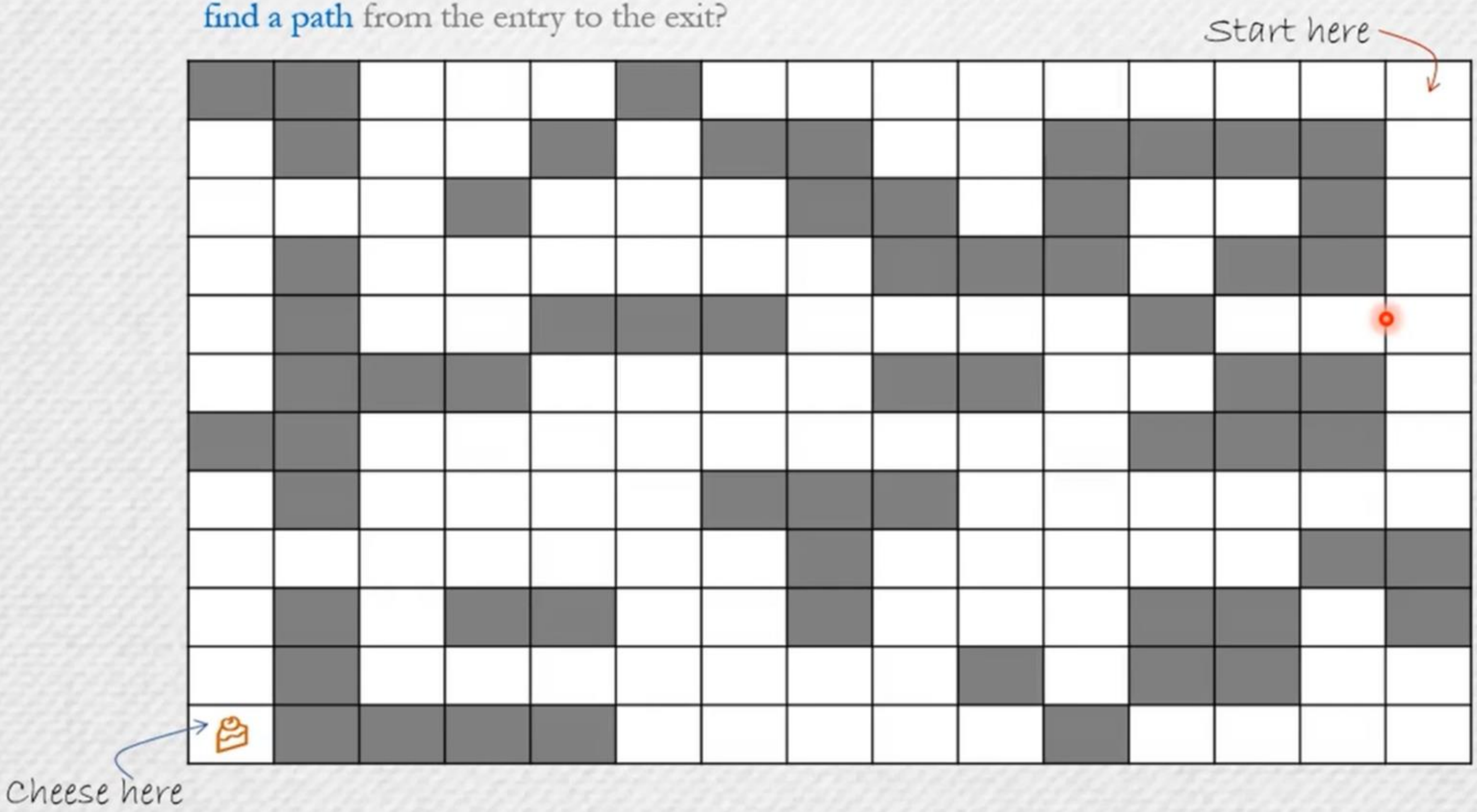
N-Queens Problem

DAA Lab 10

Rat in a Maze

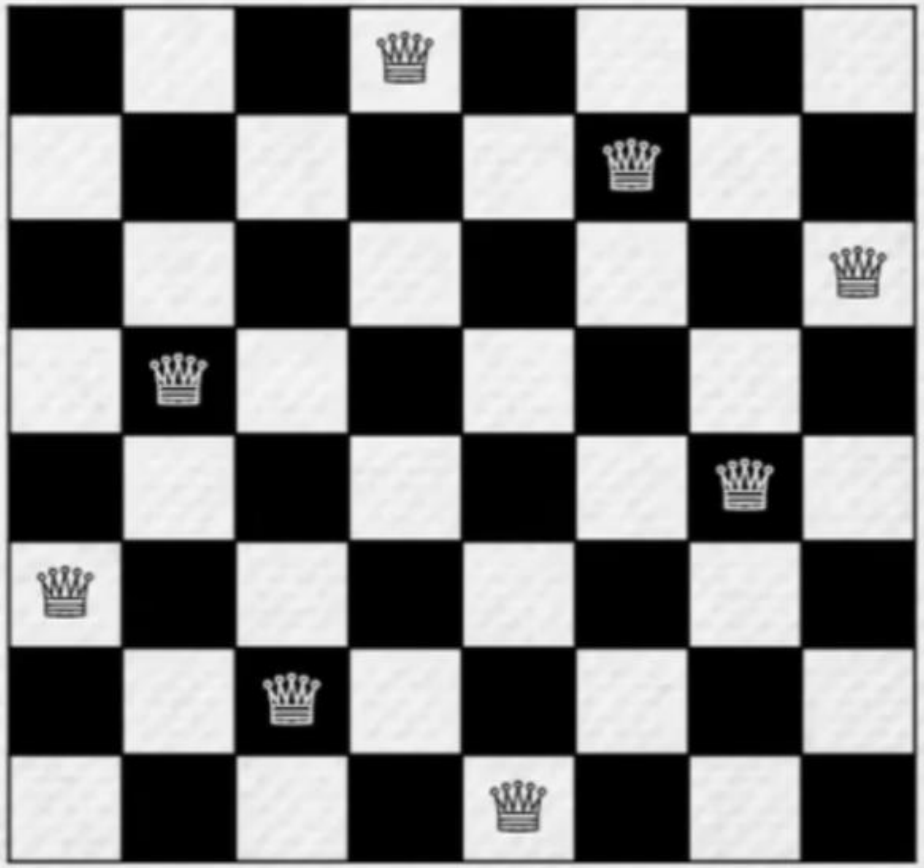
There is a rat in a maze. There is only one entry and one exit. A piece of cheese lies in the exit, but there are blocks in the path and the rat can easily take a wrong route. Can the rat find a path from the entry to the exit?

Move Order	
←	left
↓	down
→	right
↑	up



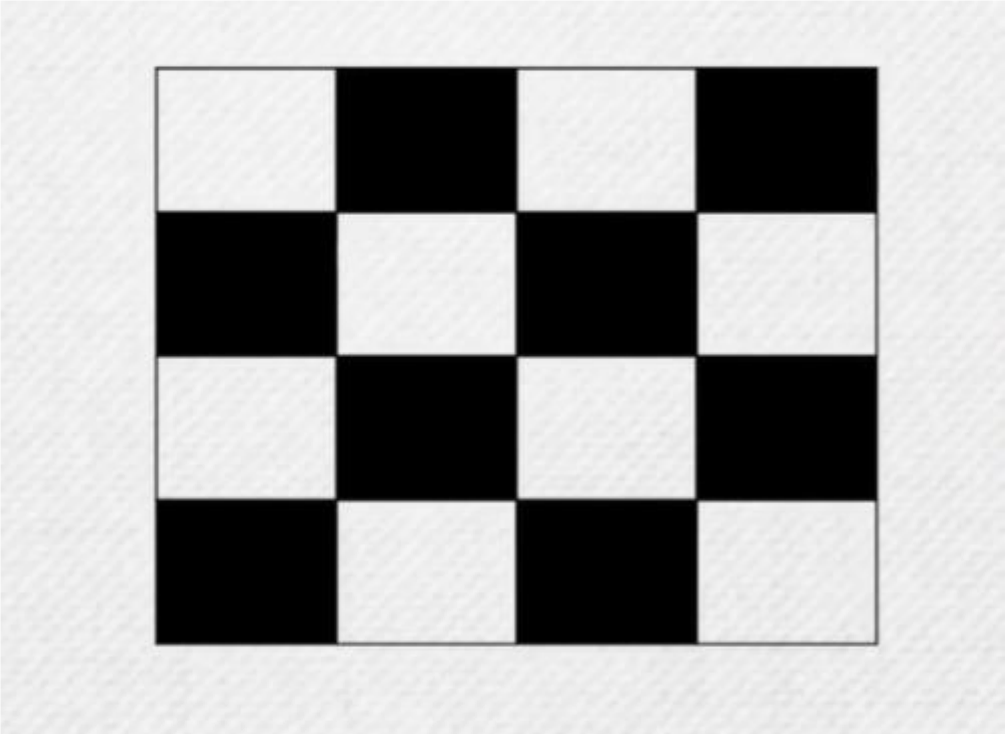
We have an 8 x 8 chessboard, our job is to place eight queens on the chessboard, so that none of them attack each other. That is, no two of them are in the same row, column or diagonal.

The generalized version of the 8 Queens problem is the n-Queens problem, where n is a positive integer.

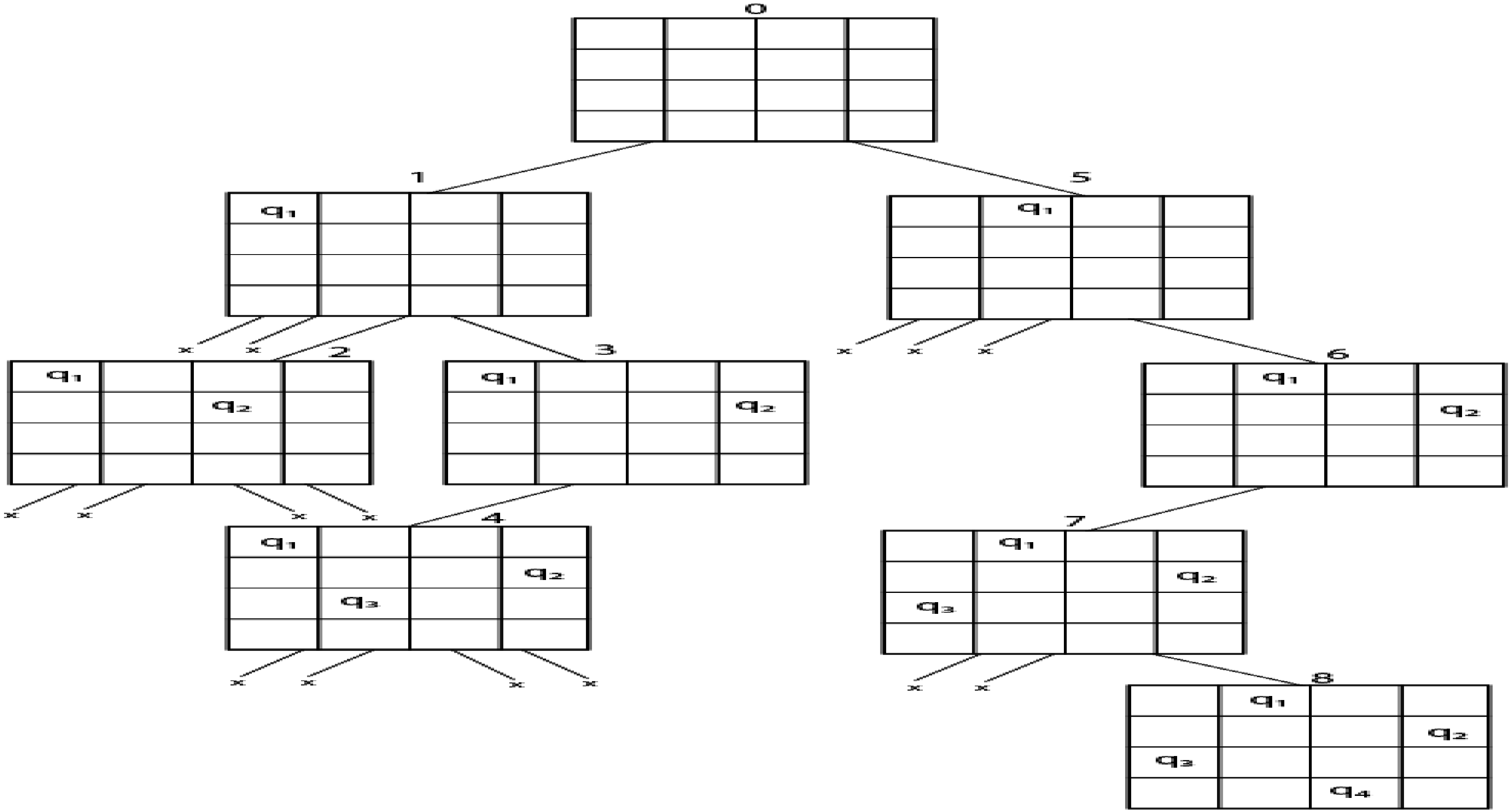


N-Queens problem

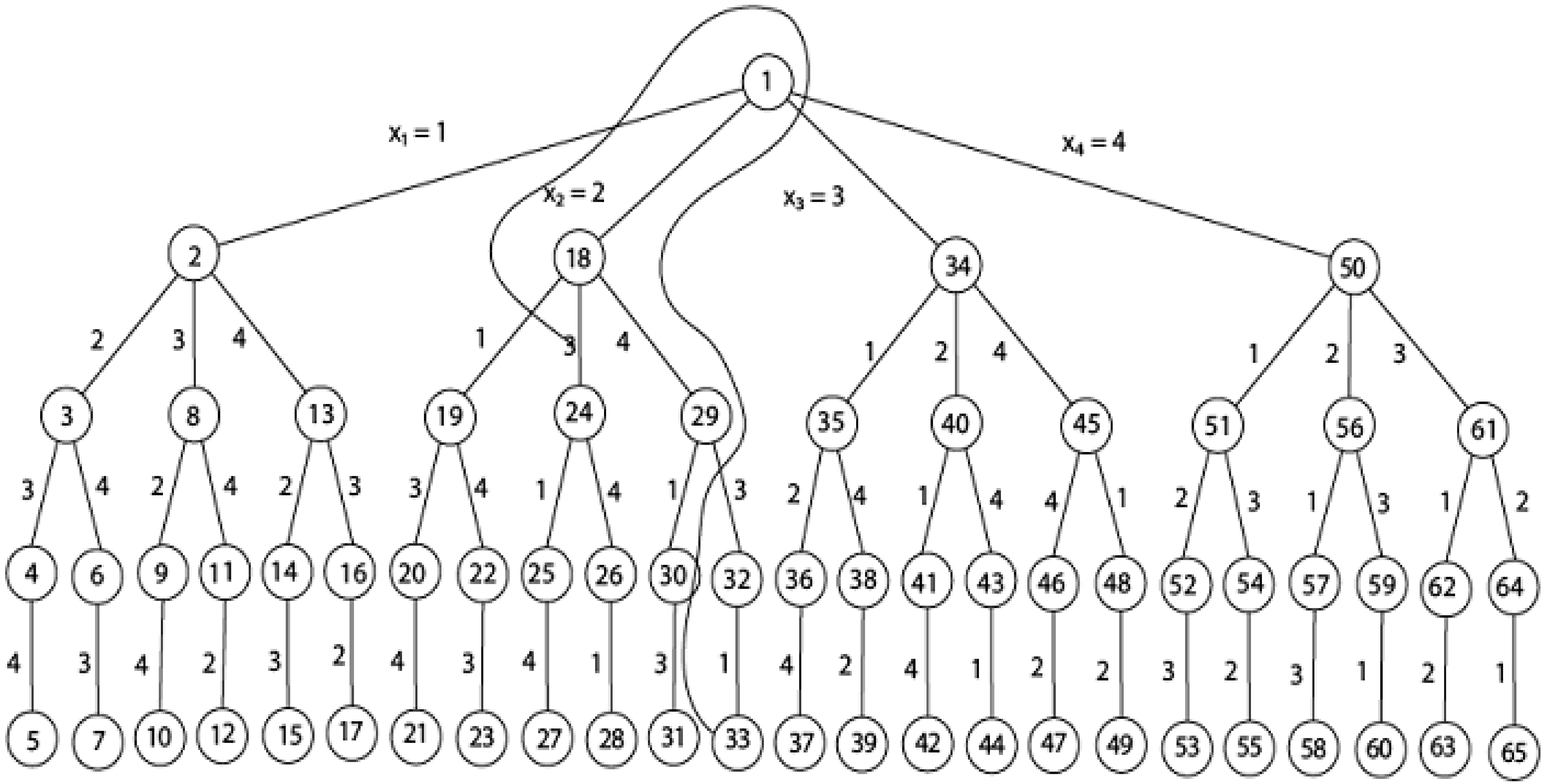
Backtracking solution for 4 - Queens



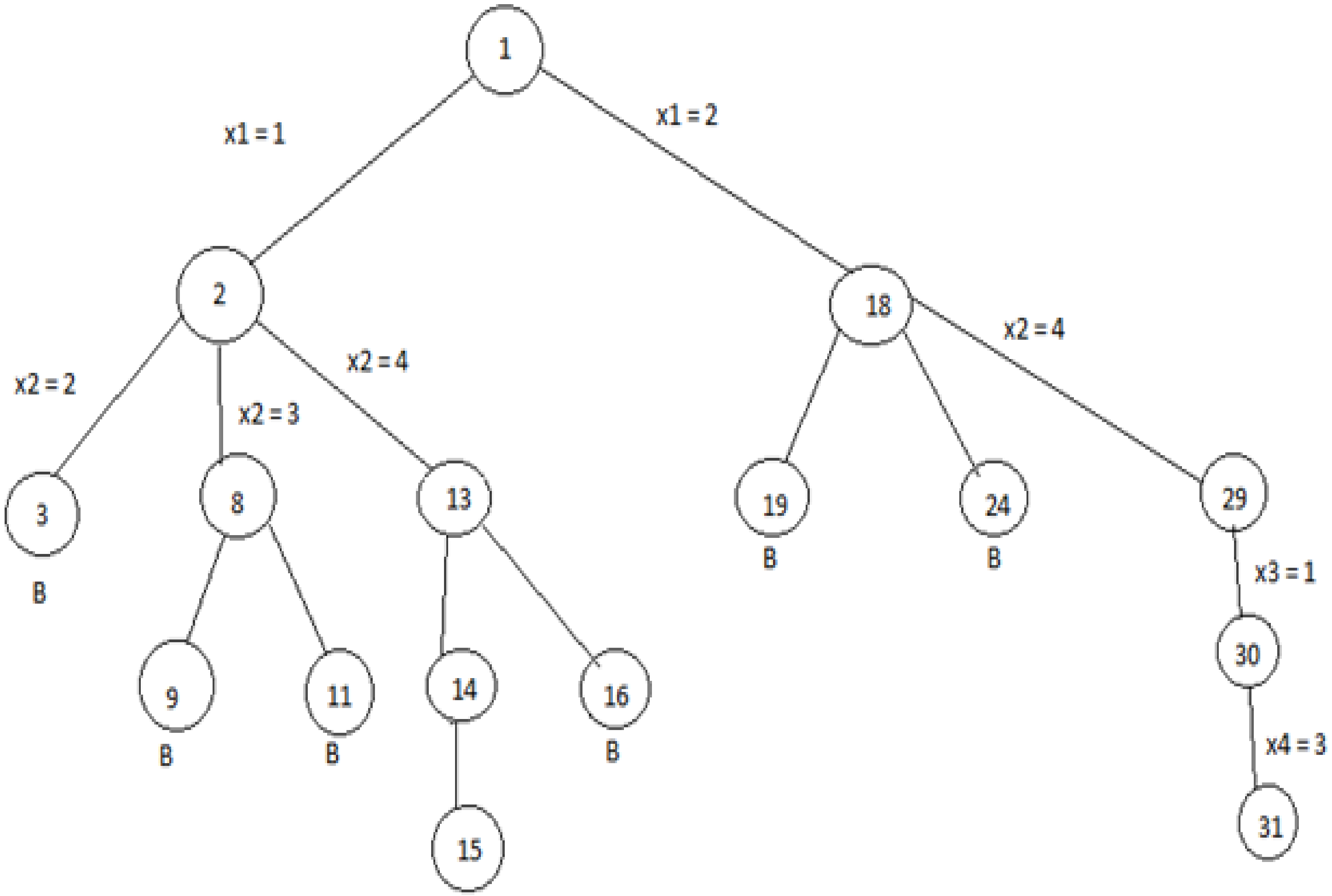
Backtracking representation



Complete state space tree for 4 Queens numbered in DFS



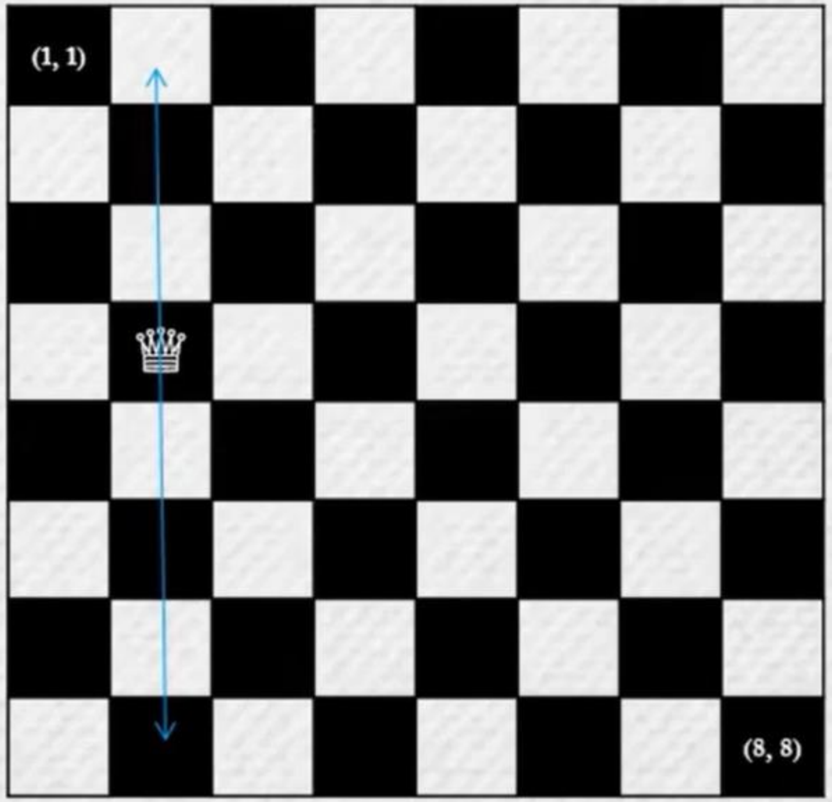
Solution for 4-Queens problem:
B denotes dead end



Think of the chessboard as a matrix of n rows and n columns.

Say, Queen1 is at position(r_1, c_1)
and, Queen2 is at position(r_2, c_2)

Under what condition does Queen1 attack Queen2 along the same column?



$$c_1 == c_2$$

Note that we need not check for the row equality because by our backtracking strategy we are only placing one queen per row. Hence there is

Detecting column attack

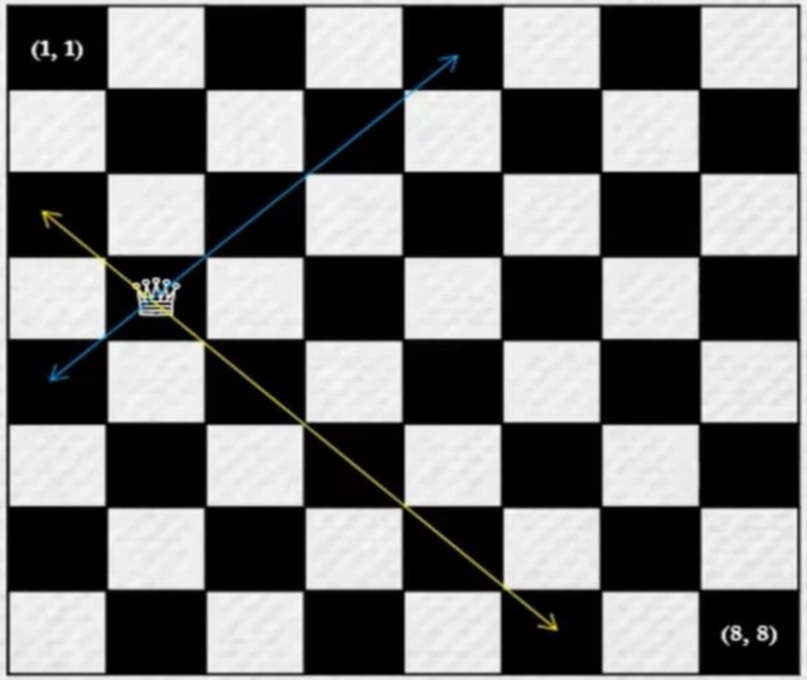
Detecting Diagonal attack

Now, say Queen1 is at (r_1, c_1) : e.g. (4, 2)

It can be attacked by Queen2 at (r_2, c_2) along two diagonals

a) Along the diagonal going from bottom-left to top-right
- Possible positions of Queen2 are (5, 1), (3, 3), (2, 4), (1, 5).
- In all the cases we observe that
- $r_1 + c_1 == r_2 + c_2$
- i.e. $\mathbf{r_1 - r_2 == c_2 - c_1}$ … (i)

b) Along the diagonal going from top-left to bottom-right
- Possible positions of Queen2 are (3, 1), (5, 3), (6, 4), (7, 5), (8, 6).
- In all the cases we observe that
- $r_1 - c_1 == r_2 - c_2$
- i.e. $\mathbf{r_1 - r_2 == c_1 - c_2}$ … (ii)



Combining (i) & (ii) we get the condition for diagonal attack as

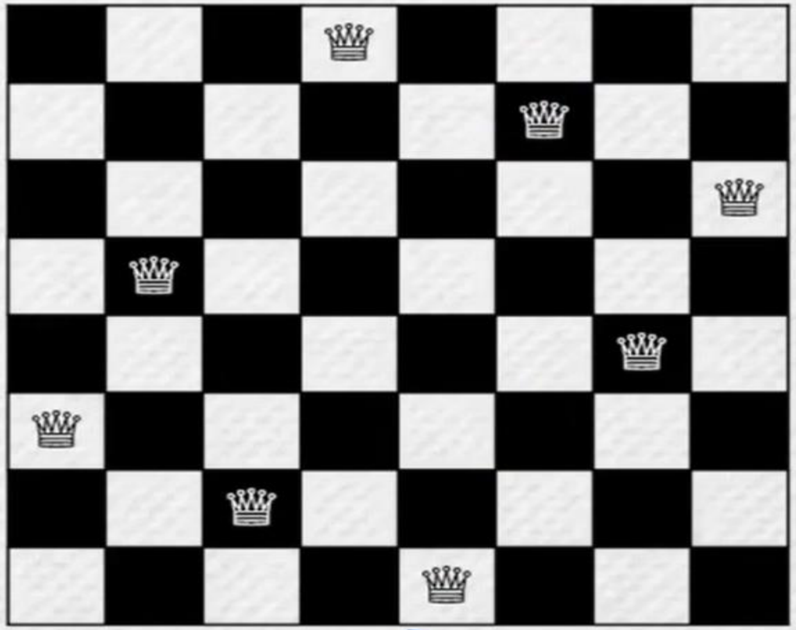
$$\mathbf{abs(r_1 - r_2) == abs(c_1 - c_2)}$$

Representing solution

n Queens on an n x n chessboard can be represented by a one dimensional array q[1 … n], where

The index j represents the row of the Queen
The corresponding value q[j] represents the column of the Queen

e.g. q[4] = 2 means queen at row 4 is at column 2,
q[2] = 6 means queen at row 2 is at column 6.



This chessboard
Is represented by this array

	1	2	3	4	5	6	7	8
q	4	6	8	2	7	1	3	5

Placing a queen on the chess board

```
Algorithm place(r2, c2)
// for all queens in the previous rows
//          if the new Queen at (r2, c2) attacks along a column or a diagonal
//                   This Queen can't be placed at (r2, c2)
// otherwise (r2, c2) is a good position to place the Queen
    for (r1 = 1 to r2 - 1)
        c1 = q[r1]                                    // column of the Queen at row r1
        if (c1 == c2 OR abs(r1 - r2) == abs(c1 - c2))
            return false
    return true                                       // since we are out of the for loop this must be a valid place
```

N-Queens Algorithm

```
Algorithm nQueens(r)
    for (c = 1 to n)
        if (place(r, c))
            q[r] = c
            if (r == n) displayQueens()
            else nQueens(r + 1)
```

Complexity

- The worst case “brute force” solution for the **N**-**queens puzzle** has an O(**n**^**n**) time **complexity**. ... However, if it is found that **N** number of **queens** cannot be placed on that board, it will backtrack and try another safe position. This is over 100 times as fast as brute force and has a time **complexity** of O(2^**n**).

Prelab Questions:

1. Define backtracking and applications of backtracking approach
2. Define live node, dead node.
3. Define implicit and explicit constraints.
4. What is the time complexity of n-queens problem.
5. Draw the generalized and solution State Space Tree of 4-queen’s problem