CSCI-GA.1170-003/004 Fundamental Algorithms April 3, 2019

Problem Set 8

Lecturer: Yevgeniy Dodis *Due: Thursday, April 11*

Problem 8-1 (Ponyless Zone) 12 points

You have an $n \times n$ square field. Unfortunetely, in some of the squares there is a beautiful little pony. You have two friends Jerry and Elaine who do not like ponies, so you want to find some (possibly) smaller square included in the original field that is completely ponyless.

More formally, you are given an $n \times n$ matrix A of bits, where $\mathrm{A}[i][j] = 1$ iff there is a pony in a square at position (i, j). Find an efficient dynamic programming algorithm for finding the biggest possible ponyless sub-square in your field.

Define an auxiliary $n \times n$ matrix B such that $B[i][j]$ memorizes the biggest possible ponyless square with the bottom-right corner at position (i, j).

(a) (4 points) Show a recursive relationship between $B[i][j]$ and $B[i-1][j-1]$. Based on this, find $O(n^3)$ dynamic programming algorithm for computing the matrix B.

(b) (7 points) Show a recursive relationship between $B[i][j]$ and $(B[i-1][j], B[i][j-1])$. Based on this, find a faster $O(n^2)$ dynamic programming for computing the matrix B.

(c) (1 point) Explain how to give the final solution to the original problem, assuming you already comptuted B. What is the final time complexity for the best final solution?

Problem 8-2 (Let's paint a fence!) 12 (+3) Points

We just built a new fence which is made up of many boards, and our task is to paint the fence efficiently. To paint the fence, you have to paint N boards that make up the fence. The lengths of the N boards are $\{L_1, L_2, \ldots, L_N\}$. You have hired K painters and you know that each painter takes 1 hour to paint 1 unit of board. If $3, 4, 5$ are the boards painter i paints, the total time he spends is $t_i = L_3 + L_4 + L_5$. Our goal is to assign each painter to boards so that the total painting time is minimized. Since the painters can work in parallel, the painting time is minimized when $\max(t_1, \ldots, t_K)$ is minimal. The painting task must be accomplished under the following constraints:

1. Two painters cannot share a board to paint. That is, a board cannot be painted partially by one painter, and partially by another. All L_i units of board i must be painted by one painter.

2. Any painter will only paint contiguous boards. For example, a configuration where painter 1 paints boards 1 and 3 but not 2 is not a valid solution.

You are given as input the following: K, the number of painters, and L, a list which will represent the length of each board, where L_i is the length of the i^{th} board. In the following problems, denote by $T[i, j]$ the minimum time to paint the first i boards with j painters. We will, successively in the following subtasks, come up with a procedure to minimize painting time.

(a) (6 points) Write a recurrence for $T[i, j]$ in terms of $T[*, j-1]$.

(b) Note that the time required to fill table T is $O(NK \cdot C)$, where C is the maximum time needed to compute any of the entries $T[i, j]$. Provide the fastest algorithm you can conceive. You will obtain

- 2 points if your algorithm satisfies $C = O(N^2)$,
- an additional 4 points if $C = O(N)$,
- and 3 points of **extra credit** if you manage to get it down to $C = O(\log N)$.

Problem 8-3 (Palindrome) 15 Points

A *palindrome* is a sequence that remains the same if written in the reverse order, e.g. DEED, DENNISSINNED, BORROWORROB. We are interested in the length of the longest palindrome subsequence of a string. For example for input string HOMEWORK, the longest palindromic subsequence is OEO, and so the answer is 3.

(a) (10 points) Given input string as an array of characters A, give an $O(n^2)$ algorithm to find the length of the longest palindrome subsequence. (**Hint**: Think of the Longest Common Subsequence done in class.)

(b) (5 points) The problem in part (a) above can be solved directly by reversing the given string and then finding the length of the longest common subsequence (using the algorithm studied in the lecture) of the string and its reverse. Formally, it is known that the length of the longest palindrome subsequence is equal to the length of the longest common subsequence of the string and its reverse. However, in general, not all longest common subsequences of a string and its reverse are necessarily palindromes. Give an example of a string where one of the longest common subsequences of the string and its reverse is not a palindrome.

Problem 8-4 (Telephone Poles) 10+(5) Points

The telephone company that you are working for has recently taken over the telephone services in a new city. You have been assigned specifically to work on the telephone poles on Main street. There are N poles in a row from positions 1 to N, and pole i has height $H[i]$ feet, which is an integer in the range $[1, maxH]$.

The city has asked you to make all poles have the same height. For each i, if the ith pole has height h and the $(i-1)$st pole has height h', then you must pay a tax of $C|h - h'|$. To help achieve this goal, you can increase the height or decrease the height of any pole to height h with a cost of $(H[i] - h)^2$. Your task is to decide how to increase or decrease the height of each pole so that your company will spend the least amount of money. In particular, you must balance the cost of changing the heights of the poles, with the tax your company will have to pay.

(a) (5 points) Let $DP[i, h]$ be the minimum cost for the first i poles, so that the ith pole has height h. Give a base case and recurrence relation for $DP[i, h]$.

(b) (5 points) Use your recurrence in the previous part to give pseudocode for an $O(NH^2)$ time algorithm to find the minimum cost your company must pay.

(c) (**Extra Credit**: 5 points) Give pseudocode for an $O(NH)$ time algorithm to find the minimum cost your company must pay.