CMPSC 465 Data Structures & Algorithms
Fall 2024 Mingfu Shao and Debarati Das

HW 12

Due November 22th (Friday), 11:59 pm

Formatting: Each problem should begin on a new page. When submitting in Gradescope, try to assign pages to problems from the rubric as much as you can. Make sure you write all your group members' names. For the full policy on assignments, consult the syllabus.

1. (15 pts.)

   Given a pair of strings $X = X_1X_2 \dots X_m$, $Y = Y_1Y_2 \dots Y_n$ their longest common subsequence denoted by $LCS(X, Y)$, is the longest subsequence that is present in both X and Y. Write a poly time dynamic program algorithm to compute the length of $LCS(X, Y)$.

2. (15 pts.)

   Suppose you have a large piece of gold foil and you need to plan on how to cut it to make the maximum profit. The size of the gold foil is $M \times N$ where M and N are positive integers. Due to technical limitations, you can only cut gold foil either horizontally or vertically. Each cut results in two smaller rectangle pieces with integral sizes, i.e. the sizes of the smaller pieces are still integers. You can make as many cuts as you want as long as sizes permit.

   When you are done cutting, you can sell your rectangles at the market. There are n buyers at the market. The i^{th} buyer is willing to pay p_i for a rectangle of size $a_i \times b_i$. You can sell as many rectangles to a buyer as you want, including not selling any.

   Design a dynamic programming algorithm that outputs the maximum profit on a piece of gold foil of size $M \times N$. What is the running time?

3. (20 pts.)

   You are given a checkboard that has 4 rows and n columns and has an integer written in each square. You are also given a set of $2n$ pebbles and want to place some or all of these on the checkboard (each pebble can be placed on exactly one square) so as to maximize the sum of the integers that are covered by the pebbles.

   There is one constraint: for a placement of pebbles: for a placement of pebbles to be legal, no two of them can be horizontally or vertically adjacent squares (diagonal adjacency is fine).

   1. Determine the number of legal patterns that can occur in any column and describe these patterns. Call two patterns compatible if they can be placed on adjacent columns to form a legal placement. Let us consider subproblems consisting of the first k columns $1 \leq k \leq n$. Each subproblem can be assigned a type which is the pattern occurring in the last column.
   2. Using the notations of compatibility and type, give an $O(n)$ time dynamic programming algorithm for computing an optimal placement.