Recitation03_6b

Input: $T[1:n, 1:n]$
Output: p =The largest value of the path such that:

1. The elements in the path are in increasing order
2. The value of the path is the product of the number on it

- *Subproblem*:
  $B[i,j]$:= the value of the path from $(n,1) \to (i,j)$
- *Order*:
  Row $(n \to 1)$ and column $(1 \to n)$. Row is finished first.
- *Start*:
  $B[n,1] = T[n,1]$
- Continue:

$$B[i,j] = T[i,j] * \max \begin{cases} B[i+1,j] & if \;\; T[i+1,j] < T[i,j], \;\; i+1 > 0 \;\; else \;\; 0 \\ B[i,j-1] & if \;\; T[i,j-1] < T[i,j], \;\; j-1 > 0 \;\; else \;\; 0 \end{cases}$$

- *End*:
  $p = \max\{B[n,n]\}$
- *Correctness*:
  We can either get a path to a entry from the left side or from the bottom side. We check if the the elements on the path coming in our entry are in increasing order with the value at the entry we have. If not than we just consider as the path coming in does not exits therefore we get 0. Now we take the max because we want to choose which path coming in is larger that is why we take the max. Then we just multiply with the entry value we are in and the get the length of the path ending at our entry. At the end we check which entry has the biggest value to get the largest path.
- *Running time*:
  We do constant at each entry building the $B[n,n]$ matrix $O(n^2)$ and we do a search for the maximum which is $O(n^2)$. Therefore overall our algorithm is $O(n^2)$.
- *Optimal object*:
  It doesn't require one, but if it asked for the path we will have stored in another matrix where did the path to the entry come from: left side or from the bottom. And then you can easy trace back till $(n,1)$.