CSE Ph.D. Qualifying Exam Questions 2024 Spring – Algorithms

This exam is closed-book, closed-notes.

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.

1. **Greedy.** There are n cities in a state. Given any two cities x and y, you know the cost of building a road between x and y. You are required to build a network of roads such that any city can be reached from any other city (through a direct road or via intermediate cities). You need to do this at minimum cost.
   Describe a greedy algorithm to solve the task and prove its correctness. What is the running time of your algorithm?

2. **Dynamic Programming.** A number of languages (including Chinese and Japanese) are written without spaces between words. You are given text in such a language, and you are required to design an algorithm to infer likely boundaries between consecutive words in the text. If English were written without spaces, the analogous problem would consist of taking a string like "meetateight" and deciding that the best segmentation is "meet at eight" (and not "me et at eight", or "meet ate ight", or any of a huge number of possibilities).
   To solve the problem, you are given a function Quality() that takes any string of letters and returns a number that indicates the quality of the word formed by the string. A high number indicates that the string resembles a word in the language (e.g. 'meet'), whereas a low number means that the string does not resemble a word (e.g. 'eeta'). The total quality of a segmentation is determined by adding up the qualities of each of its words. Design a bottom-up dynamic programming algorithm that take a string y and computes a segmentation of maximum total quality. You are required to provide the recurrence relation, pseudocode and running time analysis (you can treat the call to Quality as a single computational step, O(1)).

3. **Dynamic Programming.** Suppose you are choosing a water supplier for your house. For each of the next n weeks, you will need s_i tons of

water, which have to be supplied by a water supplier. Each week's water supply can be provided by **only** one of two water companies, A or B.

- Company A charges a fixed rate r per ton (so it costs $r \times s_i$ to provide a week's water demand s_i).
- Company B makes contracts for a fixed cost c per week, no matter how many tons are ordered. However, contracts with company B must be made in blocks of four consecutive weeks at a time.

A schedule, for your house, is a choice of water supply company (A or B) for each of the n weeks, with the restriction that company B, whenever it is chosen, must be chosen for blocks of four contiguous weeks at a time. The cost of the schedule is the total amount paid to companies A and B, according to the description above. Design a dynamic programming algorithm that takes a sequence of water values $s_1, s_2, ..., s_n$ and returns a schedule of minimum cost.

Please include the following in your answer:

- Give the recurrence relations (do not forget the base cases).
- Give the pseudocode of the top-down implementation.
- Analyze time and space complexity.

4. **NP-Completeness.**
The GraphValue Problem is defined as following:

- Input: An undirected graph G with n vertices, a list L of n nonnegative integers, and a bound b.
- Question: Is there a way to assign the numbers in L to the vertices of G such that (1) each number is assigned to exactly one vertex, and (2) the graph's value is not greater than b? The value of the graph is the sum of all the edge values; each edge value is the product of the numbers assigned to its endpoint vertices.

Prove that GraphValue is NP-complete by using the fact the Vertex-Cover is NP-complete. VertexCover(G, k) is the problem of deciding whether in graph G there exists a subset of vertices S of size at most k such that all edges in G have at least one endpoint in the selected vertex set S. Remember to include all the steps of the NP-completeness proof.

Hint: when constructing a GraphValue instance from a VertexCover instance, consider assigning some vertices value 0 and setting b to 0.