George Washington University
Department of Computer Science

Csci 6212 - Homework 6

Given: March 23, 2017
Due: 6pm, March 29, 2017

1. Give a dynamic programming algorithm for the following problem. The input is an n sided convex polygon. Assume that the polygon is specified by the Cartesian coordinates of its vertices. The output should be the triangulation of the polygon into $n-2$ triangles that minimizes the sums of the cuts required to create the triangles. Analyze the time complexity of your algorithm.

2. Give a dynamic programming algorithm to find the longest monotonically increasing subsequence of a sequence of n elements. Apply your algorithm to a sequence $A = \langle 1, 3, 2, 4, 6, 13, 14, 15, 5, 6, 8, 12, 13 \rangle$. Note that in this example, sequences $\langle 1, 3, 4, 6, 13, 14, 15 \rangle$ and $\langle 1, 2, 4, 5, 6, 8, 12, 13 \rangle$ both are monotonically increasing subsequence of A.

3. Consider a 2-D map with a horizontal river passing through its center. There are n cities on the southern bank with x-coordinates $a(1), \cdots, a(n)$ and n cities on the northern bank with x-coordinates $b(1), \cdots, b(n)$. You want to connect as many north-south pairs of cities as possible with bridges such that no two bridges cross. When connecting cities, you can only connect city i on the northern bank to city i on the southern bank. Give a dynamic programming algorithm to solve this problem and analyze the time complexity of your algorithm. Note that those x-coordinate values are not sorted, i.e., $a(i)$'s and $b(i)$'s are in an arbitrary order.

4. Give a polynomial time algorithm for the following problem. The input consists of a sequence $R = R_1, \cdots, R_n$ of non-negative integers, and an integer k. The number R_i represents the number of users requesting some particular piece of information at time i (say from a www server). If the server broadcasts this information at some time t, the requests from all the users who requested the information strictly before time t have already been satisfied, and requests arrived at time t will receive service at the next broadcast time. The server can broadcast this information at most k times. The goal is to pick the k times to broadcast in order to minimize the total time (over all requests) that requests/users have to wait in order to have their requests satisfied. As an example, assume that the input was $R = 3, 4, 0, 5, 2, 7$ (so $n = 6$) and $k = 3$. Then one possible solution (there is no claim that this is the optimal solution) would be to broadcast at times $2, 4$, and 7 (note that it is obvious that in every optimal schedule that there is a broadcast at time $n+1$ if $R_n \neq 0$). The 3 requests at time 1 would then have to wait 1 time unit. The 4 requests at time 2 would then have to wait 2 time units. The 5 requests at time 4 would then have to wait 3 time units. The 2 requests at time 5 would then have to wait 2 time units. The 7 requests at time 6 would then have to wait 1 time units. Thus the total waiting time for this solution would be

$$3 * 1 + 4 * 2 + 5 * 3 + 2 * 2 + 7 * 1.$$