Mubarra Khan Z5161249 Thursday, 30 July 2020

Q2

You are given a 2D map consisting of an R×C grid of squares; in each square there is a number representing the elevation of the terrain at that square. Find a path going from square (1, R) which is the top left corner of the map to square (C, 1) in the lower right corner which from every square goes only to the square immediately below or to the square immediately to the right so that the number of moves from lower elevation to higher elevation along such a path is as small as possible. (20 pts)

Solution:

Begin by creating a new matrix D which stores the minimum number of elevations from (1,R) to every cell.

- Basically every cell will correspond to the matching cell in the original matrix M but will store the total number of elevations it took to get there from the starting point (1,R)
- Each cell will also store the coordinates of the previous cell in its path that lead to the minimum number of elevations.

* elevation is defined as - if for any x, y values if x > y then x is an elevation

<u>How to set values for matrix D</u>

In matrix D we will set (1, R) to 0

<u>Top row</u>

- If the cell on the right hand side of (1,R) in matrix M is an elevation we will add 1 to the value at (1,R) and set that as the value for (2, R) in matrix D
- Else we will set (2, R) to 0 as there is no elevation
- We will also set the predecessor for this cell to be (1, R)
- We will continue this process for the whole top row adding 1 to the previous total if the next cell is an elevation or nothing if it is not an elevation.

<u>First column</u>

- This will be similar to the top row however we will be comparing the node above each cell beginning with (1, R-1).
- If (1,R-1) is an elevation from (1,R) we will add 1 to the total elevations and set that as the cell value in matrix D

- The predecessor will be the cell above.

Recursion:

This will begin at cell (2, R-1).

- First we will check if this cell is an elevation from the cell above (2, R) and call this total totalU.
  - Basically totalU = (2,R) + 1 if it is an elevation or totalU = (2,R) if not
- Next we will check if this cell (2, R-1) is an elevation from (1, R-1) and call the this total totalR in the same way
  - Basically totalM = (2,R-1) + 1 if it is an elevation or totalM = (2,R-1) if not
- We will then compare totalU with totalR and the smaller of the two we will set as the value for cell (2, R -1).
  - Depending on which total we choose we will set the predecessor for the cell.
    - totalU -> means predecessor is from above
    - totalR - means predecessor is from the left hand side.

The above process will be recursively continued for all the cells in the matrix until we reach (C, 1).

By backtracking using the predecessor values we saved in each cell for matrix D we can find the path with the minimum number of elevations using the original Matrix M.

- We first store the value at (C,1) in matrix M in a list
- Next check the predecessor value using matrix D as a reference and append to the start of the list
- Continue this process until the predecessor is the value at (1, R).

The time complexity for this will be O(RXC) the size of the grid.