Exercise 9.17 *(Final exam - Question 1 - Winter 2022)*

Samantha is cooking from her garden, which is arranged in grid with n rows and m columns. Each cell $(i, j)(1 \leq i \leq n, 1 \leq j \leq m)$ *has an ingredient growing in it, with tastiness given by a positive value* T_{ij}*. To prepare a dinner, Samantha stands at a cell* (i, j) *and pick one ingredient from each quadrant relative to that cell. The tastiness of her dish is the product of the tastiness of the four ingredients she chooses. Help Samantha find an* $O(nm)$ *dynamic programming algorithm to maximize the tastiness of her dish.*

The four quadrants relative to a cell (i, j) *are defined as follows:*

top-left quadrant of $(i, j) =$ *all cells* (a, b) *such that* $a < i, b < j$,

bottom-left quadrant of $(i, j) =$ *all cells* (a, b) *such that* $a > i, b < j$,

top-right quadrant of $(i, j) =$ *all cells* (a, b) *such that* $a < i, b > j$,

bottom-right quadrant of $(i, j) =$ *all cells* (a, b) *such that* $a > i, b < j$.

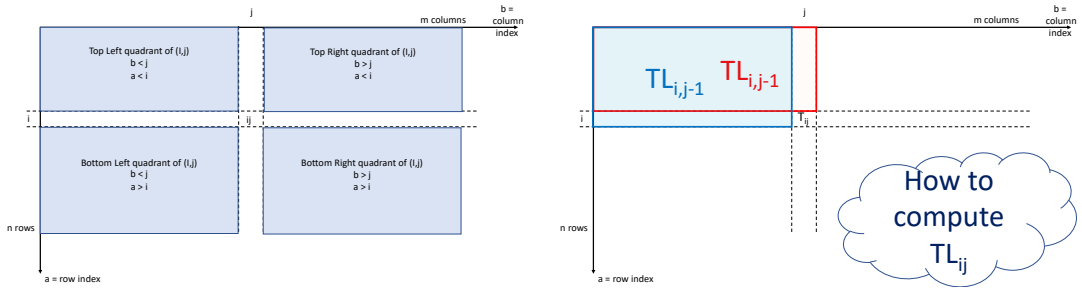
Because Samantha needs all four quadrants to be non-empty, she can only stand on cells (i, j) *where* $1 < i < n$ *and* $1 < j < m$.

(a) *Define* TL_{ij} *to be maximum tastiness value in the top-left quadrant of cell* (i, j)*:*

$$TL_{ij} = \max\{T_{ab} : 1 \leq a \leq i, 1 \leq b \leq j\}.$$

Find a dynamic programming algorithm to compute TL_{ij}*, for all* $1 < i < n$ *and* $1 < j < m$*, in* $O(nm)$ *time. While writing the algorithm, explain clearly how you proceed with the initialization of the values. What is the space complexity requirement?*

(b) *Use the idea in part (a) to obtain an* $O(nm)$ *algorithm to find the tastiest dish. Again, what is the space complexity requirement?*



(a) Quadrants. (b) Computing TL_{ij}.

Figure 9.6: Garden of Samantha

Solution

(a) When trying to calculate TL_{ij}, we see that the maximum can be at cell (i, j). If not, it must lie either in the rectangle from $(1, 1)$ to $(i, j-1)$, or the rectangle from $(1, 1)$ to $(i-1, j)$, or both. These three overlapping cases cover our required rectangle. We have then,

$$\text{TL}_{ij} = \max\{T_{ij}, TL_{i-1,j}, TL_{i,j-1}\}.$$

For the base cases, we can just set $\text{TL}_{0,j} = \text{TL}_{i,0} = 0$ for all valid values of i and j. We can compute the DP value for each state in $O(1)$ time. There are nm states, so our algorithm is $O(nm)$ (time complexity). If all values are stored in an array, then our algorithm is $O(nm)$ (space complexity).

(b) In part (a) we calculated range maximum for the top-left quadrant. We can similarly define range maximums for the other quadrants. Let $\text{BL}_{ij} = \max\{T_{ab} : i \le a \le n, 1 \le b \le j\}$, $\text{TR}_{ij} = \max\{T_{ab} : 1 \le a \le i, j \le b \le m\}$, and $\text{BR}_{ij} = \max\{T_{ab} : i \le a \le n, j \le b \le m\}$. Each of these can be computed in $O(nm)$ time similar to TL. To calculate the tastiest dish Samantha can cook when she stands at cell $(i, j)(1 < i < n$ and $1 < j < m)$, we now just need to compute the product $\text{TL}_{i-1,j-1} \times \text{BL}_{i+1,j-1} \times \text{TR}_{i-1,j+1} \times \text{BR}_{i+1,j+1}$ and pick the maximum product. This can be done in $O(nm)$ time.