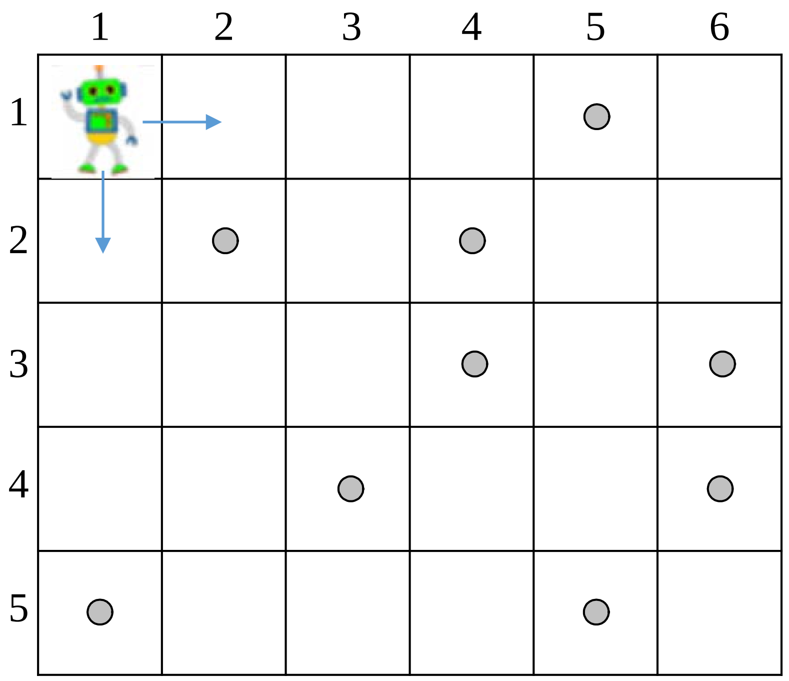
Dynamic Programming: Coin-collecting Robot

(Chapter 8)

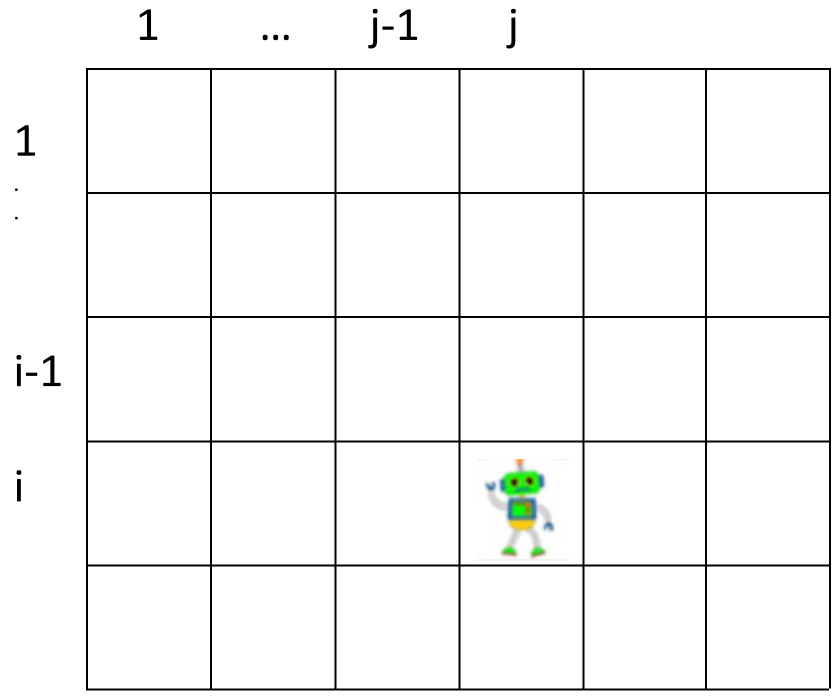
Coin-collecting robot

Several coins are placed in cells of an *n*×*m* board. A robot, located in the upper left cell of the board, needs to collect as many of the coins as possible and bring them to the bottom right cell. The robot can only move *right* or *down*.



Solution

- Let F(i,j) be the largest number of coins the robot can collect and bring to cell (i,j) in the *ith* row and *j*th column.

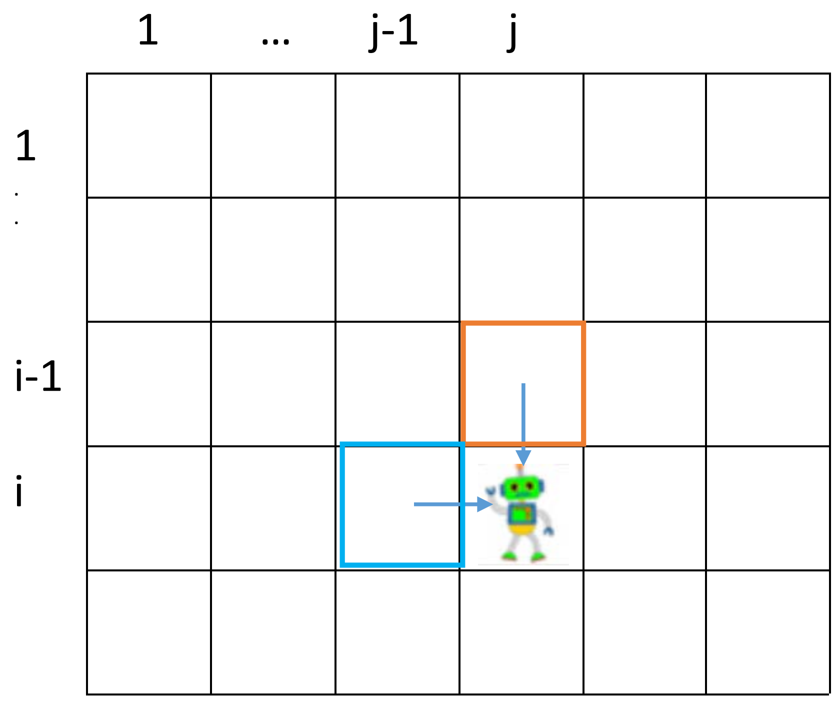


Solution

How many coins could the robot bring to cell (i,j)?

If it comes from the left → F(i, j-1)

If it comes from above → F(i-1, j)

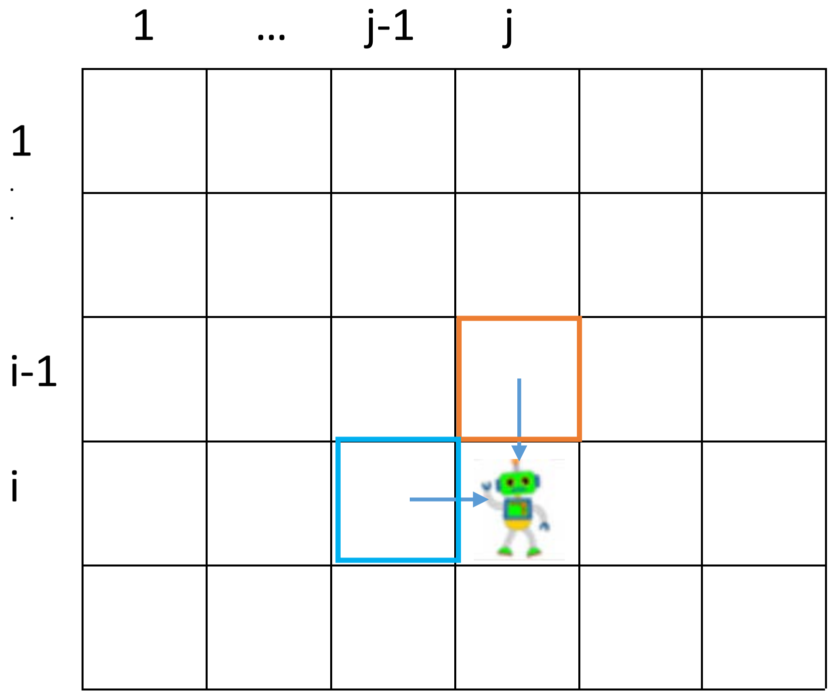


Solution

Recursive definition:

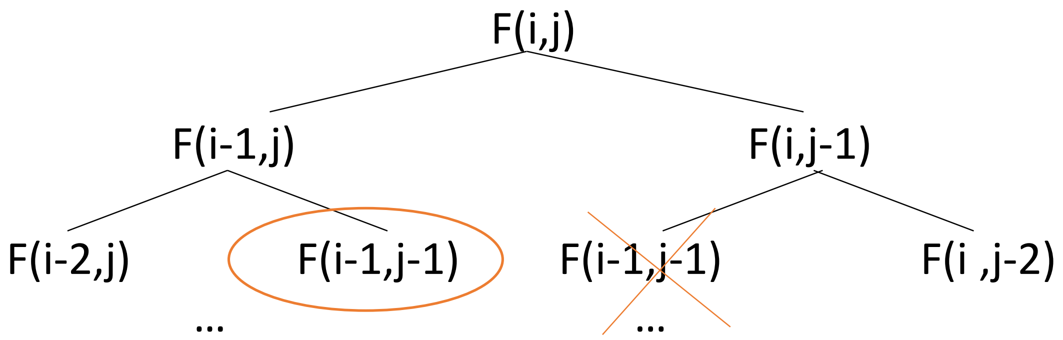
$$F(i, j) = \max\{F(i-1, j),\ F(i, j-1)\} + c_{ij} \quad \text{for } 1 \le i \le n,\ 1 \le j \le m$$

where $c_{ij} = 1$ if there is a coin in cell (i,j), and $c_{ij} = 0$ otherwise



Solution (cont.)

- $F(i, j) = \max\{F(i-1, j), \; F(i, j-1)\} + c_{ij}$
- $F(0, j) = 0$ for $1 \leq j \leq m$ and $F(i, 0) = 0$ for $1 \leq i \leq n$.



Solution (cont.)

Bottom-up calculation

$$F(i, j) = \max\{F(i-1, j),\ F(i, j-1)\} + c_{ij} \quad \text{for } 1 \le i \le n,\ 1 \le j \le m$$

	1	2	3	4	5	6
1					●	
2		●		●		
3				●		●
4			●			●
5	●				●	

	1	2	3	4	5	6
1	0	0	0	0	1	1
2	0	1	1	2	2	2
3	0	1	1	3	3	4
4	0	1	2	3	3	5
5	1	1	2	3	4	**5**

Robot Coin Collection

```
ALGORITHM RobotCoinCollection(C[1..n, 1..m])
  // Robot coin collection using dynamic programming
  // Input: Matrix C[1..n, 1..m] with elements equal to 1 and 0 for
  //        cells with and without coins, respectively.
  // Output: Returns the maximum collectible number of coins
  F[1, 1] ← C[1, 1]
  for j ← 2 to m do
    F[1, j] ← F[1, j - 1] + C[1, j]
  for i ← 2 to n do
    F[i, 1] ← F[i - 1, 1] + C[i, 1]
    for j ← 2 to m do
      F[i, j] ← max(F[i - 1, j], F[i, j - 1]) + C[i, j]
  return F[n, m]
```

Complexity? Θ(*nm*) time, Θ(*nm*) space