CSE Ph.D. Qualifying Exam, Spring 2023
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**

   Assume you will drive from city A to city B along a highway and you wish to minimize the time you spend on adding gas to your tank. The following quantities are given to you:

   - the capacity C of your gas tank in liters
   - the rate F of fuel consumption in liters/mile
   - the rate r in liters/minute at which you can fill your tank at a gas station. r is the same for all gas stations.
   - the distances of the n gas stations from your start point $x_1, x_2, \ldots, x_n$ along the highway. We know $x_1 = 0$, $x_1 < x_2 < ... < x_n$ and $x_i - x_{i-1} \leq C/F$ for $2 \leq i \leq n$.

   You will start with an empty tank. For example, if you stop to fill your tank from 2 liters to 8 liters, you would have to stop for $6/r$ minutes. The goal is to minimize the total time you stop for gas.

   Consider the following two algorithms:

   (a) Stop at every gas station, and fill the tank with just enough gas to make it to the next gas station.

   (b) Stop if and only if you don't have enough gas to make it to the next gas station, and if you stop, fill the tank up all the way.

   For each algorithm either prove or disprove that this algorithm solves the problem optimally. Your proof of correctness can use an exchange argument.

2. **Dynamic Programming**

   This problem is to determine the set of states with the smallest total population that can provide the votes to win the election.

   Formally, the problem is: We are given a list of states $\{1, ..., n\}$ where each state i has population p_i, and v_i, which is the number of electoral votes for state i. All electoral votes of a state go to a single candidate. The overall winning candidate is the one who receives at least V electoral votes, where $V = (\sum_i v_i)/2 + 1$. Our goal is to find a set of states S that minimizes the value of $\sum_{i \in S} p_i$ subject to the constraint that $\sum_{i \in S} v_i \geq V$.

   Please design a dynamic programming algorithm for this problem. Define the subproblem, give the recurrence relations, and analyze the time and space complexity of your algorithm. Remember to include steps to output the optimal set of states (you do not need to output all the optimal solutions if there are more than one.)

3. **Dynamic Programming: merging two sequences**

   Let $X = \{x_1, x_2, \ldots, x_m\}$ and $Y = \{y_1, y_2, \ldots, y_n\}$ be two genomic sequence represented by strings of letters and $C[i, j]$ be the cost function defined on pairs of letters, one from X and one from Y.

   Our task is to merge these two sequences to create a new genomic sequence Z and we need to find the cheapest merge of X and Y, while maintaining the order of letters from both X and Y. Therefore, for instance, if $X = \{a, b, a, c\}$ and $Y = \{d, a, e, b\}$, then $Z = \{a, d, a, b, a, e, c, b\}$ and $Z = \{a, d, a, e, b, b, a, c\}$ are valid merges, but $Z = \{a, b, a, e, c, d, a, b\}$ is not because e from the second sequence is used before d and a.

   Total cost of the merge is the sum of the merging costs of the adjacent letters from different sequence. So if Z includes x_i and x_{i+1} as consecutive letters, there is no cost, but if Z has $x_s y_t$ as consecutive letters, then there is a cost $C[s, t]$ and it should be added to the total cost of the merge. *Note: You can assume that the cost function is symmetric.*

   Please design a dynamic programming algorithm to find the minimum cost of merging X and Y. Please define the subproblem(s) and give the recurrence relations. Analyze the time and space complexity of your algorithm. Backtracing step and pseudocode are not required.

4. **NP-complete**

   The Degree-Constrained-ST problem is defined as follows: given an undirected, unweighted graph $G = (V, E)$, does there exist a spanning tree of this graph where each node in the spanning tree has a degree of at most k?

   Prove that this problem is NP-Complete using the statement that the HamPath problem is NP-Complete.

   In the HamPath problem, we are given a connected graph G, and are asked whether it contains a simple path that visits all vertices of the graph (the path should visit each vertex exactly once).

   Please remember to include all steps of the NP-completeness proof.