# Backtracking

Textbook: Chapter 12.1

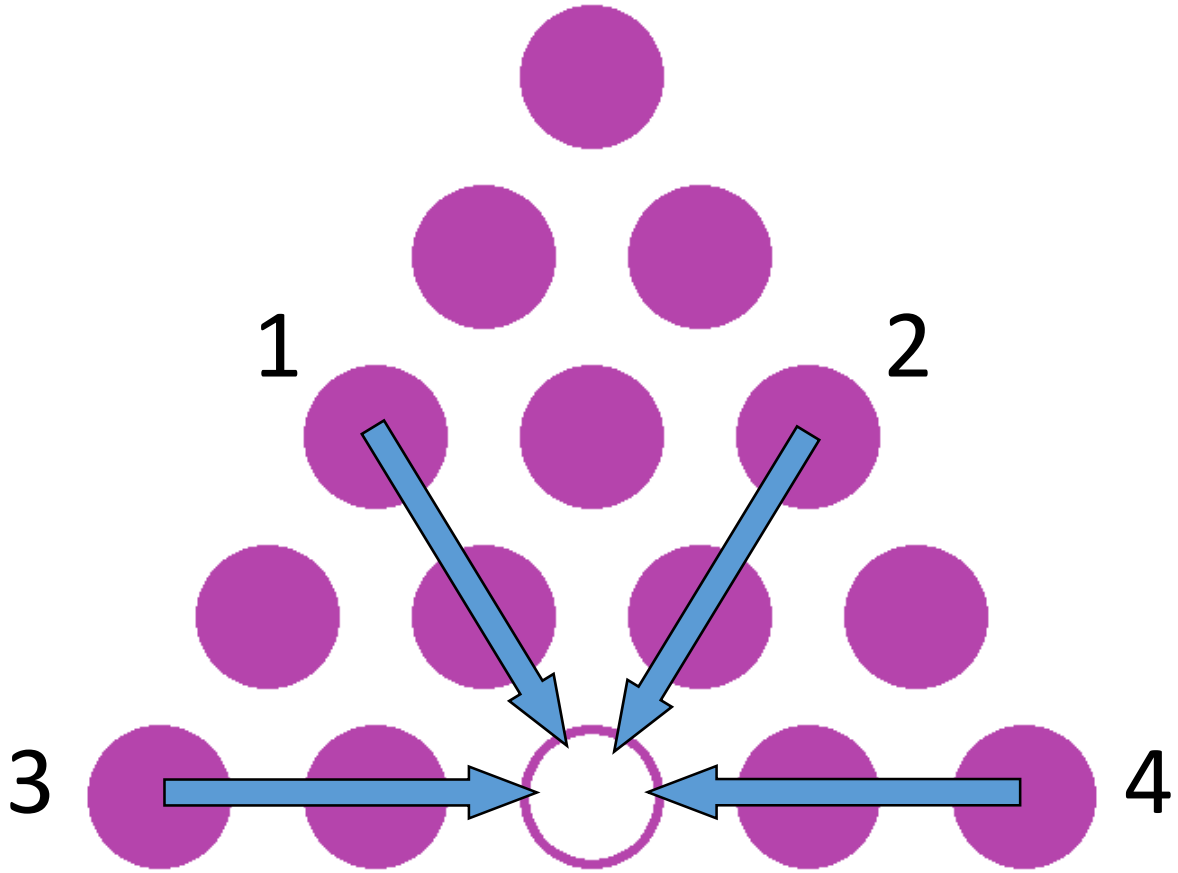
# Backtracking

- Suppose you have to make a series of *decisions,* among various *choices,* where
  - You don't have enough information to know what to choose
  - Each decision leads to a new set of choices
  - Some sequence of choices (possibly more than one) may be a solution to your problem
- Backtracking is a methodical way of trying out various sequences of decisions, until you find one that "works"

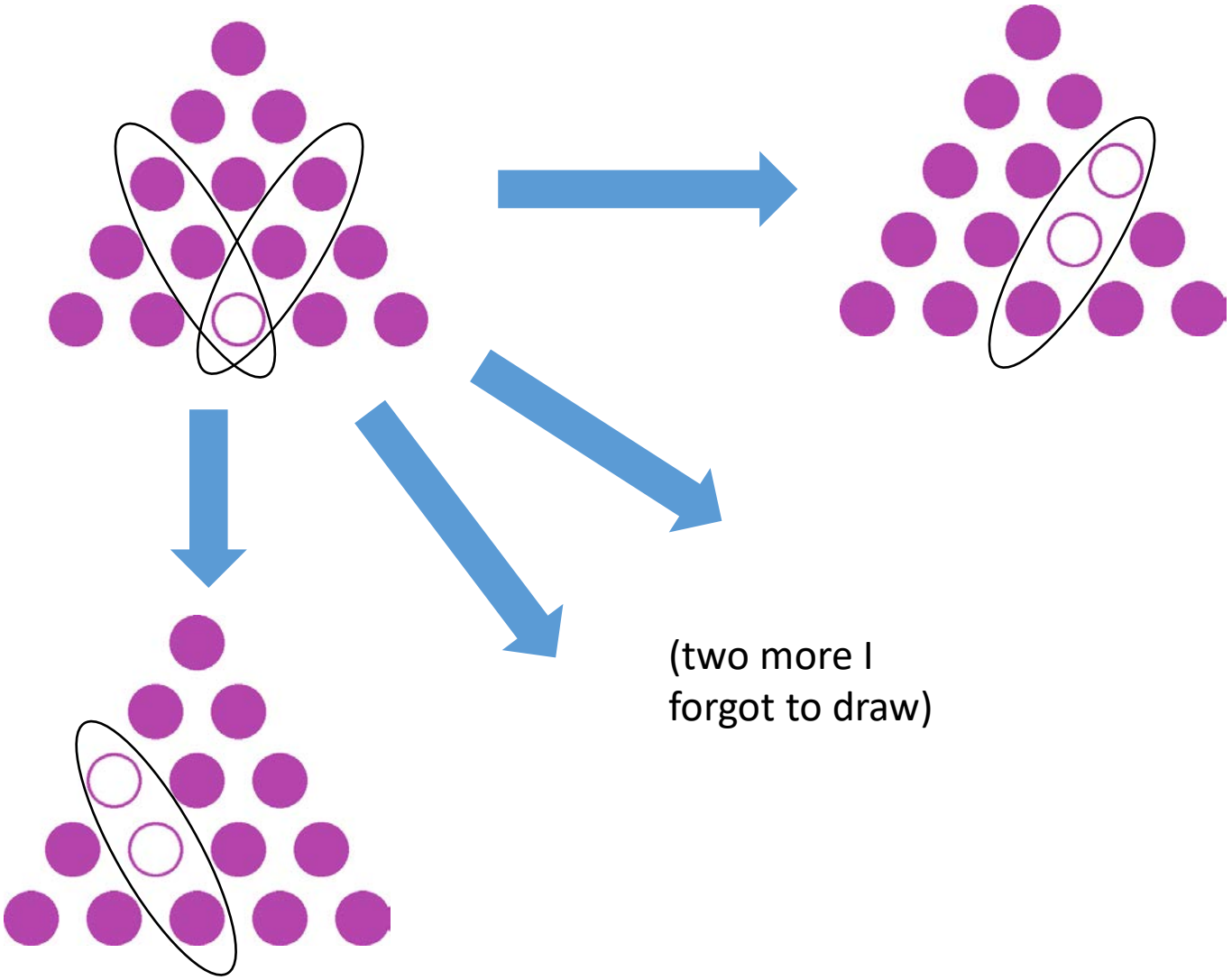
# Golf-tee puzzle

# Valid moves

This position has four valid moves:

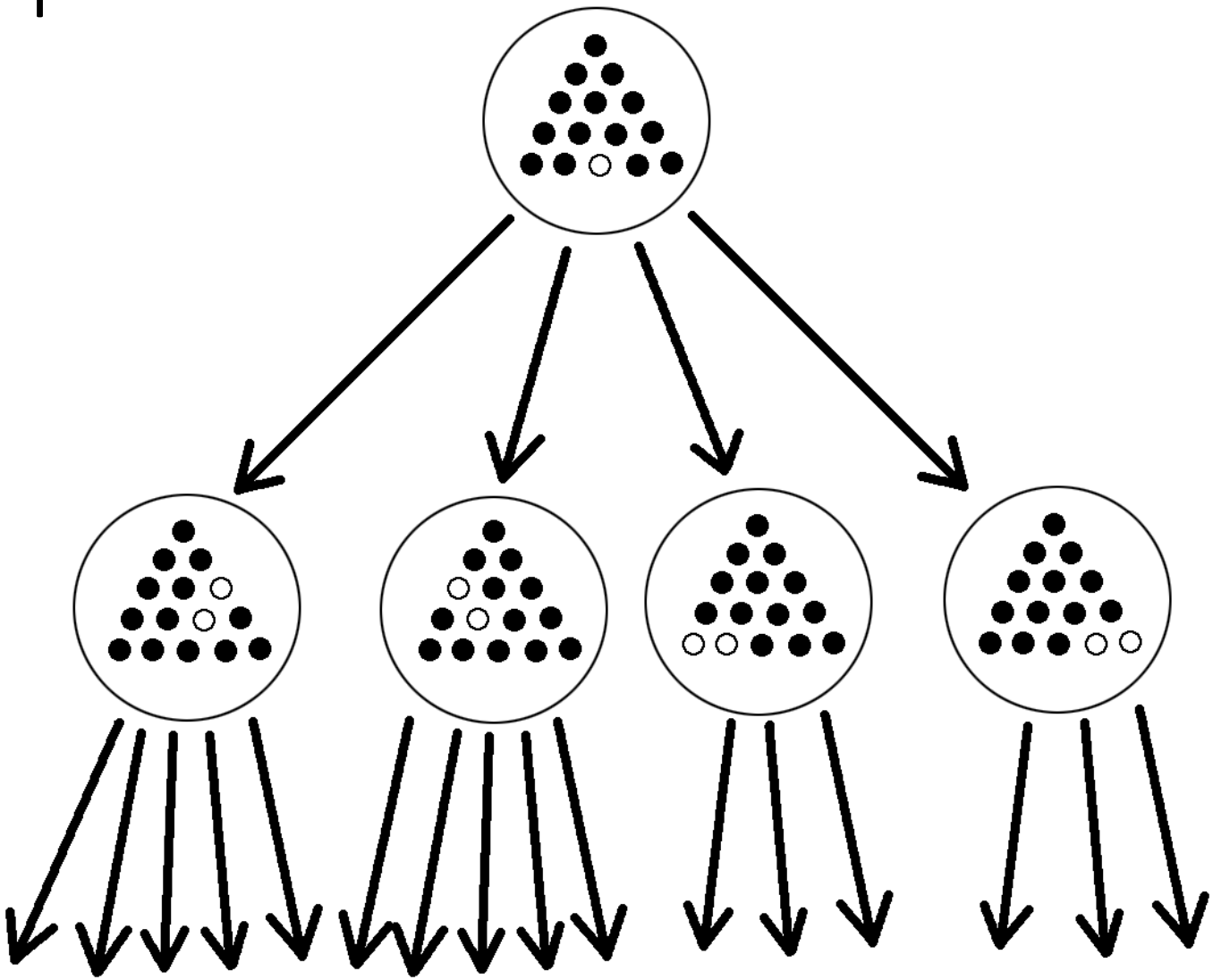


# Changing state



(two more I forgot to draw)

# State-space tree



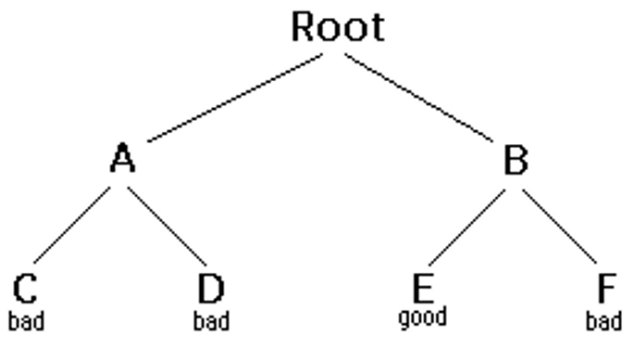
So many more places to go from here!

# Backtracking in words

- IDEA:
  - Construct solutions one component at a time
  - If a partial solution can be developed further without violating constraints:
    - Choose first legitimate option for the next component
  - If there is *no option* for the next component
    - Backtrack to replace the last component of partial solution

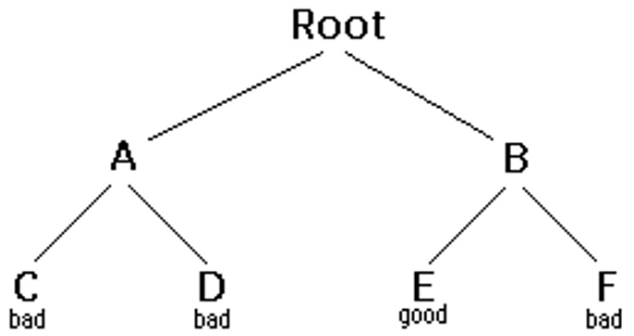
# Backtracking

- Think of the solutions as being organized in a tree
  - The root represents initial state before the search begins
  - Nodes at first level represent first choice
  - Second… second choice..etc

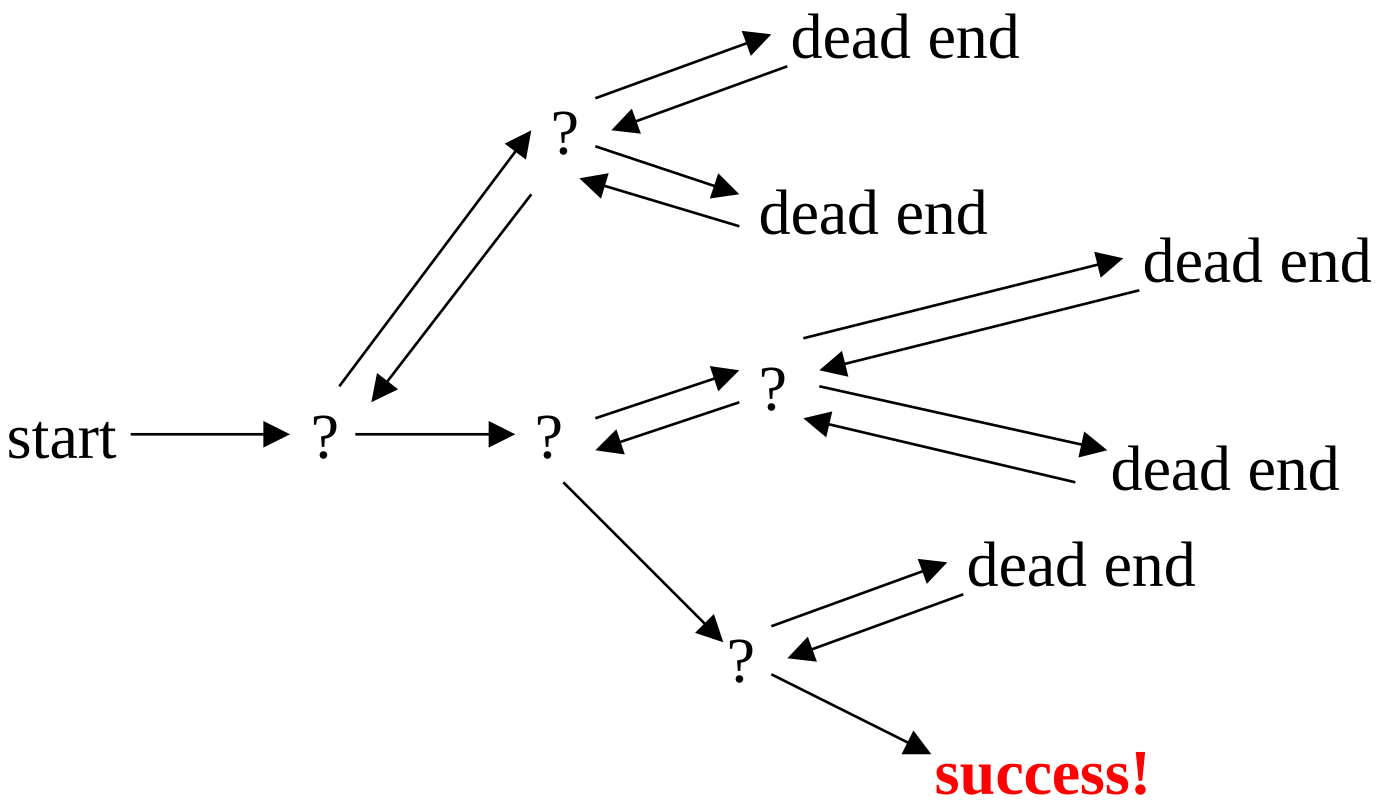


# Backtracking – Abstract Example

- Starting at Root, your options are A and B. You choose A.
- At A, your options are C and D. You choose C.
- C is bad. Go back to A.
- At A, you have already tried C, and it failed. Try D.
- D is bad. Go back to A.
- At A, you have no options left to try. Go back to Root.
- At Root, you have already tried A. Try B.
- At B, your options are E and F. Try E.
- E is good. Congratulations!



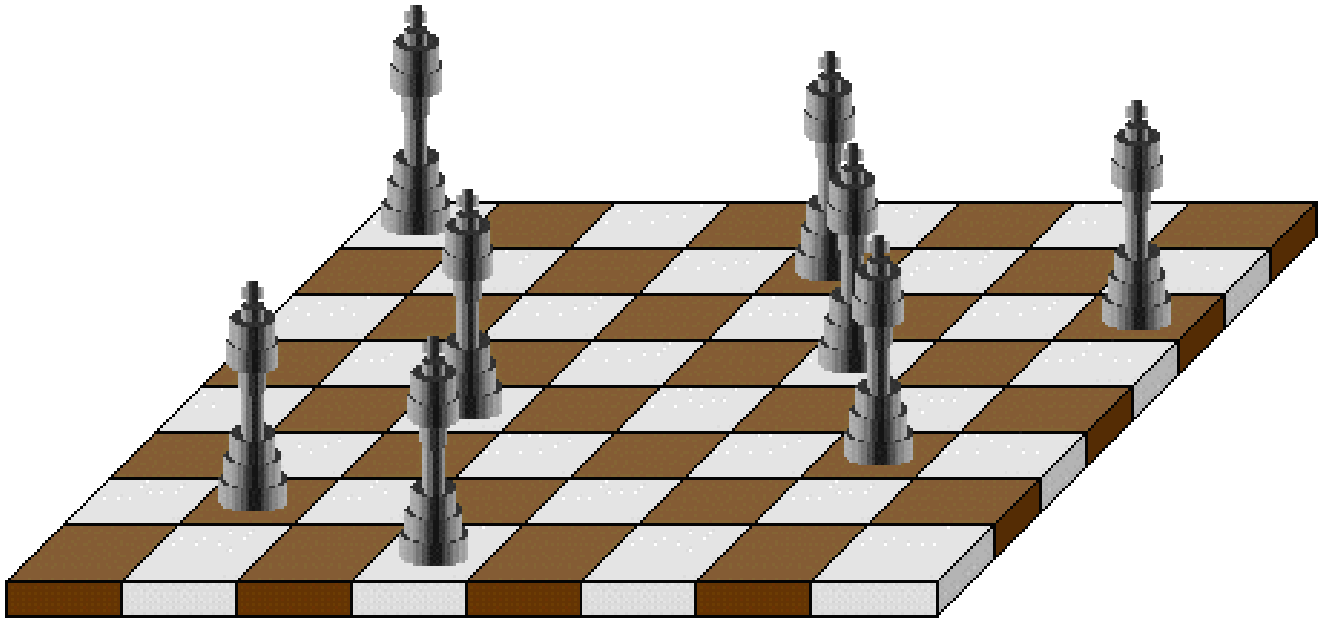
# Backtracking (animation)




The tree used to build solutions is called the *state-space tree*
The nodes are *partial solutions*
The edges are *choices*

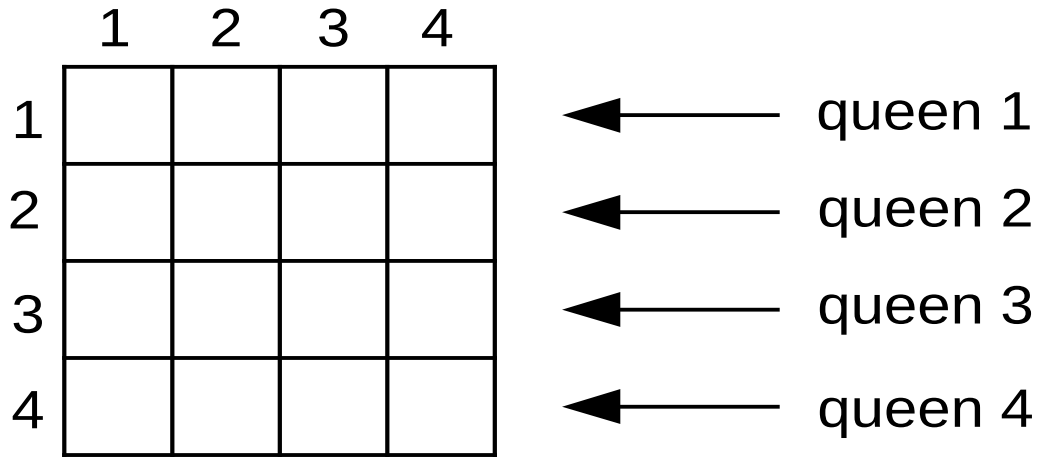
# Example: n-Queens Problem

- Place *n* queens on an *n*-by-*n* chess board so that no two are in the same row, column or diagonal
  - i.e. no queens are attacking each other



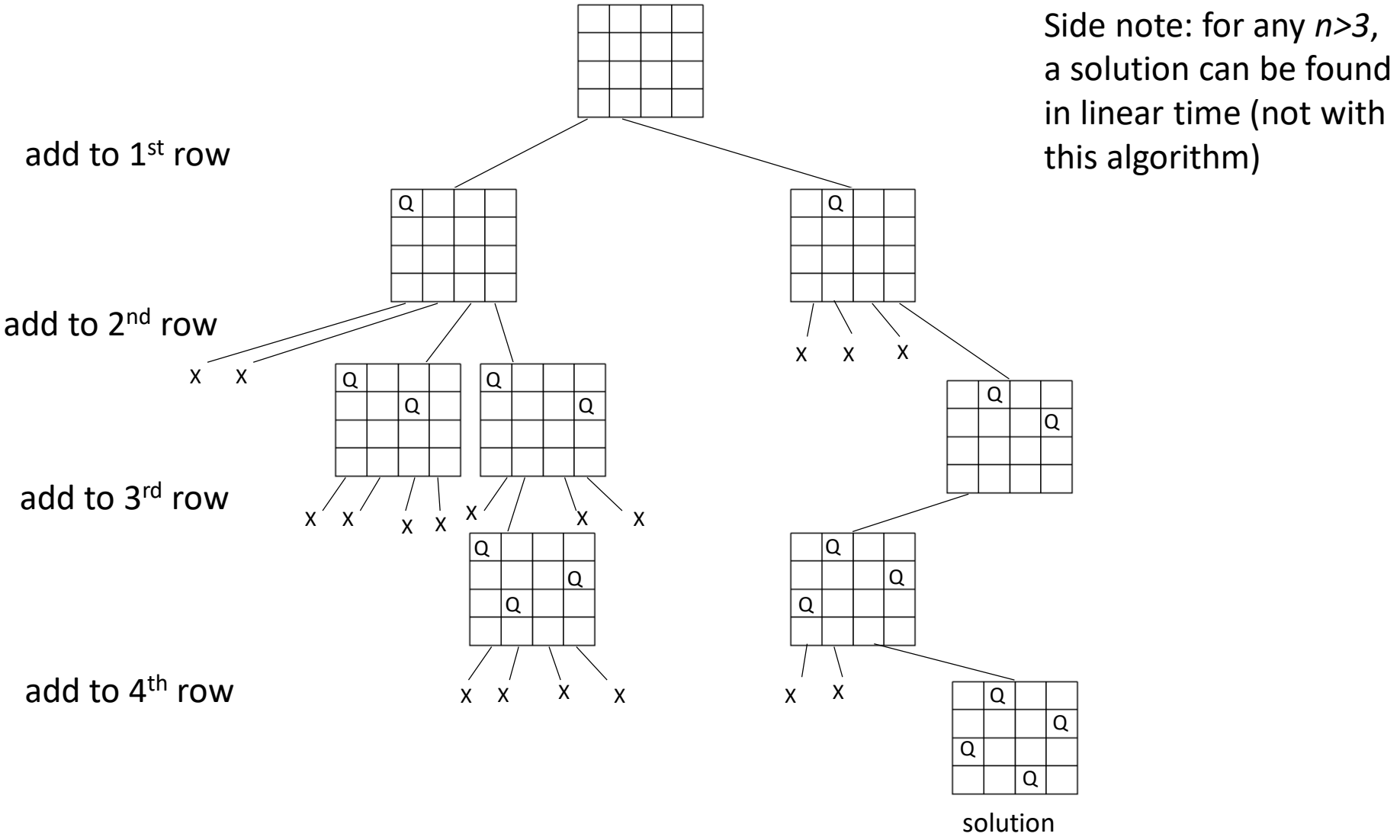
# Example: 4-Queens

- n=4




- We can solve it by backtracking
  - Root is empty board
  - At level i… put a queen in row i

# State-Space Tree of 4-Queens




Side note: for any *n>3*, a solution can be found in linear time (not with this algorithm)

# Example: Hamiltonian cycles