Problem Set 5

All parts are due May 9, 2017 at 11:59PM.

Name: Faaya Abate Fulas

Collaborators: Ebenezer Nkwate

Part A

Problem 5-1. This is a variation of the Longest Common Subsequence problem since the longest subsequence of matching colors will be a set containing the most number of points that can be connected without intersecting. Therefore:

-Use an alphabet with *k* different characters, label each color, and put the colored points on the y-axis in increasing order of appearance into a string A, and the points on the x-axis in increasing order of appearance into a string B.

-Then, use LCS logic covered in lecture to solve the problem in $O(k^2)$.

Problem 5-2.

(a) Each turn, Ben compares the first and the last digits and picks the minimum. For a number with n-digits, this algorithm runs in $O(n)$ time. This greedy strategy is not always optimal since picking the larger number on one turn might have a better pay off on the next turn.

- For example, if on Ben's turn the number is 6158 , Ben would pick 6, Alyssa would pick 1, then Ben would pick 5, and Alyssa would pick 8 and win with a minimum score of 9.

(b) Let DP(i,j) be the minimum score a player can have from the i^{th} to the j^{th} digit. Then:

- $DP(i, j) = d$, for $i = j$ and,
- $DP(i, j) = Min(d_i\ , d_j)$, for $j = i + 1$

Since the opponent also plays optimally, the best digit a player can pick on each turn is determined by the minimum of the two edge digits, and the maximum payoff of the remaining digits after his opponent picks on the next turn.

Then,

$$DP(i, j) = Min(d_i + Max(DP(i + 1, j - 1), DP(i, j - 2))$$

Number of Subproblems: For every choice of i and j, since there are n digits $\implies \Theta(n^2)$

Time/Subproblem: $\Theta(1)$

Total Running Time: $\Theta(n^2)$

Problem 5-3.

(a) Let $NW_{i,j}$ be the minimum value of any square in the northwest quadrant. Then:

- $NW_{0,j}$ and $NW_{i,0} = \infty, \forall i, j$ (edge cells) and,
- $NW_{1,1} = n_{0,0}$

Then,

$$NW_{i,j} = Min(NW_{i-1,j}, NW_{i,j-1}, n_{i-1,j-1})$$

Number of Subproblems: For an n-row by m-column grid $\implies \Theta(nm)$

Time/Subproblem: $\Theta(1)$

Total Running Time: $\Theta(nm)$

(b) Similarly, the edge cells for each of the following calculations have a value of ∞, and:

$SW_{i,j} = Min(SW_{i+1,j}, SW_{i,j-1}, n_{i+1,j-1})$, with base case $SW_{n-1,1} = n_{n,0}$

$NE_{i,j} = Min(NE_{i-1,j}, NE_{i,j+1}, n_{i-1,j+1})$, with base case $NE_{1,m-1} = n_{0,m}$

$SE_{i,j} = Min(SE_{i+1,j}, SE_{i,j+1}, n_{i+1,j+1})$, with base case $SE_{n-1,m-1} = n_{n,m}$

- NE, SW, NW, SE values can be computed in $O(nm)$ time for all cells.
- The products of these values for each cell can be computed in $O(1)$ time.
- And, the cell with the minimum product value can be found in $O(nm)$ time.

Therefore, this algorithm will find the optimal square in $O(nm)$ time.

Problem 5-4.

(a) Let $P(c, i, j)$ be the optimal number of steps needed to go from some character *c* to a substring $S[i, j]$ of S. Then:

- $P(c, i, j) = 0$, for $i = j$ (Since no step is needed to convert c to itself)

To get the best value for P, the substring S[i,j] can be subdivided by some index r, so that for each stage, all the possible ways of getting to the target substring are explored. Then, the optimal P is minimum of those P values plus the current step.

Therefore,

$$P(c, i, j) = Min_{i<r<j}(P(c, i, r-1), P(c, r, j)) + 1$$

(b) **Number of Subproblems:** For every choice of i and j, and for every possible values of c, the number of subproblems is $O(kn^2)$.

Time/Subproblem: For each c, there are m possible rules to be applied and there are n possible ways to split the substring, $S[i, j]$. Therefore, each subproblem can be solved in $O(nm)$ time.

Total Running Time: $O(n^3mk)$

Part B

Problem 5-5. Submit your implementation on `alg.csail.mit.edu`.