\}$ to minimize the total cost $\sum_{t=0}^{k} C[i_t + 1, i_{t+1}]$, where $i_0 = 0$ and $i_{k+1} = n$.

   In other words, you need to partition the sequence $\{1, 2, \ldots, n\}$ into $k + 1$ segments, where the t^{th} segment ends with number i_t, and you want to find the best way of partitioning such that the total cost is minimized. Note that k is not fixed and it depends on the solution you find.

   Explain a Dynamic Programming algorithm to find the minimum cost of the segmentation and provide the recurrence (including the base case(s)). Analyze the time and space complexity.

3. **Dynamic Programming**

   You are taking a course this semester which has n weeks. Each week, there are two assignments released, an "easy" assignment and a "difficult" assignment. You can choose from one of them but you can not choose both during the same week. At week i, accomplishing the easy assignment will earn you e_i points, and the difficult assignment will earn you d_i points. However, if you plan to do a difficult assignment in week i, you must do no assignment in week $i - 1$, because you will need the time of week $i - 1$ to study and prepare for the difficult assignment in week i. On the other hand, you can take an easy assignment in week i no matter what assignment you do in week $i - 1$.

   Given a sequence of e_i values for the points one can gain through the easy assignments, and a sequence of d_i values for the points one can gain through the difficult assignments, $1 \leq i \leq n$, you need to come up with a plan for each week, in order to maximize the total points you gain for the semester. Note that when you devote a week to work on an assignment you always get full credits, that is, getting partial credits is not a scenario to consider. Therefore, for each week i, there are three choices: 1) take an easy assignment and earn e_i points; 2) take an difficult assignment and earn d_i points; 3) take no assignment and earn 0 points.

Please design a dynamic programming algorithm that finds the optimal total points and the optimal weekly plan. Please describe the algorithm clearly (you may give the pseudocode), give the recurrence relations, and analyze the time and space complexity of your algorithm.

4. **NP: Dinner with Frenemies**

   Prove that the following decision problem is NP-complete. Given n students and a set of pairs of students who are enemies, is it possible to arrange a dinner around a round table so that two enemies do not sit side by side? Remember to include all the steps of the NP-completeness proof.