Sudoku

Backtracking

N Queens problem

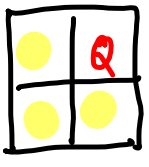
Chess board N x N

Queen attacks all squares on its row, column, diagonals

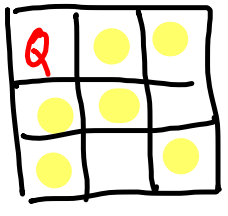
Place N queens so that they do not attack each other

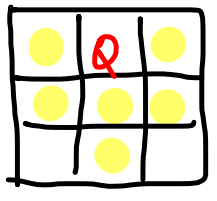
N=1

N=2

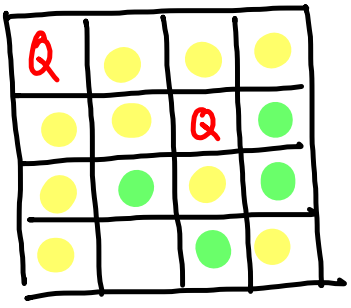


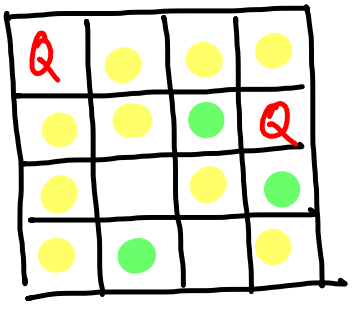
N=3

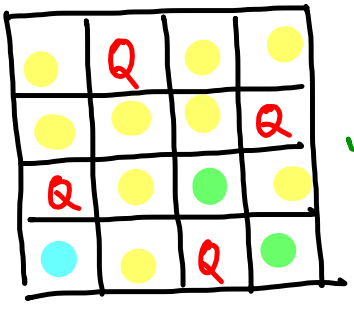




N=4







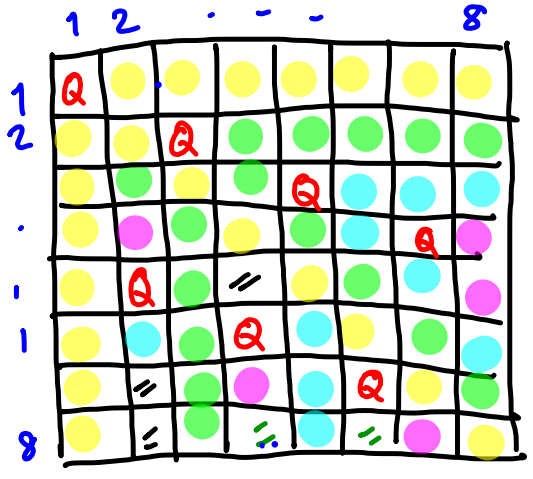
✓

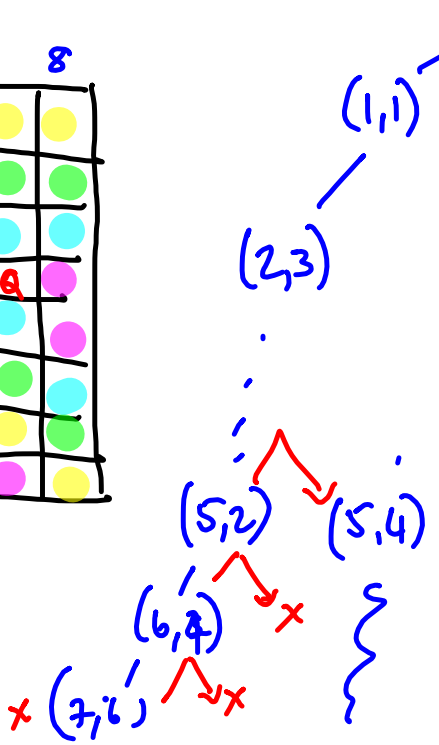
Claim

Every N ≥ 4 can be solved

Solve N Queens for N ≥ 4

N = 8





Exhaustively searching tree of all poss. solutions

How to do this effectively?

Backtracking

- Add a queen
- Remove a queen

} update the current state of the board

What is a good representation for the state of the board

N x N grid of {0,1}

0 - no queen, 1 - queen

Know that we can have only 1 queen per row/col

Column with entries [1..N]

board[i] = j Queen at (i,j)

Code for backtracking

```
def placequeen(i, board): // Place queen i, in row i
    for each c s.t. (i,c) available
        place queen i at (i,c) & update board
        if i==N:
            return(True)
        else:
            attempt = placequeen(i+1, board)
            if attempt:
                return(True)
            else: undo this move and update board
    else: return(False)
```

global data

undo the earlier update

Board representations

N x N grid

- 0 - no queen
- 1 - Queen

Recording free vs attacked squares

- Attack (not available)
- Free (available)

N column

j - queen i in col j

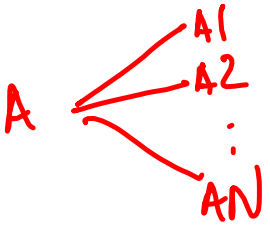
Add a queen at (i,j) - update all entries

Remove a queen? Which A become F

Remember which queen attacked which square

Exploit the order in which queens are added/removed

Record earliest queen that attacks each square



Remove queen i, reset all Ai to F

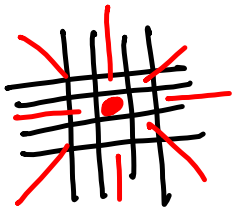
N^2 representation of board that records attacked & free squares, that can be updated when queen is added/removed

Can we do this with O(N) space?

How many queens attack row i? Only 1

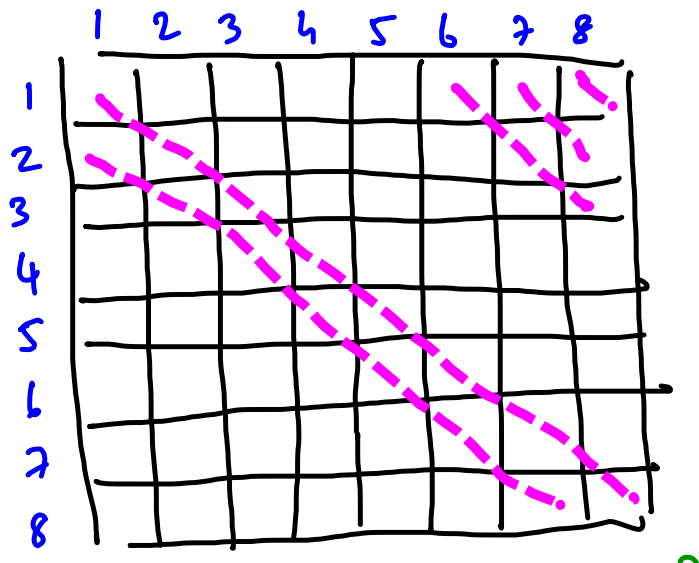
" " " " col j? Only 1

A square can be attacked by upto 4 queens



How many queens attack a fixed diagonal d? Only 1

How many diagonals are there?

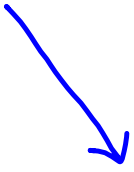


↘ diagonals
2N-1
diagonals

↗ diagonals
2N-1

Record attacked positions in terms of rows, cols, diags

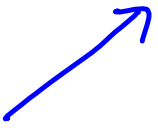
(i, j) is attacked if row(i) attacked or col(j)
or ↗diag(?) or ↘diag(?)



(c-r) is invariant

Number diagonals $-7, \ldots, 0, \ldots +7$

NW to SE



SW to NE

(c+r) is invariant

Number diagonals 2 .. 16

(i,j) is attacked if NWtoSE(j-i) or SWtoNE(i+j) is attacked

O(N) representation

board[1..N] board[i]=j Queen i at (i,j)

row[1..N] <0, not attacked
1, attacked

col[1..N] <0,

NWtoSE[-N-1..+N-1] <0,

SWtoNE[2 ... 2N] <0,

Add a queen at (i,j) Remove from (i,j)

Board[i] = j → 0

Row[i] = 1 → 0

Col[j] = 1 → 0

NWtoSE[j-i] = 1 → 0

SWtoNE[j+i] = 1 → 0

Use this to update board in the earlier code

All solutions ?