CS3510-A: Design and Analysis of Algorithms, Spring 2021

Homework-8

March 7, 2021

DUE DATE: Wednesday, March 17, 11:59pm

Note-1: Your homework solutions should be electronically formatted as a single PDF document that you will upload on Gradescope. If you have to include some handwritten parts, please make sure that they are very clearly written and that you include them as high resolution images.

Note-2: Please think twice before you copy a solution from another student or resource (book, web site, etc). It is not worth the risk and embarrassment.

Note-3: You need to **explain/justify** your answers. Do not expect full credit if you just state the correct answer. This includes their correctness and runtime.

Note-4: You will get 2 extra points if you submit electronically typed solutions instead of hand-written.

Problem-1 (30 points)

Using Huffman encoding of n symbols with frequencies $f_1, f_2, ...f_n$, what is the length of the longest possible code-word? Additionally, what values of $f_1, f_2, ...f_n$ lead to this length? Please justify why these values of f_i lead to the maximum possible length code-word.

Problem-2 (35 points)

Suppose you're consulting for a company that manufactures PC equipment and ships it to distributors all over the country. For each of the next n weeks, they have a projected supply s_i of equipment (measured in pounds), which has to be shipped by an air freight carrier.
Each week's supply can be carried by one of two air freight companies, A or B.

- Company A charges a fixed rate r per pound (so it costs $r * s_i$ to ship a week's supply s_i).
- Company B makes contracts for a fixed amount c per week, independent of the weight. However, contracts with company B must be made in blocks of four consecutive weeks at a time.

A schedule, for the PC company, is a choice of air freight 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 company A and B, according to the description above.

Example: Suppose $r = 1, c = 10$, and the sequence of values is 11,9,9,12,12,12,12,9,9,11. Then the optimal schedule would be to choose company A for the first three weeks, then company B for a block of four consecutive weeks, and then company A for the final three weeks.

Give a **polynomial-time** algorithm that takes a sequence of supply values $s_1, s_2, ..., s_n$, which represent the supply for n weeks, and returns a schedule of minimum cost. Note that you must not only return the **minimum cost** itself, but also the **schedule**.

Please provide your answer in four sections:

1. Dynamic Programming Table: Must state the dimensions of the table, and explain what each element of this table represents.
2. Recurrence Equation: Express mathematically each value in the table as a function of previous values in the table (do not forget to write down the equation(s) for the base case of the recursion).
3. Pseudo-code: Construct your algorithm: takes $s_1, s_2, ..., s_n$, n, r, c as input and it should return the min cost as well as the schedule.
4. Run-Time Analysis: Explain why your algorithm is polynomial time.

Problem-3 (35 points)

Recall that in the basic Load Balancing Problem, we're interested in placing jobs on machines so as to minimize the makespan— the maximum load on any one machine. In a number of applications, it is natural to consider cases in which you have access to machines with different amounts of processing power, so that a given job may complete more quickly on one of your machines than on another. The question then becomes: How should you allocate jobs to machines in these more heterogeneous systems?

Here's a basic model that exposes these issues. Suppose you have a system that consists of m slow machines and k fast machines. The fast machines can perform twice as much work per unit time as the slow machines. Now you're given a set of n jobs; job i takes time t_i to process on a slow machine and time $(1/2)t_i$ to process on a fast machine. You want to assign each job to a machine; as before, the goal is to minimize the makespan – that is the maximum, over all machines, of the total processing time of jobs assigned to that machine.

Describe a **polynomial-time algorithm** that produces an assignment of jobs to machines with a makespan that is at most three times the optimum. Your solution should be of the form:

- Algorithm and runtime: Briefly describe how you will assign tasks to machines – and explain why your algorithm can run in poly-time. (This part can be done with 1-2 sentences only!)
- Proof of Upper Bound: Show why your greedy solution can have a makespan that is at most 3 times the optimal solution's makespan.