CS 477/677 Analysis of Algorithms

Fall 2019

Homework 6

Due date: November 14, 2019

For the programming problem below, include in your hardcopy submission a listing of your algorithm and of the output. Please follow the electronic submission instructions.

1. (U&G-required) [100 points]

You and your friends are planning to watch some astronomical events that will happen this evening. You know that there will be *n* events, occurring in sequence and separated by 1 minute intervals. Event *j* will happen at minute *j* and if you do not look at the event at that minute, you will miss it. You also know that your telescope maps the sky as a one dimensional coordinate system that is measured in degrees from some central viewpoint. Each event *j* will take place at coordinate d_j (integer value). The telescope starts at coordinate 0 at minute 0 and can only move at the rate of one degree per minute. Due to this, you will not be able to observe all *n* events, but you and your friends would like to see as many of them as possible. However, last event of the night (*n*) is the most important so you must observe it.

A subset S of events is called *observable* if it is possible to view each event $j \in S$ at its given time *j* and if the telescope has enough time to move between consecutive events, given that it can move at maximum one degree per minute. Given a set of coordinates for *n* astronomical events, develop a dynamic programming algorithm that finds an *observable* subset S of maximum size, with the constraint that event *n* (the last one) must be viewed.

For example, for the set of events shown below, the optimal solution is to view events 1, 3, 6 and 9. The telescope has enough time to move from one event to the next at the speed of one degree per minute.

Event	1	2	3	4	5	6	7	8	9
Coordinate	1	-4	-1	4	5	-4	6	7	-2

To find the optimal value and the optimal solution for this problem follow the steps outlined below.

(a) [20 points] Write a recursive formula of an optimal solution (i.e., define the variable that you wish to optimize and explain how a solution to computing it can be obtained from solutions to subproblems).
Submit: the recursive formula, along with definitions and explanations on what is computed.

(b) [30 points] Write an algorithm that computes the optimal value to this problem, based on the recurrence above. Implement your algorithm in C/C++ and run it on the following values:
Submit:

- A printed version of the algorithm
- A printout of the table that contains the solutions to the subproblems, run on the values given above (print the entire table!)
- The optimal value computed by your algorithm

(c) [20 points] Update the algorithm you developed at point (b) to enable the reconstruction of the optimal solution, i.e., which events you choose to view for an optimal solution. (Hint: use an auxiliary table like we did in the examples in class.) Include these updates in your algorithm implementation from point (b).
Submit:

- A printed version of the algorithm
- A printout of the values that you obtain in the table containing the additional information needed to reconstruct the optimal solution, run on the values given above (print the entire table)

(d) [30 points] Using the additional information computed at point (c), write an algorithm that outputs the events you choose to view in the optimal solution. Implement this algorithm in C/C+.

Submit:

- A printed version of the algorithm
- A printout of the **solution** to the problem run on the values given above.

2. (G-required) [20 points] Show that in order to fully parenthesize an expression having *n* matrices we need *n-1* pairs of parentheses.

Extra credit

6. [20 points] List all the different orders in which we can multiply five matrices A, B, C, D, and E.