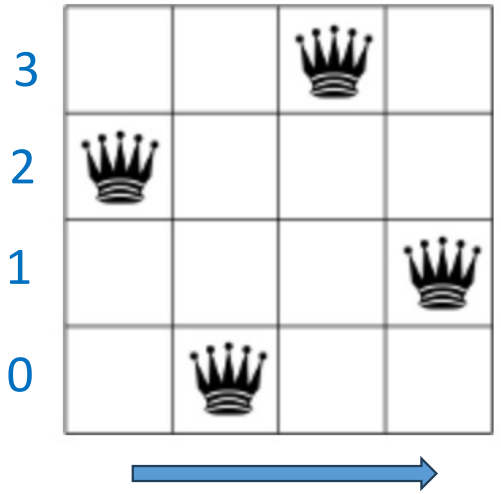
Algoritmos y Programación

Práctica 4.2: Backtracking

N-Queens mediante vuelta atrás (backtracking)

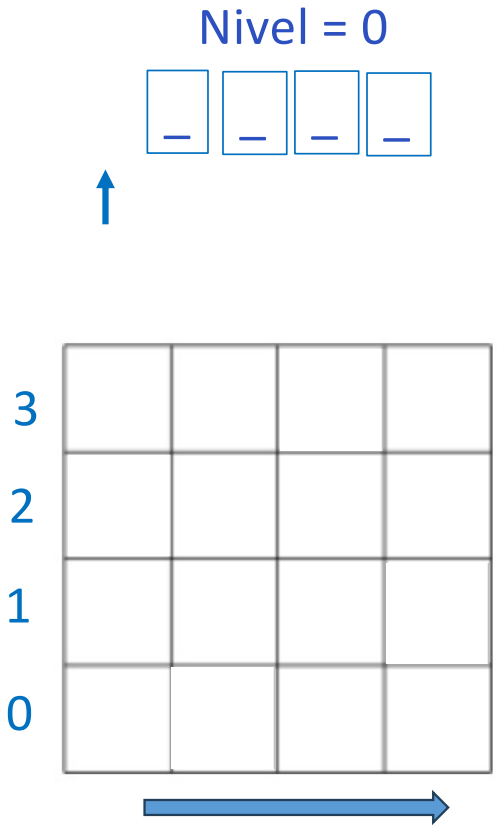
- Un VPL:
  1) Realizando un recorrido en profundidad (DFS) encuentra todas las soluciones del problema de las reinas mediante backtracking.

Nuestro programar debe generar exactamente las mismas soluciones que la versión que programamos mediante fuerza bruta.

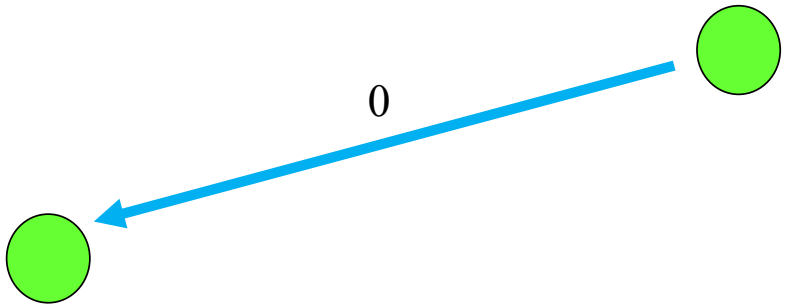


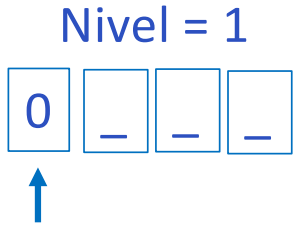
Nivel de profundidad de la recursividad

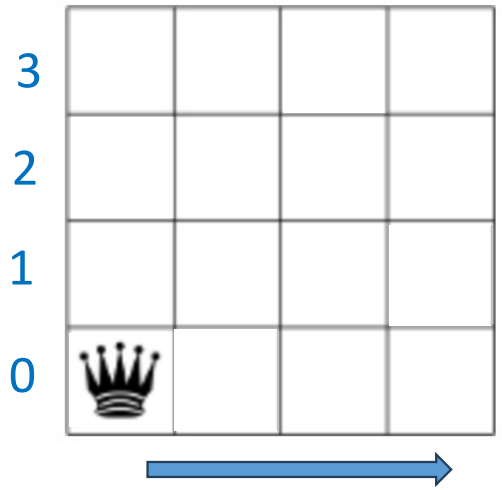
Ejemplo: Solución por Backtracking



Ejemplo: Solución por Backtracking

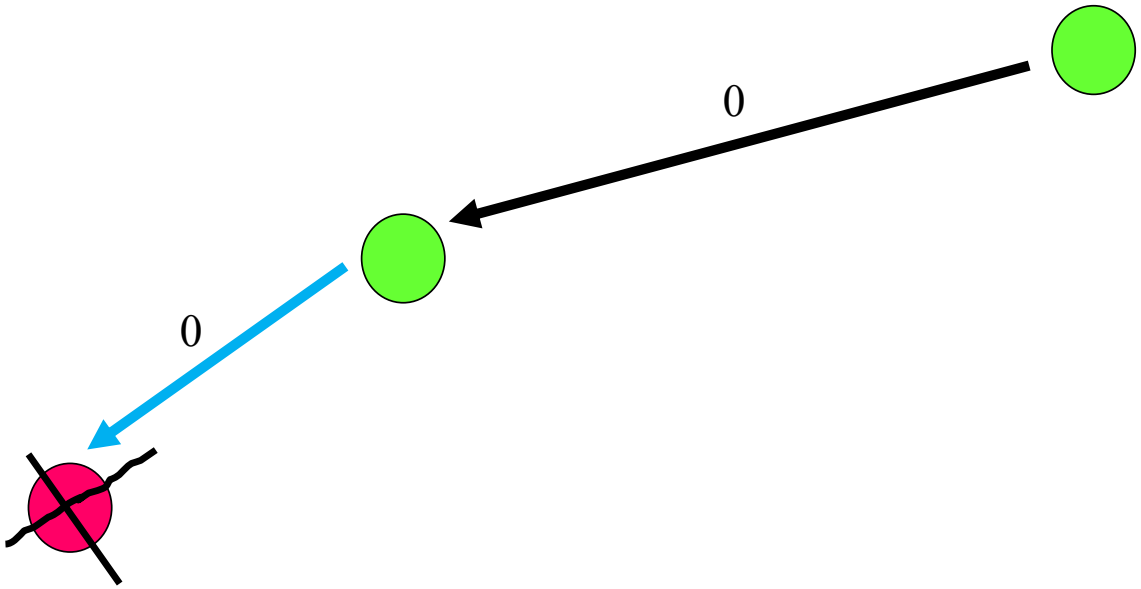






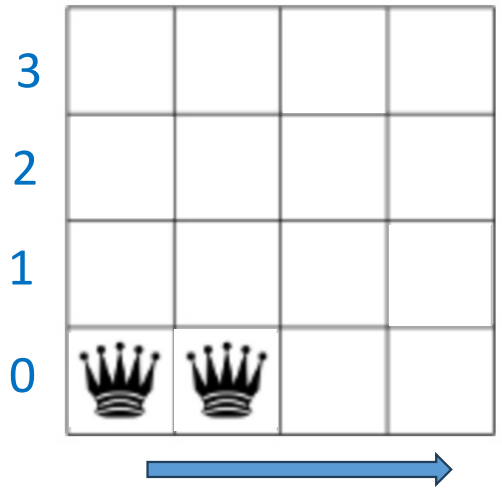
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



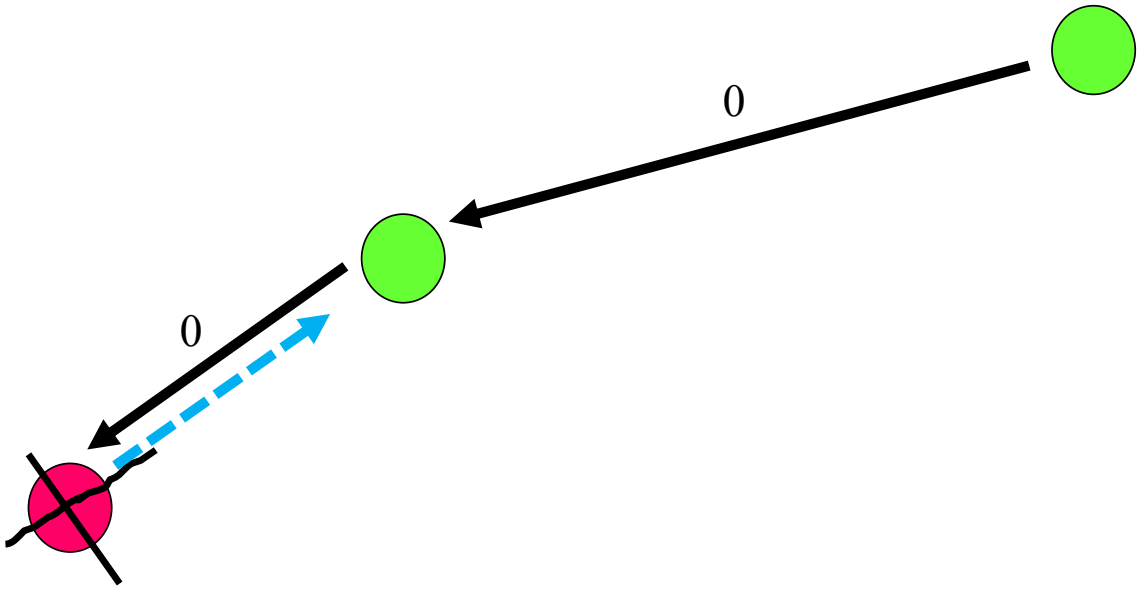
Nivel = 2

0	0	_	_



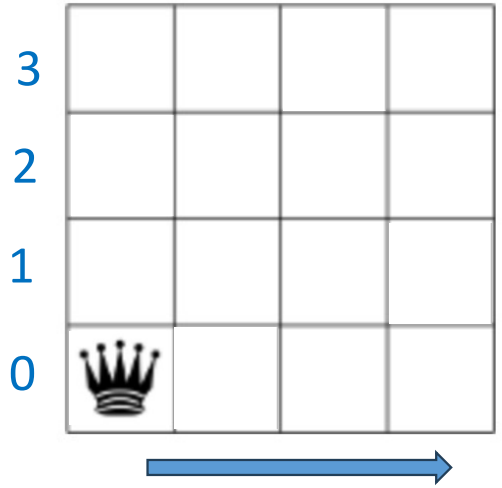
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



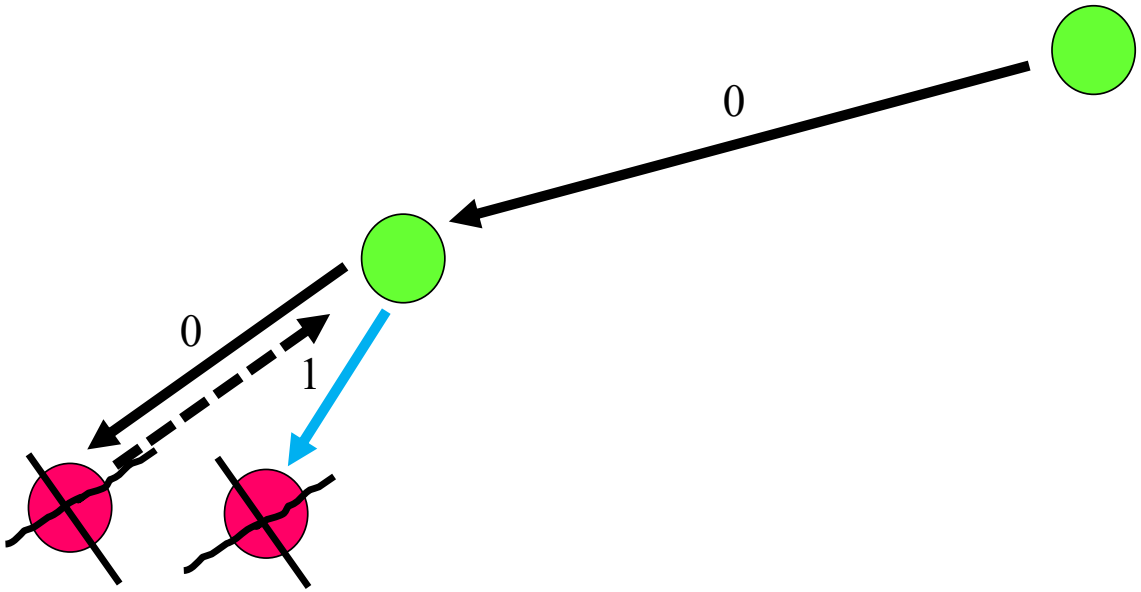
Nivel = 1

0	_	_	_

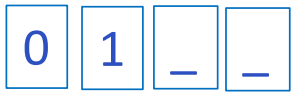


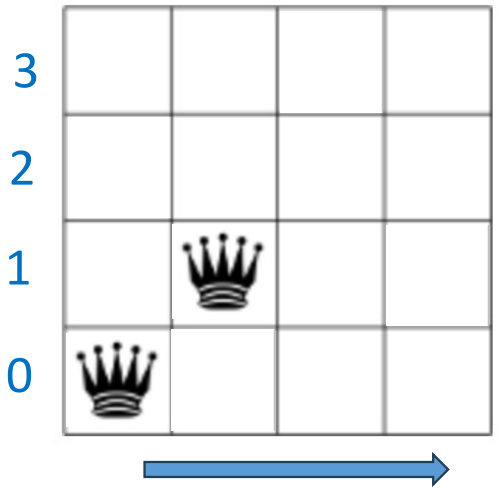
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



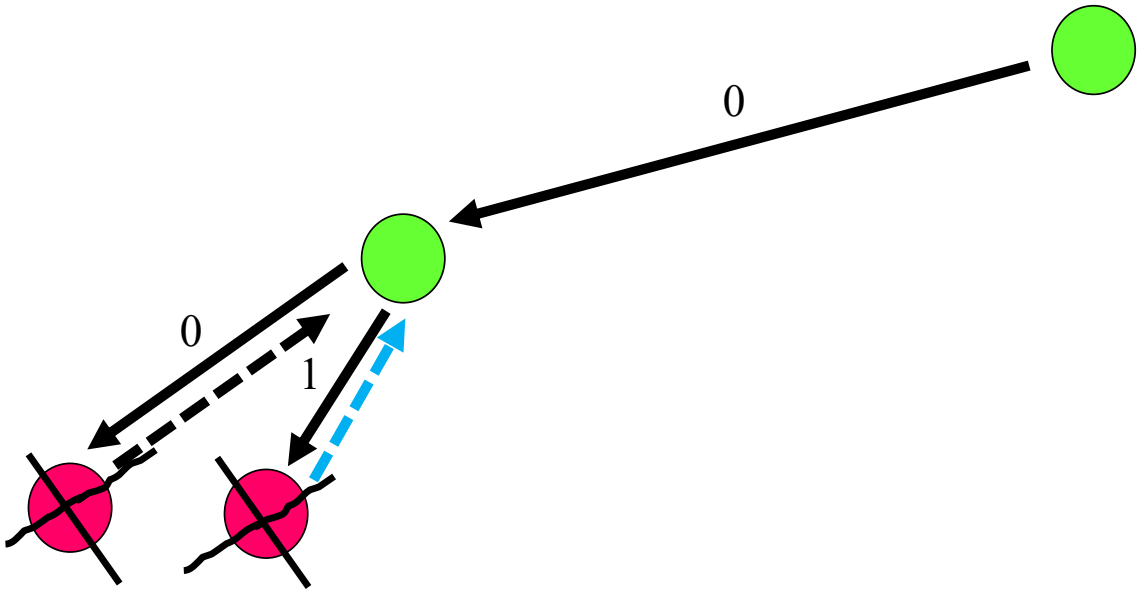
Nivel = 2



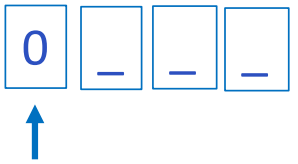


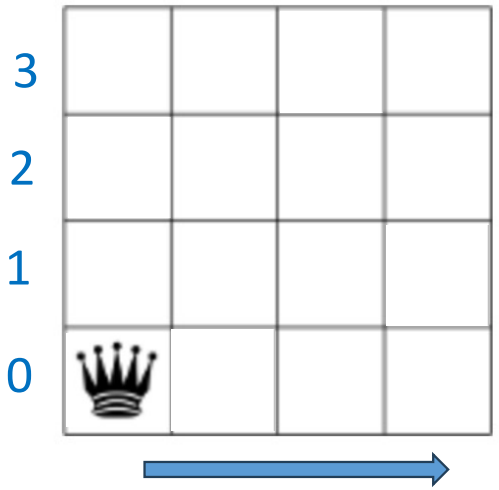
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



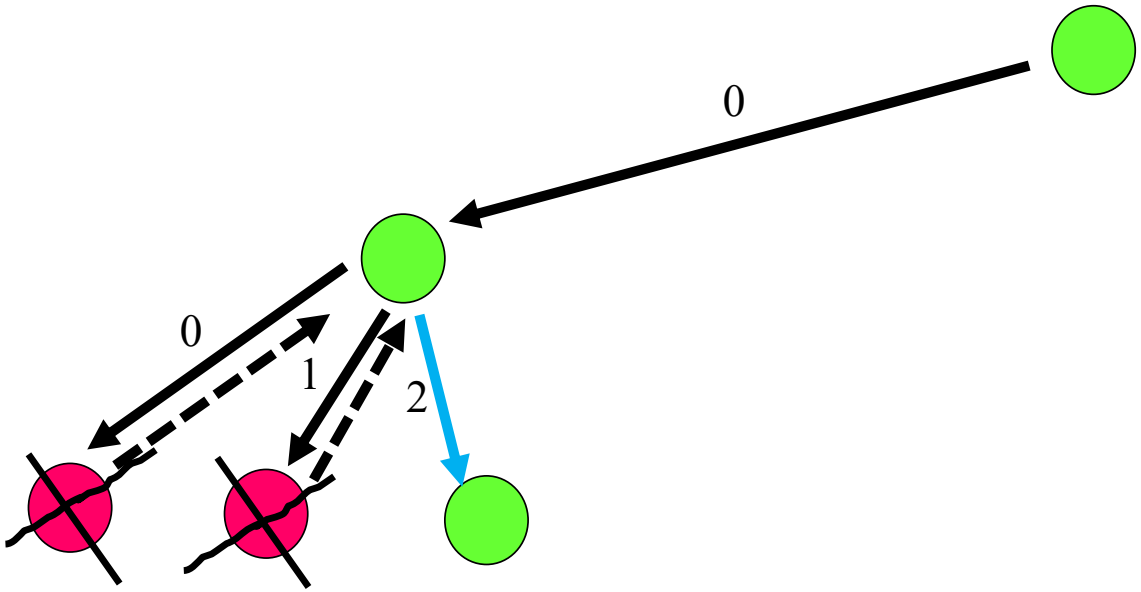
Nivel = 1



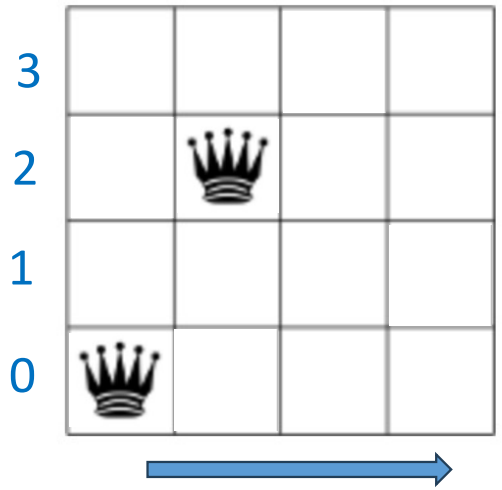


Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking

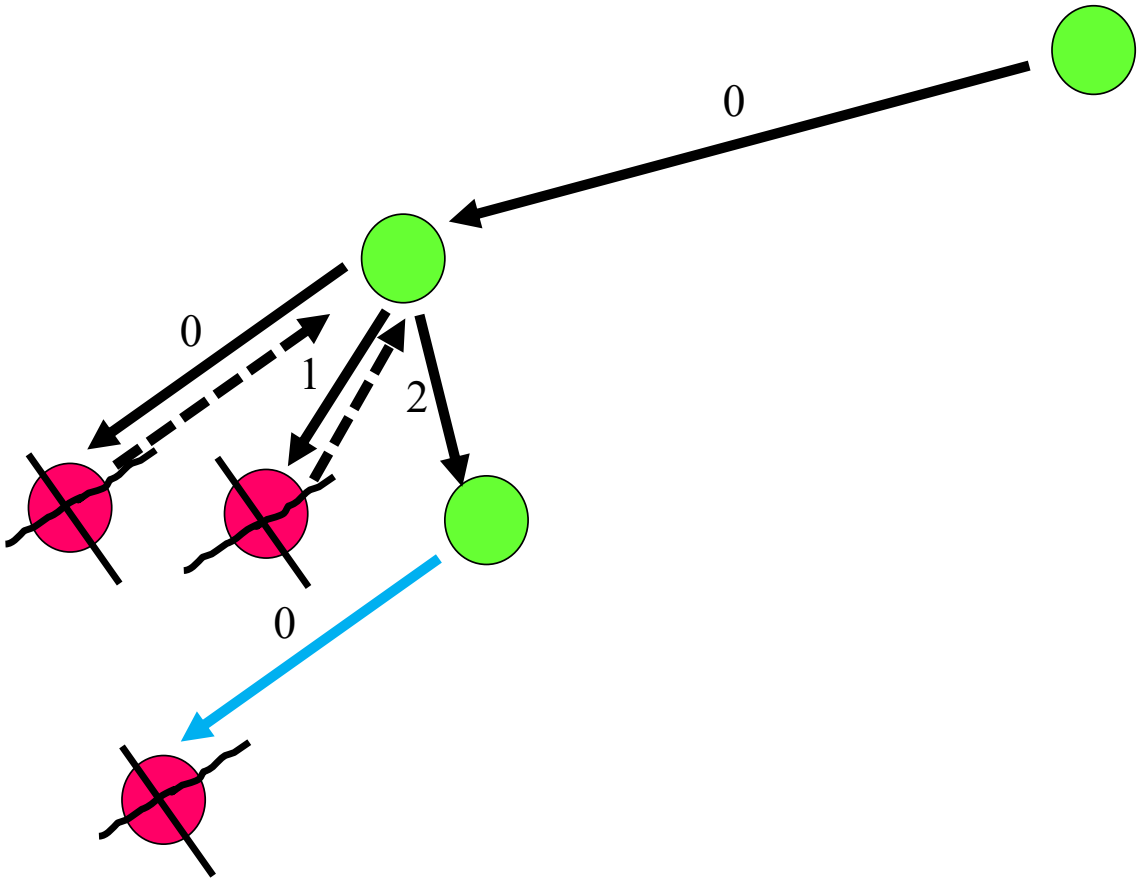


Nivel = 2



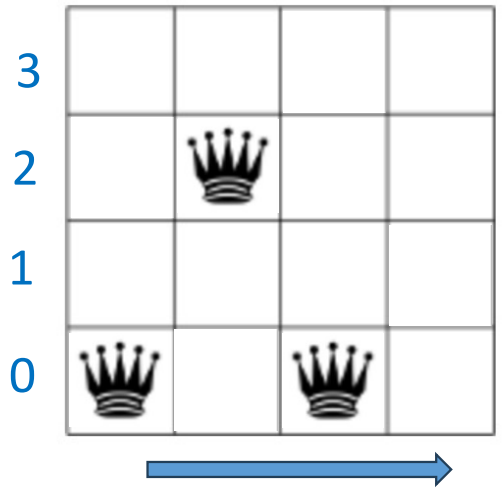
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



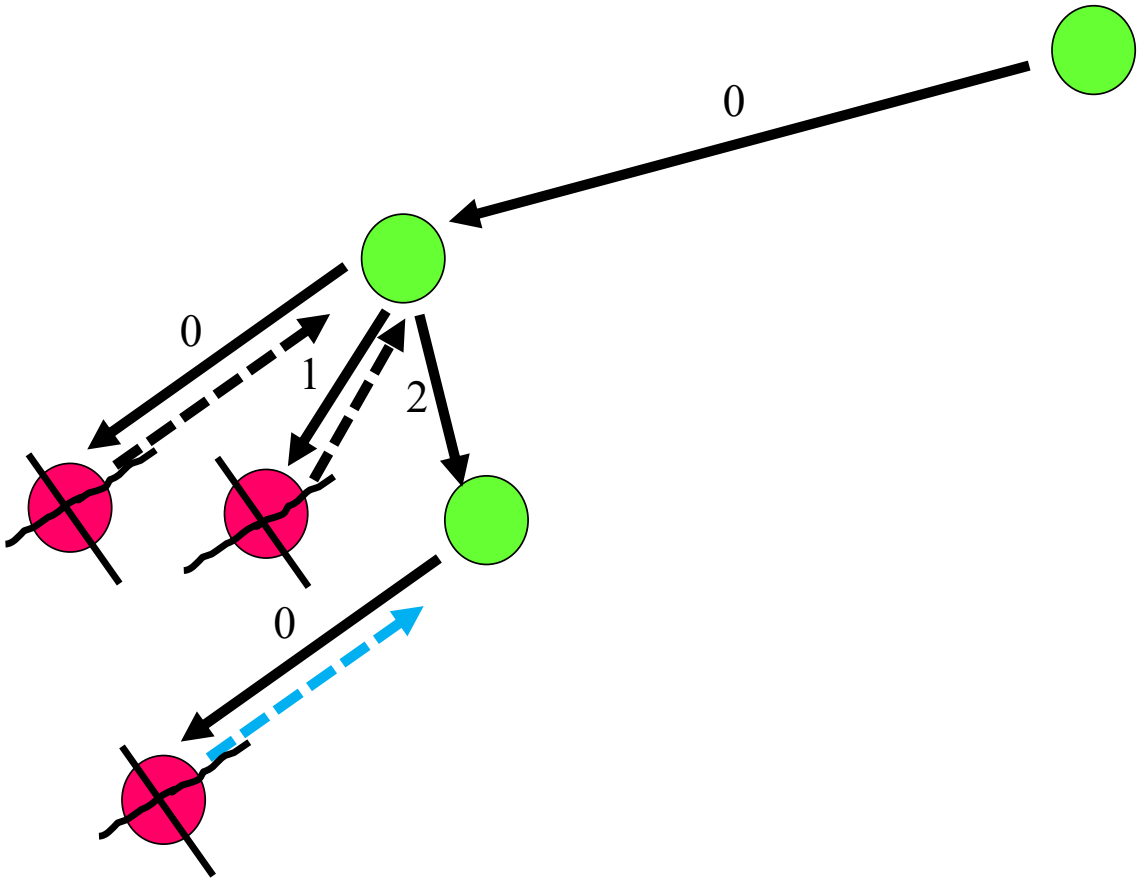
Nivel = 3

0	2	0	_



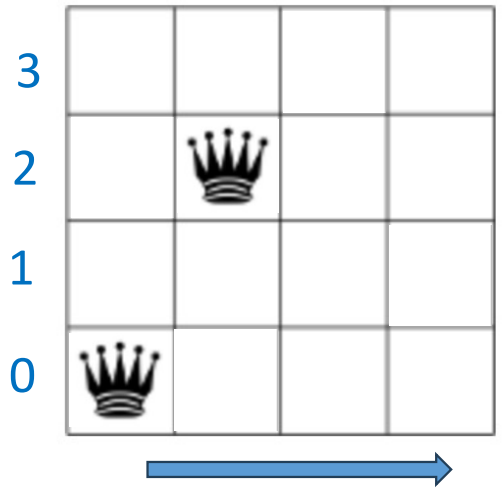
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



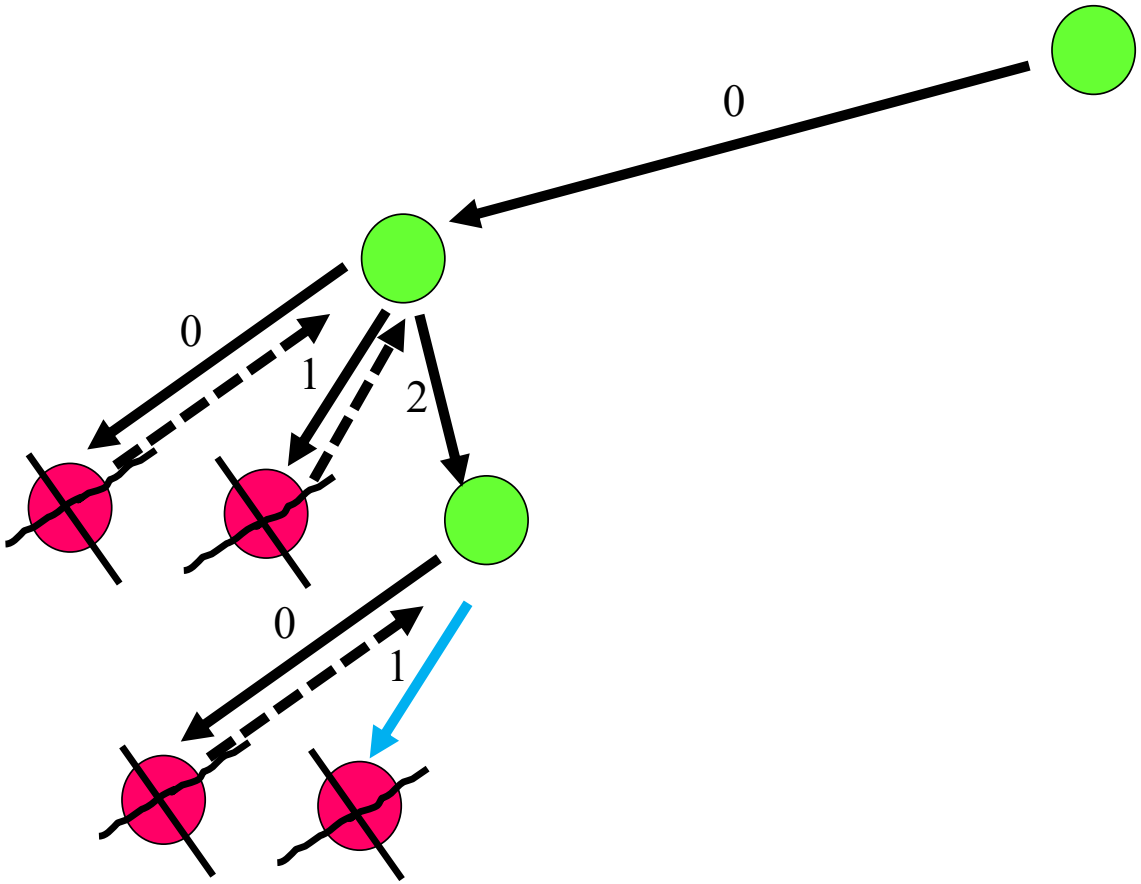
Nivel = 2

0	2	_	_



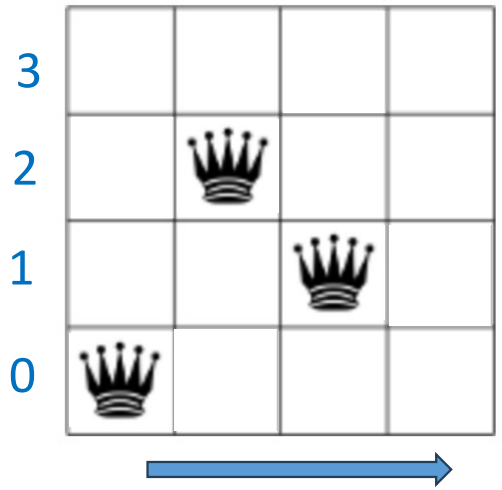
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



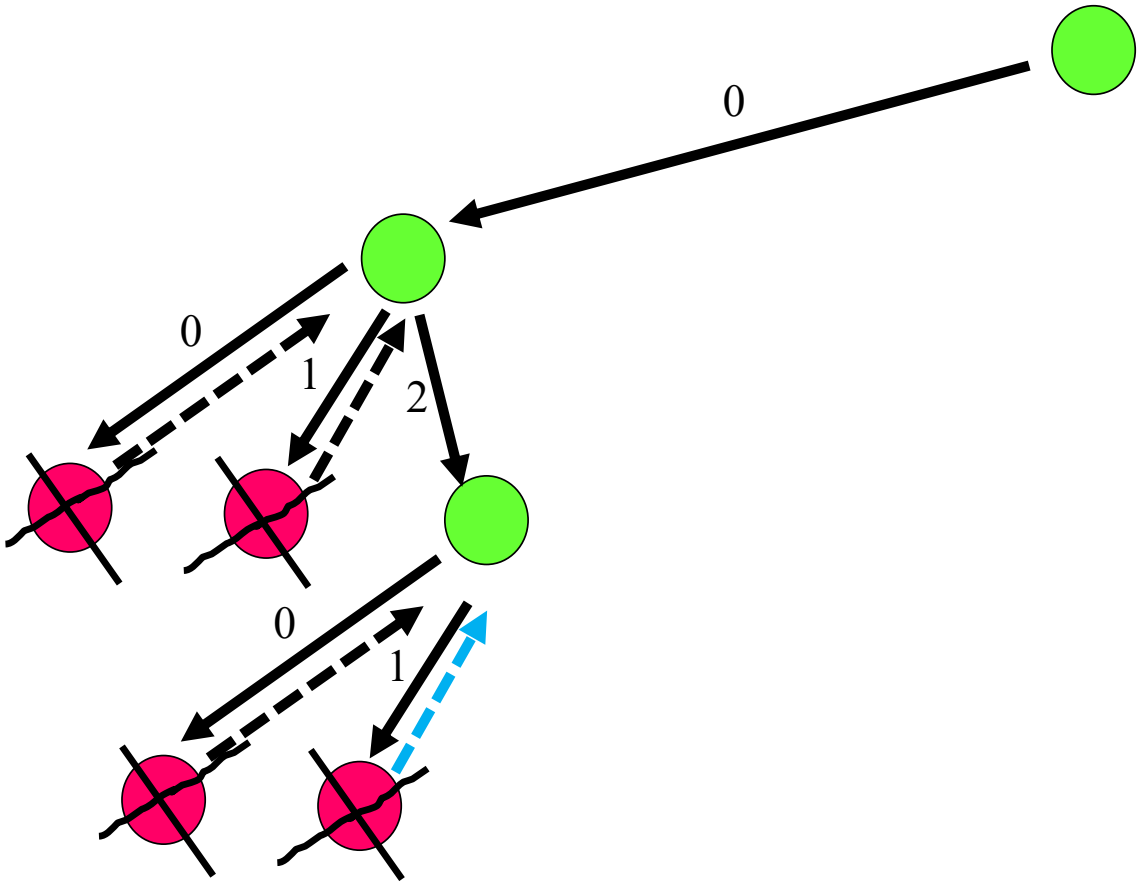
Nivel = 3

0	2	1	_



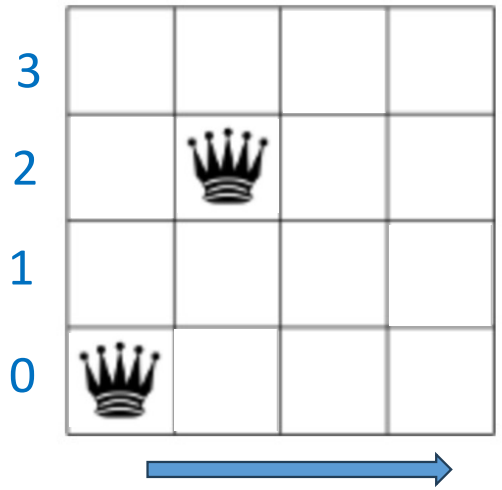
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



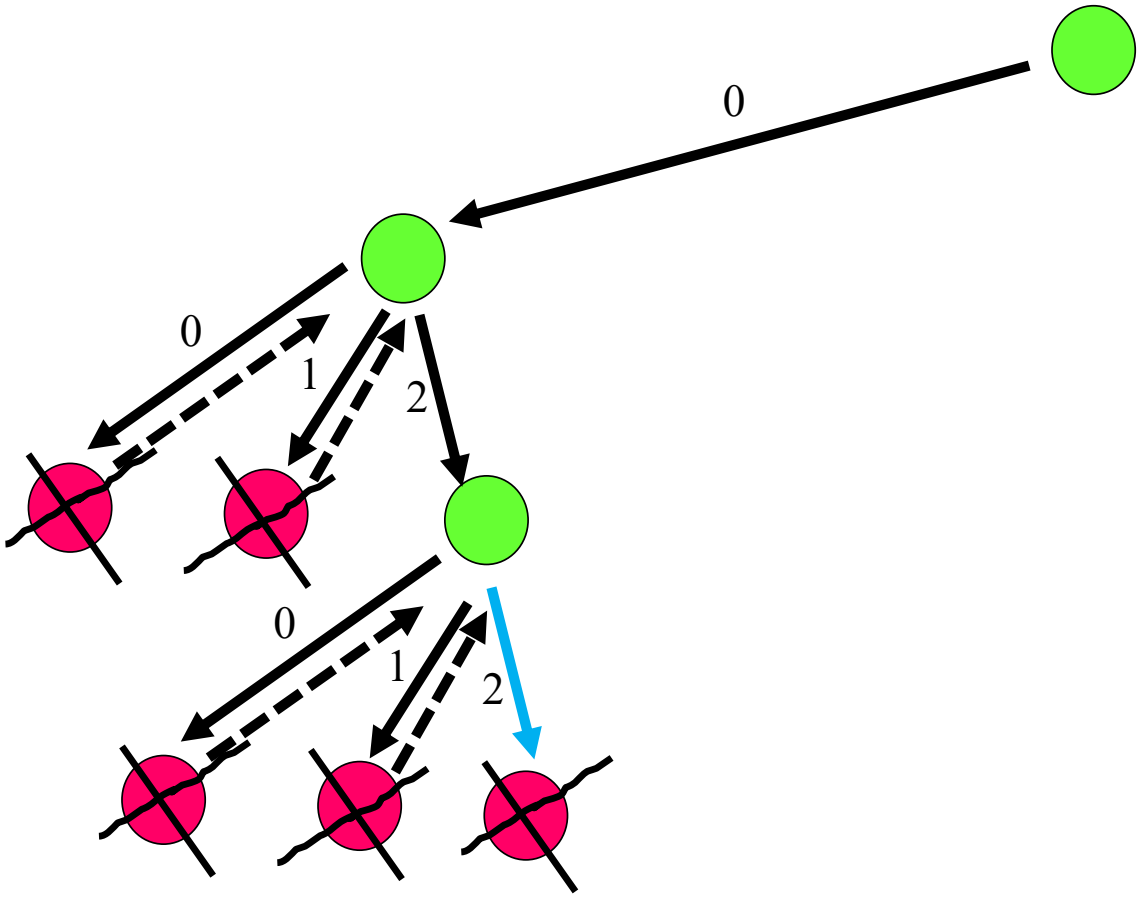
Nivel = 2

0	2	_	_



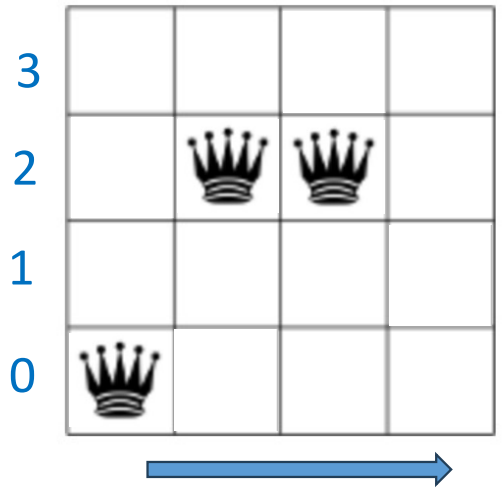
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



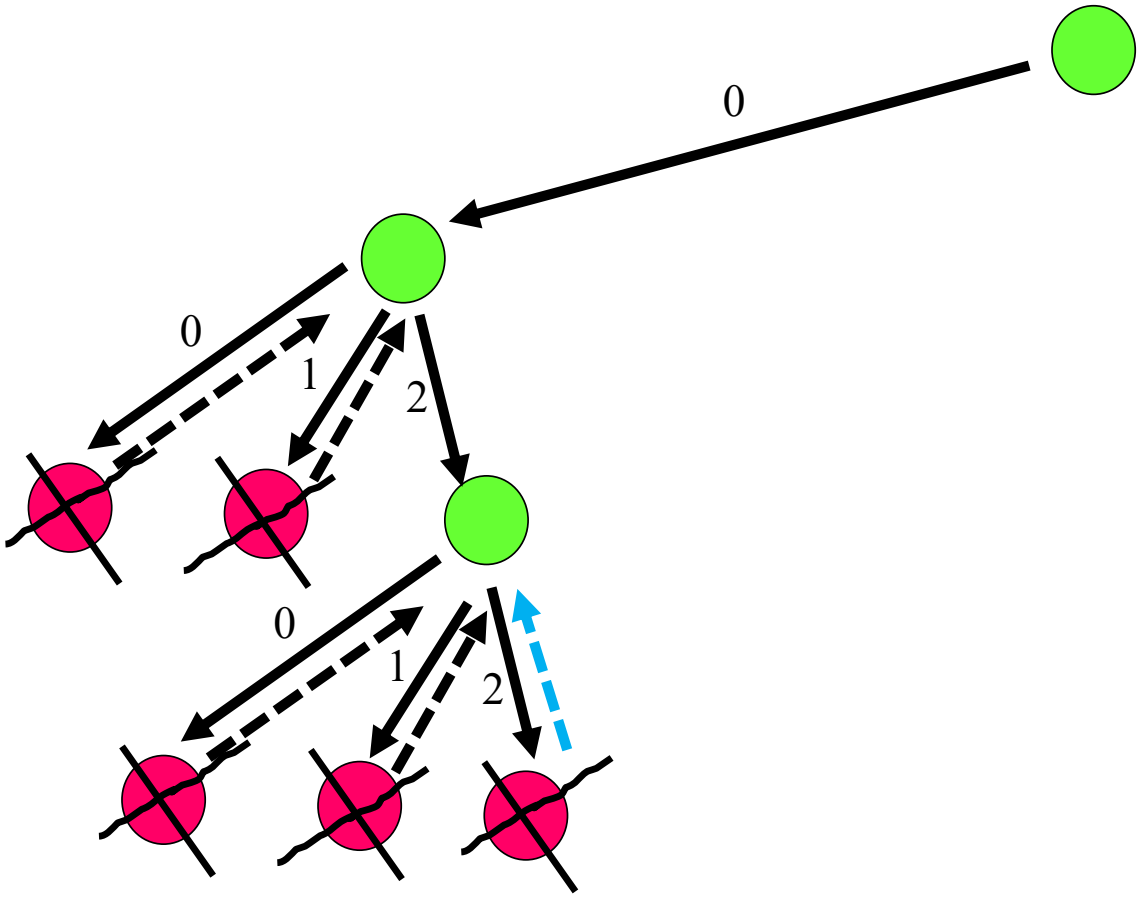
Nivel = 3

0	2	2	_



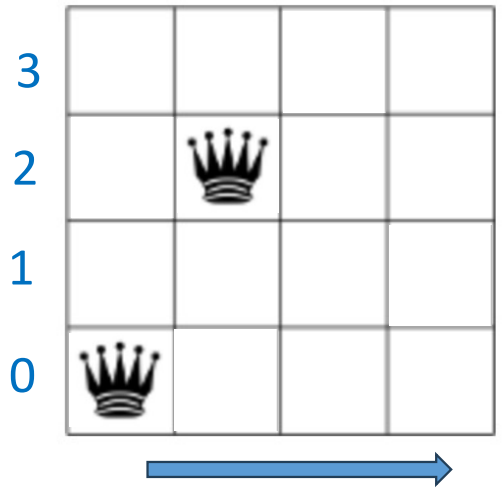
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



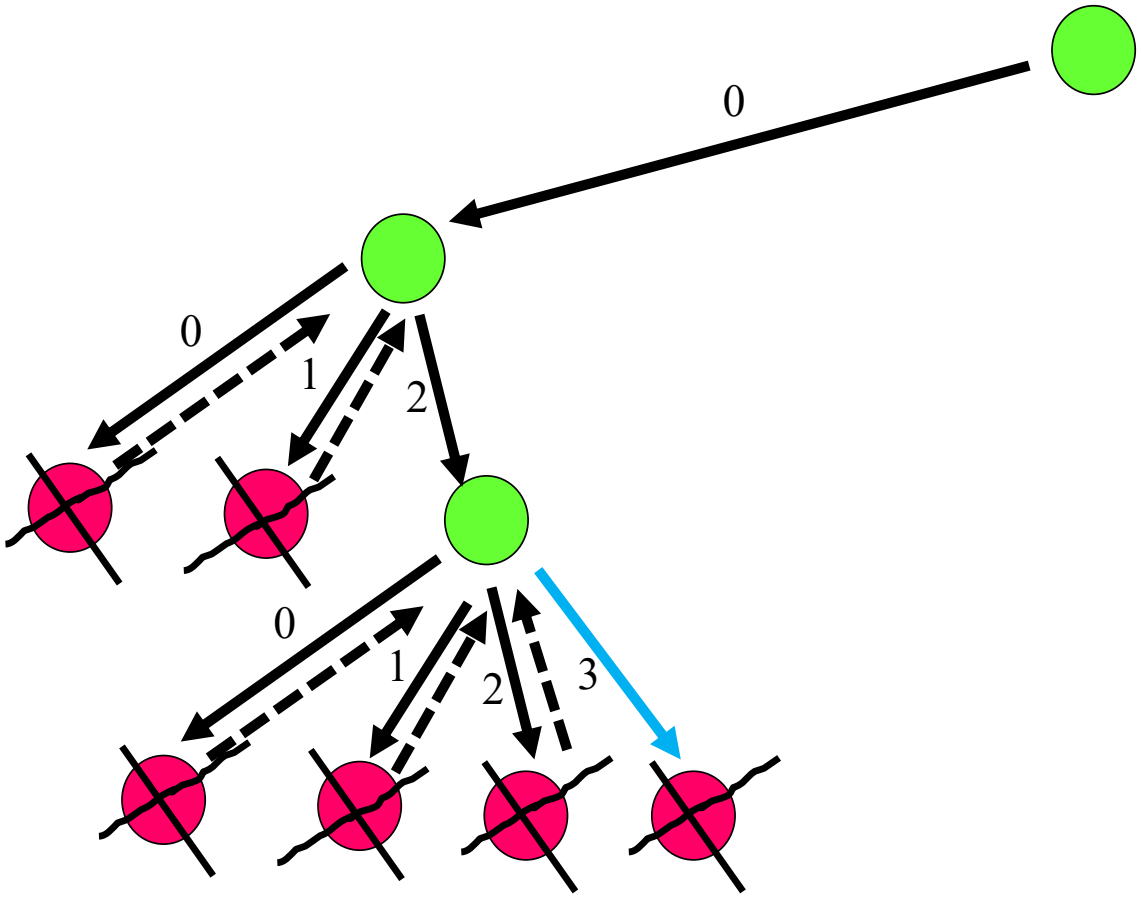
Nivel = 2

0	2	_	_



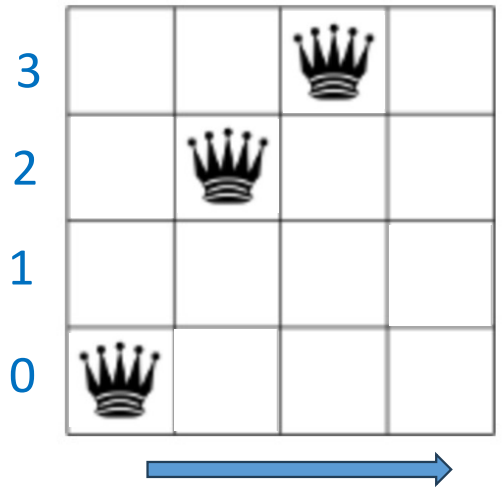
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



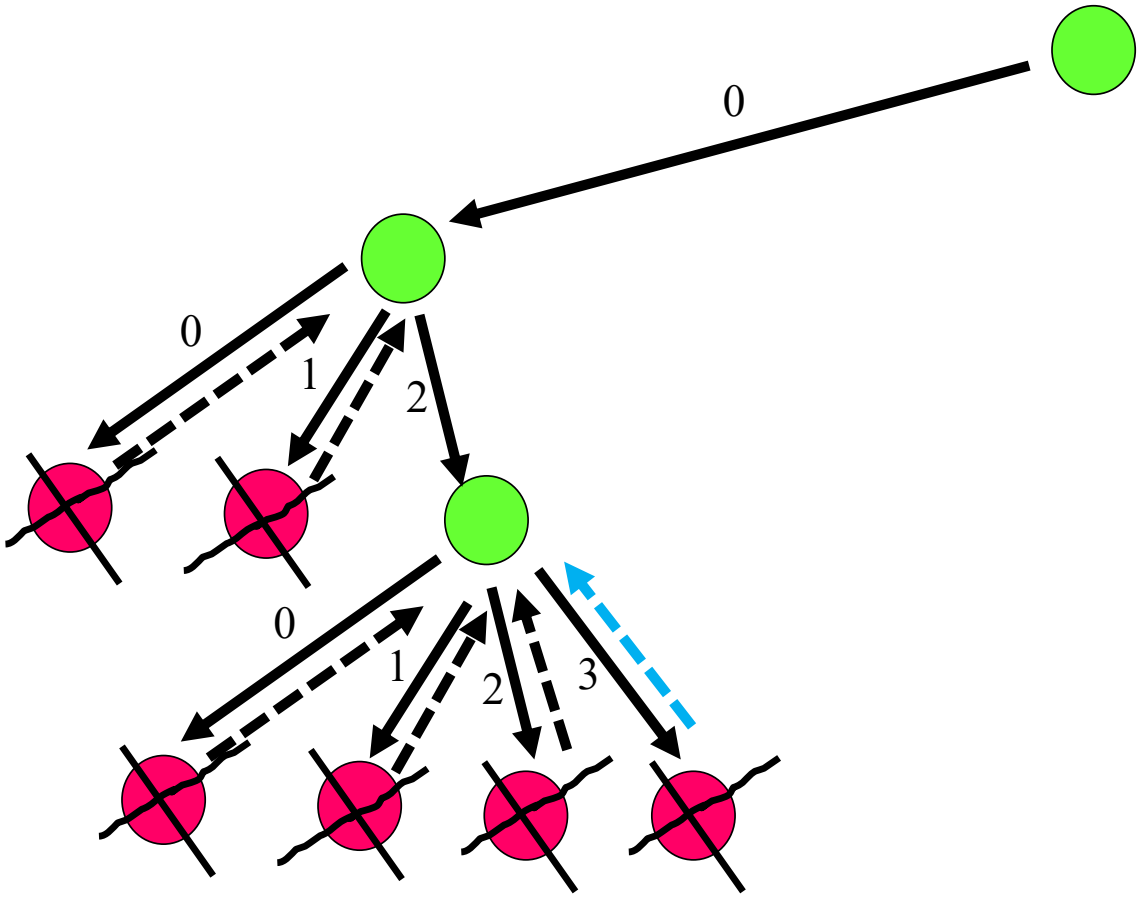
Nivel = 3

0	2	3	_



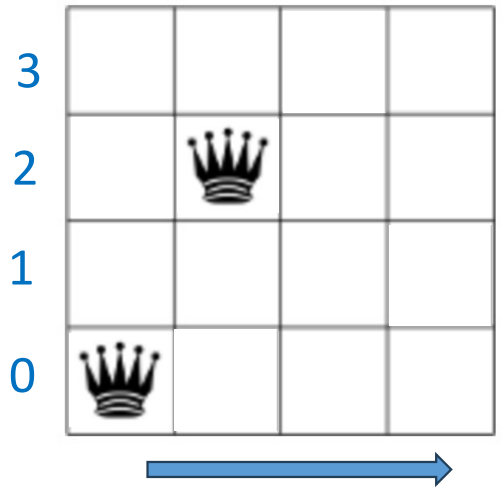
Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking



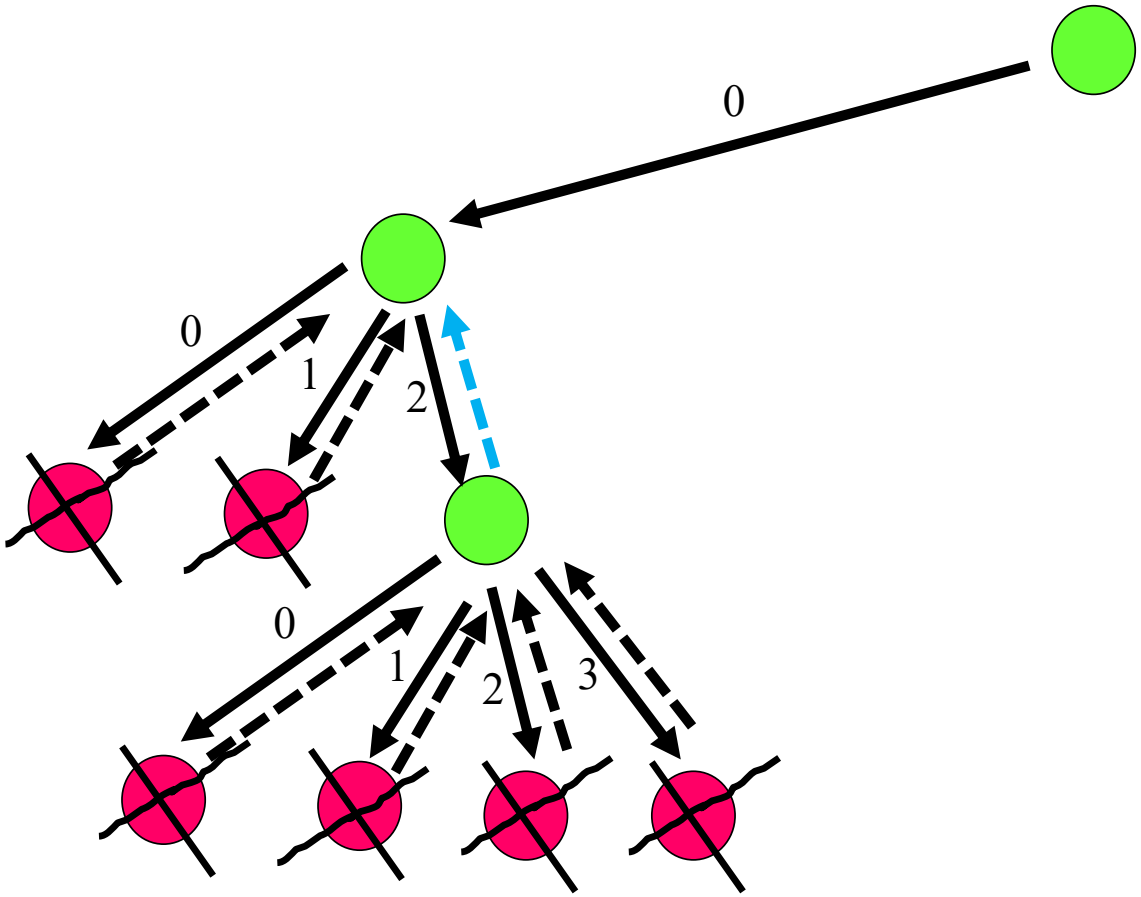
Nivel = 2

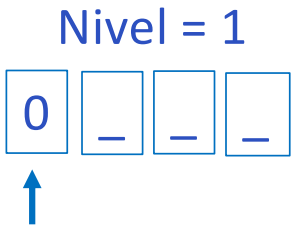
0	2	_	_

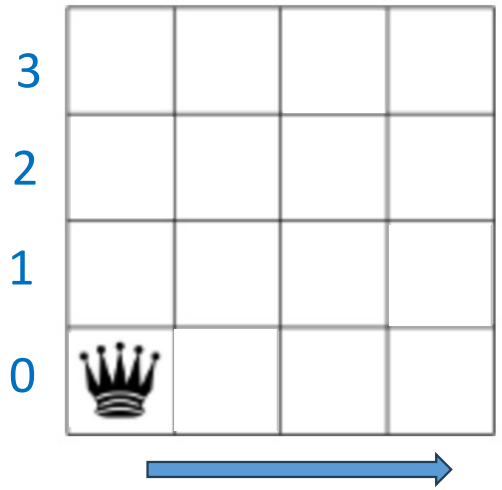


Nivel de profundidad de la recursividad

Ejemplo: Solución por Backtracking

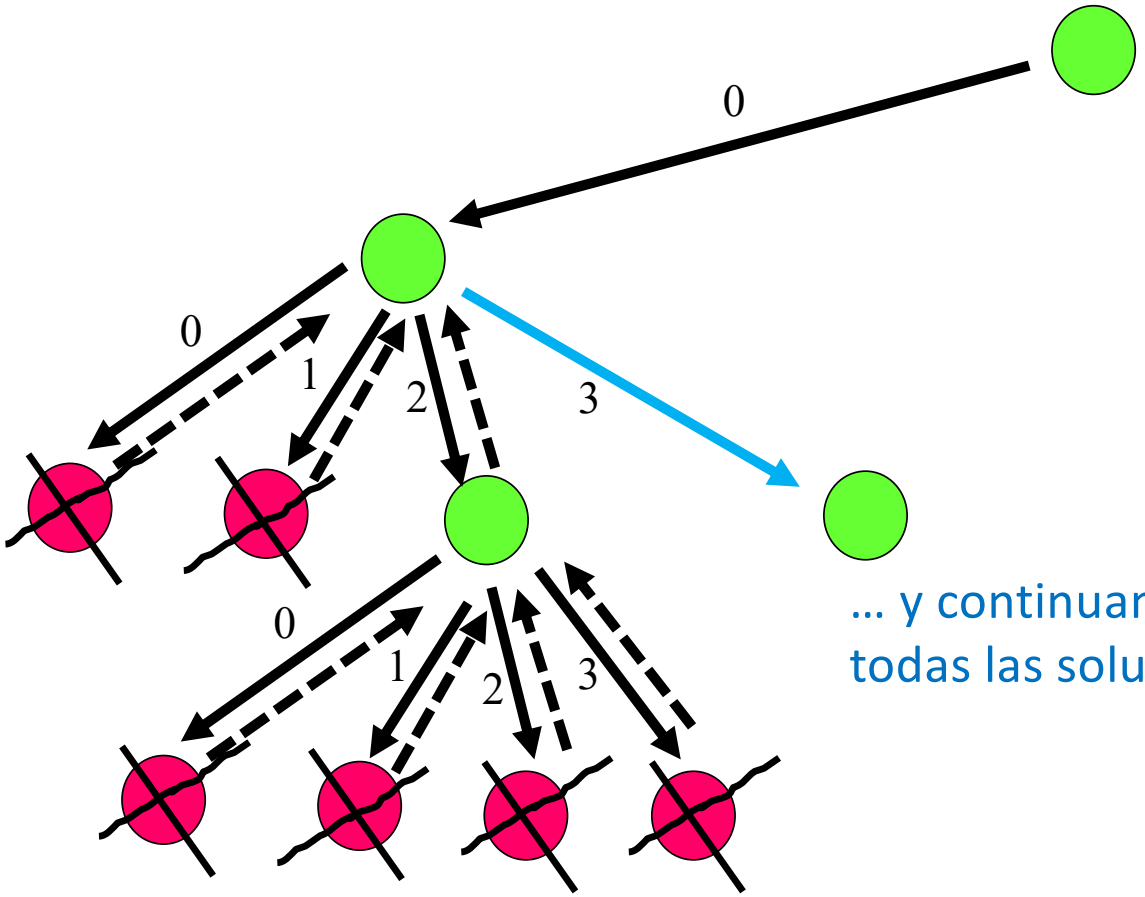






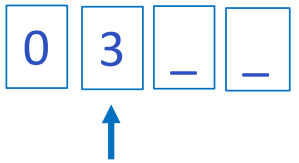
Nivel de profundidad de la recursividad

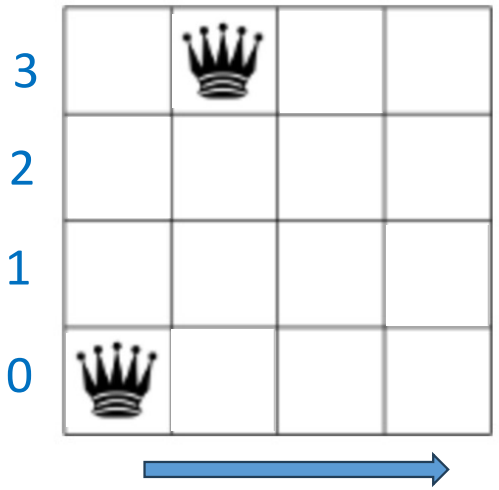
Ejemplo: Solución por Backtracking



… y continuamos buscando todas las soluciones

Nivel = 2





Nivel de profundidad de la recursividad

VPL 4: Formato del fichero de entrada

- El mismo formato de la version mediante fuerza bruta.
- El fichero sólo tiene una línea con el valor de N.

Ejemplo

4

4 reinas en un tablero de 4 x 4

Salida de nuestro programa:

[1, 3, 0, 2]
[2, 0, 3, 1]

VPL 4

solve.py

```python
def solve(num_queens):
    """
    Using backtracking compute all the solutions to place the
    given number of queens in a square board.

    :param num_queens: number of queens to place in the board
    :return: list of lists containing all the solutions

    For example, if num_queens = 4 there are two solutions,
    and it returns:
       solutions_list = [ [1, 3, 0, 2], [2, 0, 3, 1] ]

    """

    solutions_list = []

    # solve it here!


    return solutions_list
```

main.py

```python
from solve import *

first_line = input().split()
num_queens = int(first_line[0])

solutions_list = solve(num_queens)

for solution in solutions_list:
    print(solution)
```