Learn DAA: From B K Sharma

TCS-503: Design and Analysis of Algorithms

Advanced Design and Analysis Techniques: Dynamic Programming

Unit III

- Advanced Design and Analysis Techniques:
  - Dynamic Programming
  - Greedy Algorithms
  - Amortized Analysis
  - Backtracking.

Learn DAA: From B K Sharma

Advanced Algorithm Design Techniques: Optimization Techniques

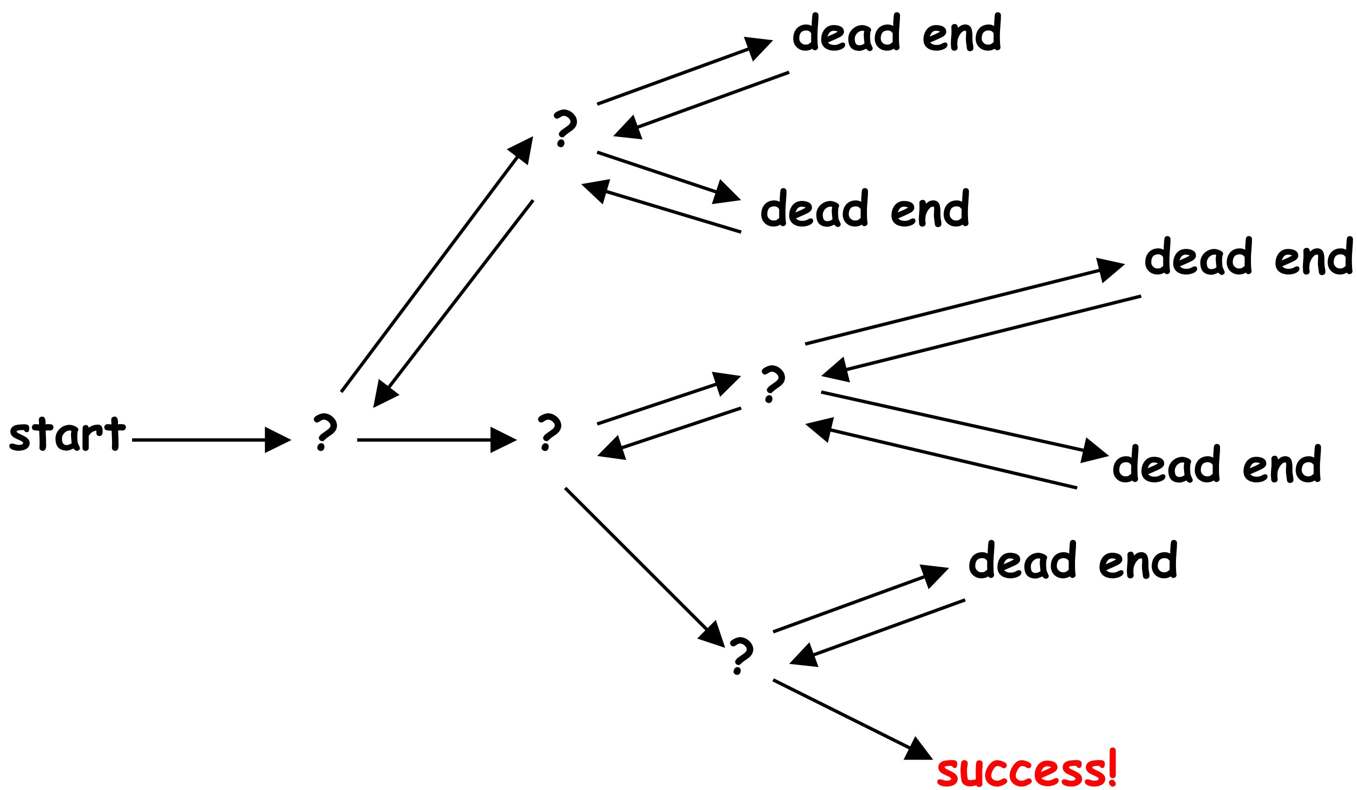
- Dynamic Programming
- Greedy Algorithms
- Backtracking

Advanced Algorithm Analysis Technique

- Amortized Analysis

Advanced Algorithm Design Techniques: Optimization Techniques

- Dynamic Programming
- Greedy Algorithms
- Backtracking



Just Try All Solutions.

Try all options for first choice,

for each option, recursively make other choices.

Can be applied to almost all problems,

but gives very slow algorithms..

What is Backtracking?

Backtracking is a technique used to solve problems with a large search space, by systematically trying and eliminating possibilities.

The backtracking strategy says to try each choice, one after the other,

if you ever get stuck, *"backtrack"* to the junction and try the next choice.

If you try all choices and never found a way out, then there IS no solution to the problem.

Learn DAA: From B K Sharma

Backtracking : Examples

Sum of Subsets Problem

AND

n-Queens Problem

Backtracking : Example

Sum of Subsets Problem

Given *n* positive integers $w_1, w_2, w_3, \ldots\ldots w_n$ and a positive integer S.

For example , given three positive integers $w_1=2$, $w_2=4$, $w_3=6$ and a positive integer $S=6$.

Find all subsets of $w_1, w_2, w_3, \ldots\ldots w_n$ that sum to S.

Subsets=

$\{\ \},\{w_1\},\{w_2\},\{w_3\}, \{w_1,w_2\},\{w_1,w_3\}, \{w_2,w_3\},\{w_1,w_2,w_3\}$

=

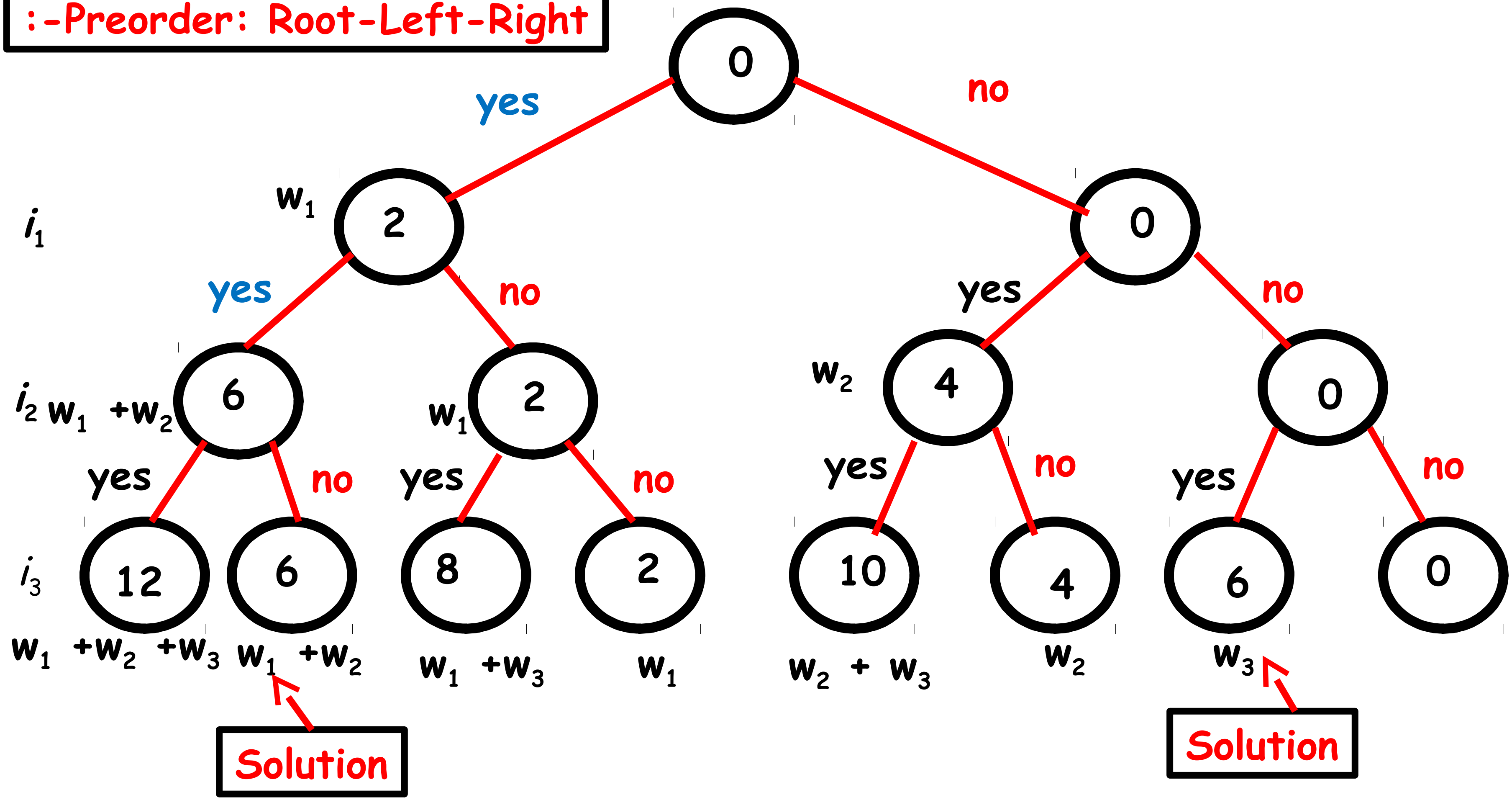
$\{\ \},\{2\},\{4\},\{6\},\{2,4\},\{2,6\},\{4,6\},\{2,4,6\}$

Solutions=$\{w_1,w_2\}$ and $\{w_3\}$= $\{2,4\}$ and $\{6\}$

Sum of subset Problem:

$w_1 = 2, \quad w_2 = 4, \quad w_3 = 6$ and $S = 6$

Nodes are ordered in DFS :-Preorder: Root-Left-Right

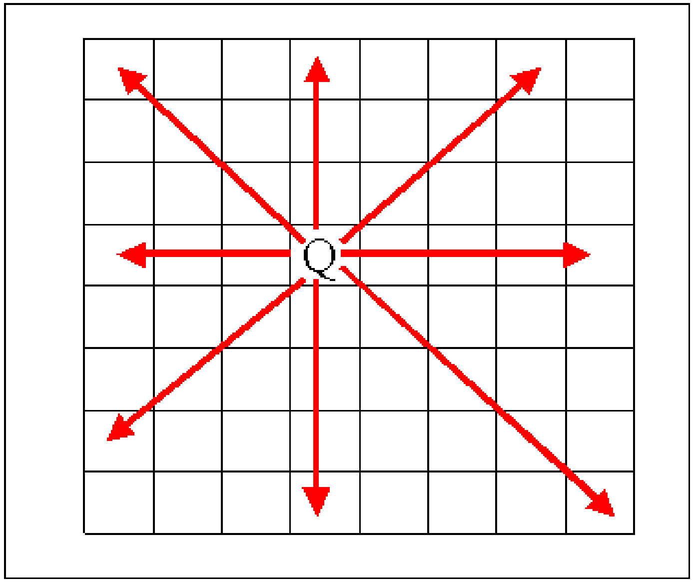


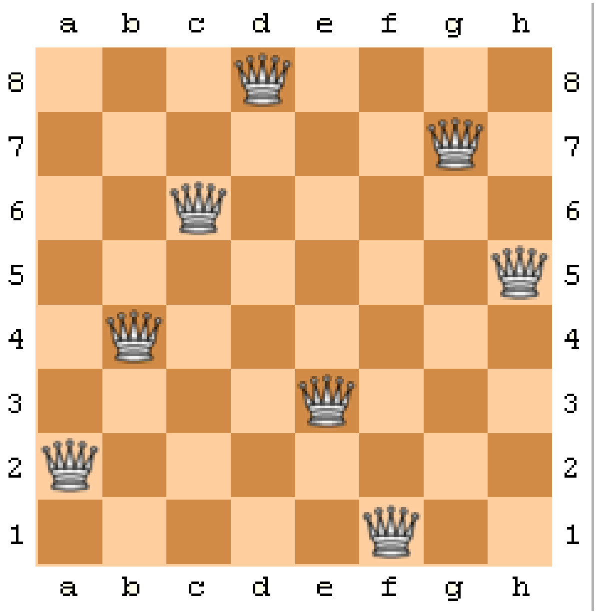
The sum of the included integers is stored at the node.

Backtracking : Example

n-queens Problem

Given n, place n-queens on the n x n board so that they are non-attacking.





8-Queens Problem one Solution

Backtracking : Example

n-queens Problem

The possible number of configurations are:

For 4-queens problem, there are 256 different configurations.

For 8-queens problem, there are 16,777,216 different configurations.

For 16-queens problem, there are 18,664, 744, 073, 709, 551, 616 configurations.

In general, for n-queens, we have n^n configurations.

For n=16, this would about 12,000 years on a fast machine.

Humans would find it hard to solve n-queen problems when n becomes more.

Backtracking : Example

n-queens Problem

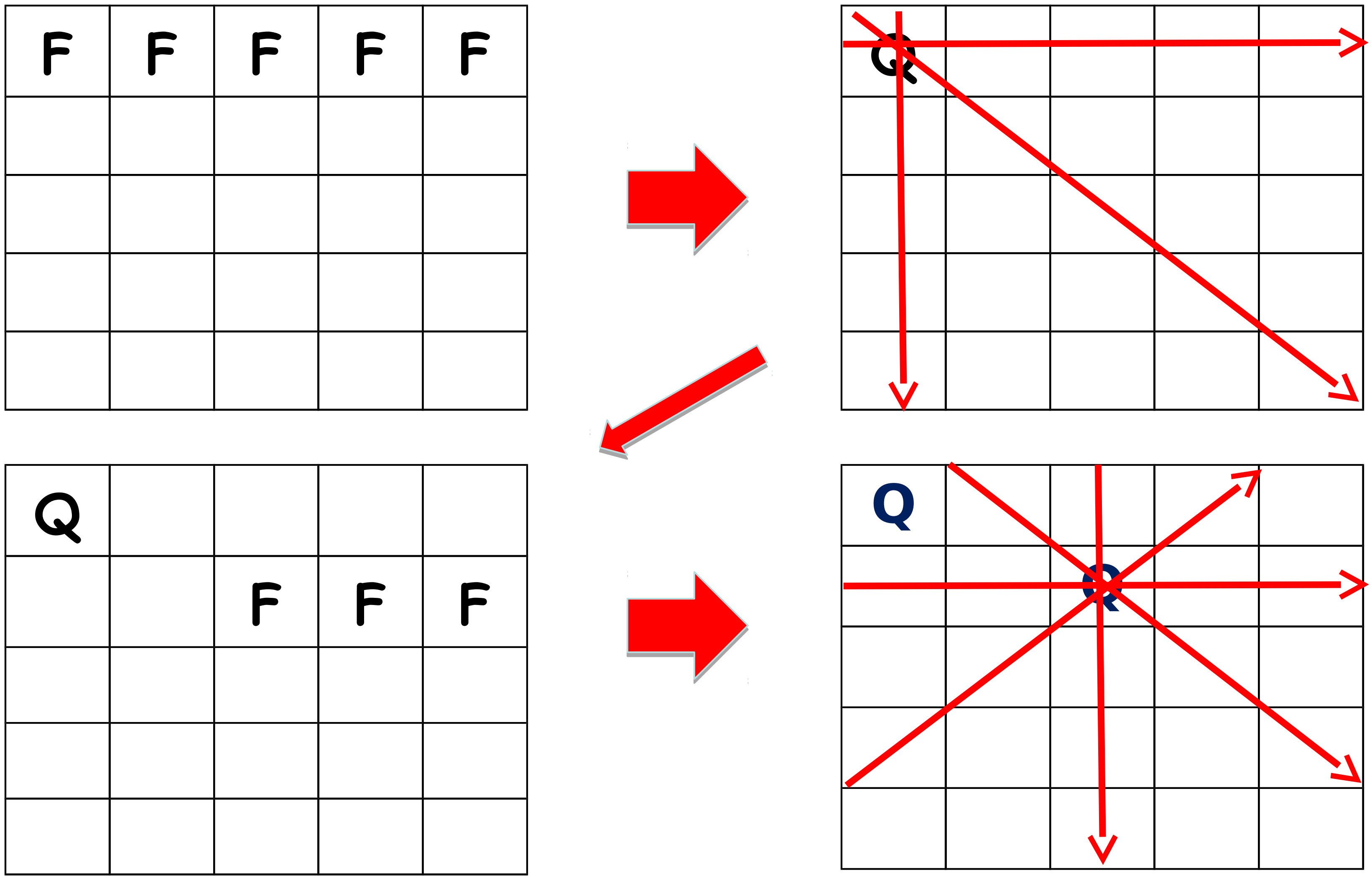
The Solution proceeds either by row or by column:

The backtracking strategy is as follows:

1) Place a queen on the first available square in row 1.
2) Move onto the next row, placing a queen on the first available square there (that doesn't conflict with the previously placed queens).
3) Continue in this fashion until either:
   a) you have solved the problem, or
   b) you get stuck.
      - When you get stuck, remove the queens that got you there, until you get to a row where there is another valid square to try.

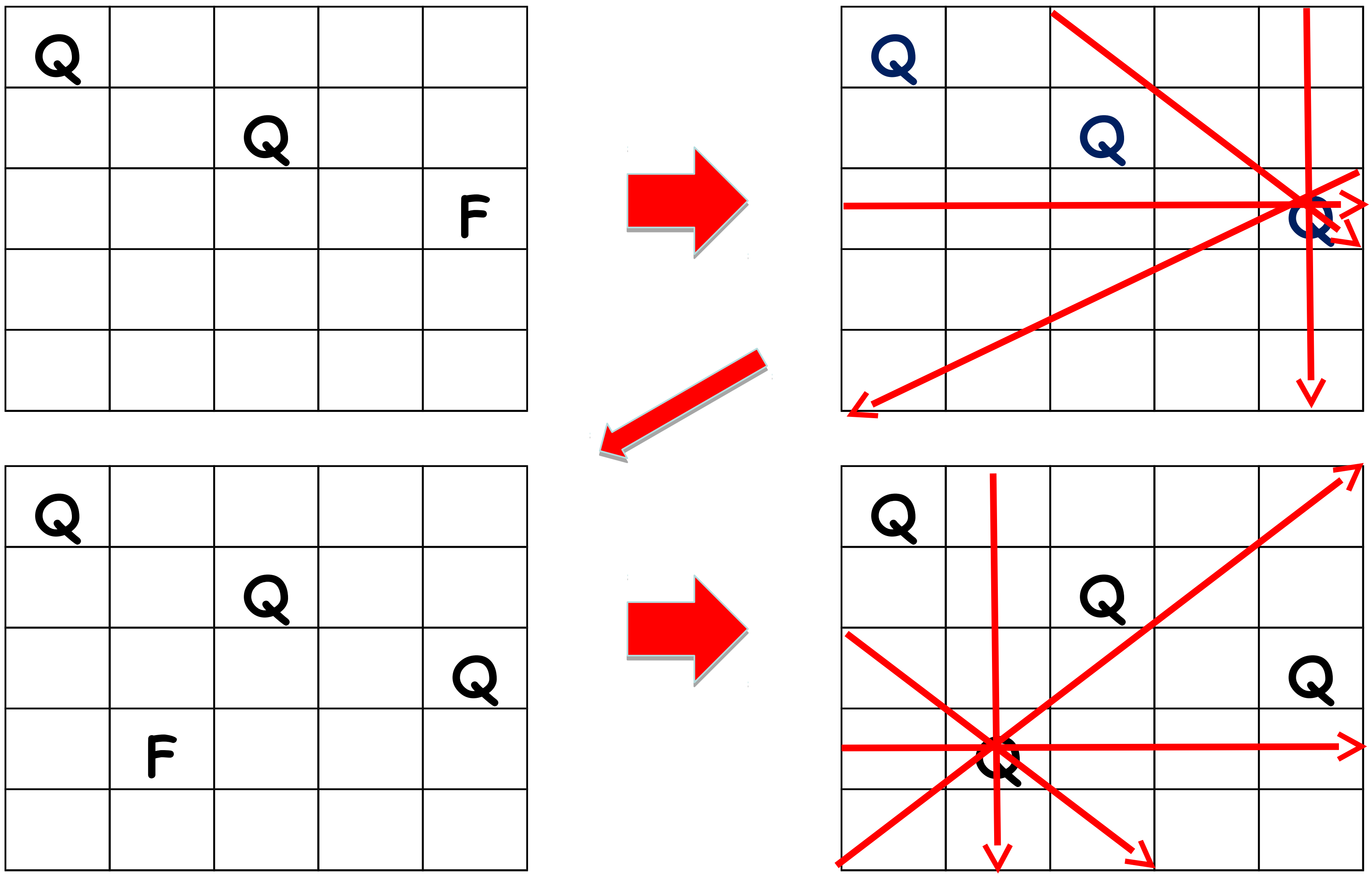
Backtracking : Example

5-queens Problem



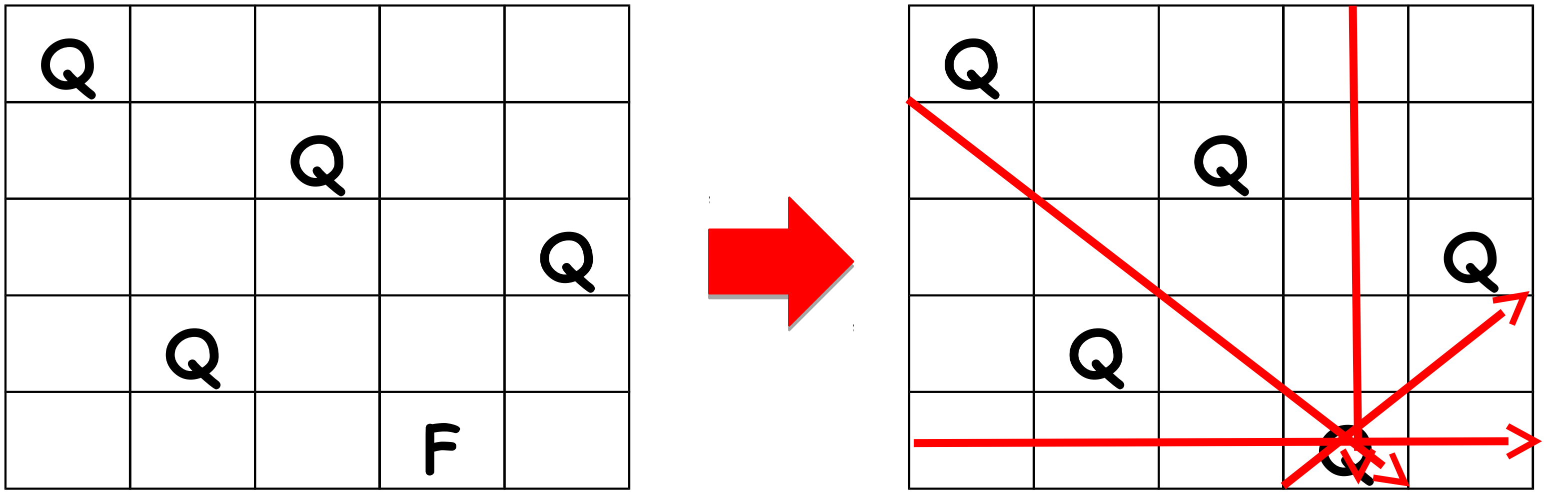
Backtracking : Example

5-queens Problem



Backtracking : Example

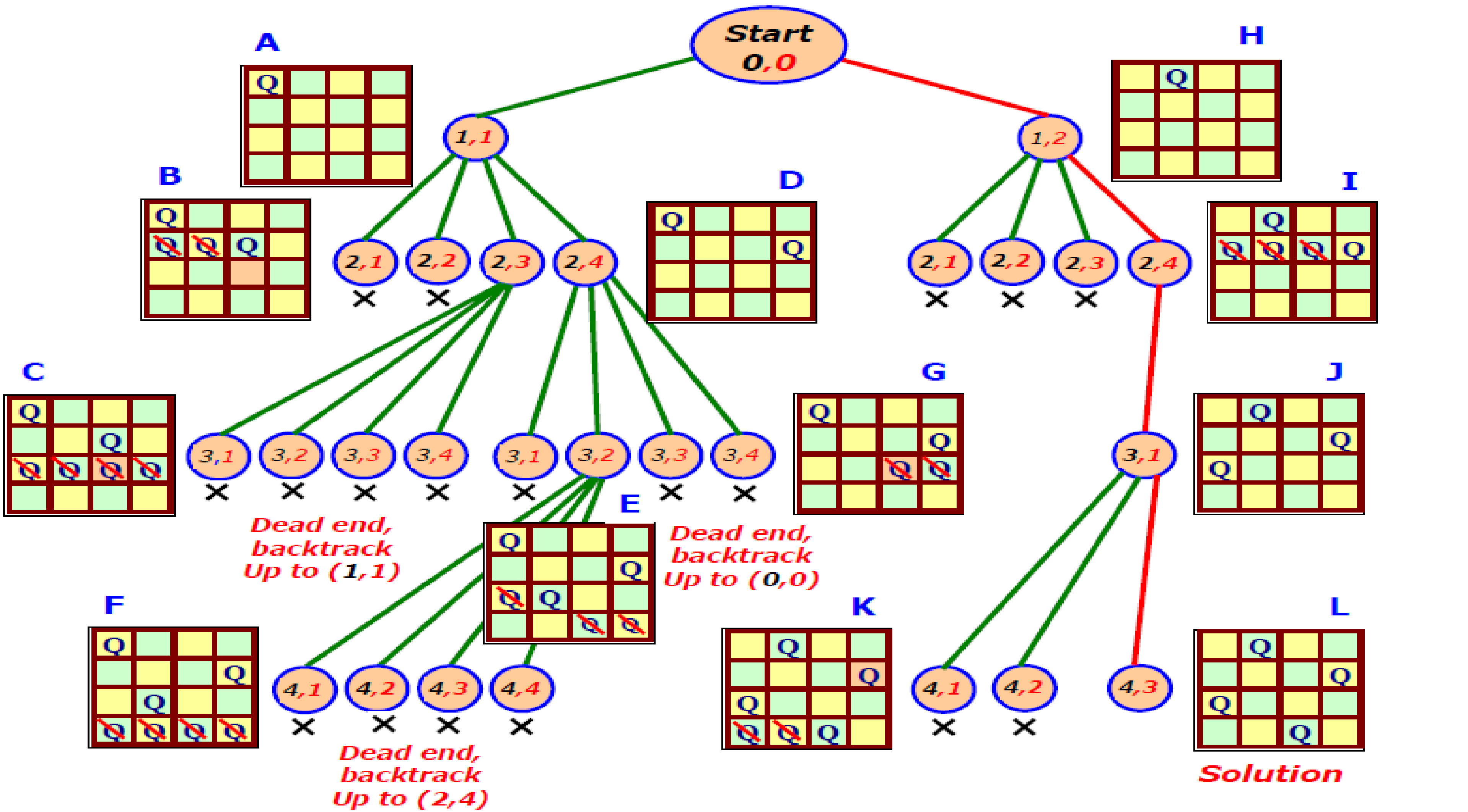
5-queens Problem



DONE

Backtracking : Example

4-queens Problem



Backtracking : Example

n-queens Problem

The backtracking strategy is as follows:

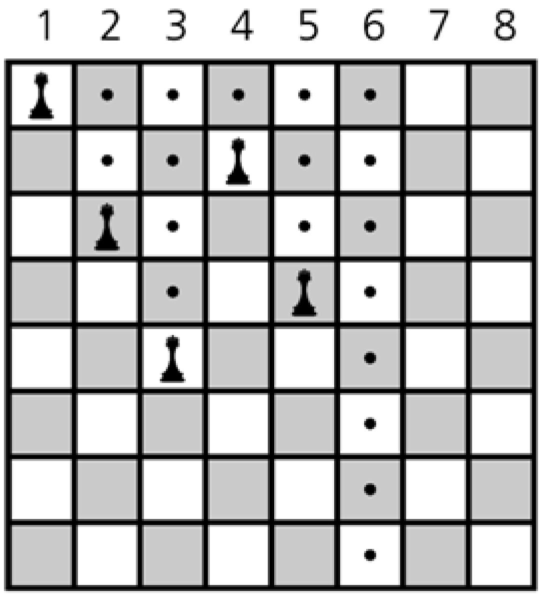
1) Place a queen on the first available square in column 1.

2) Move onto the next column, placing a queen on the first available square there (that doesn't conflict with the previously placed queens).

3) Continue in this fashion until either:
   a) you have solved the problem, or
   b) you get stuck.
      - When you get stuck, remove the queens that got you there, until you get to a column where there is another valid square to try.

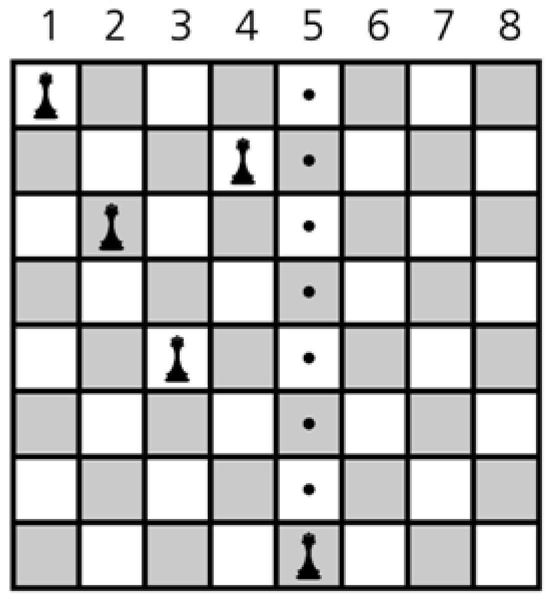
Backtracking : Example

8-queens Problem



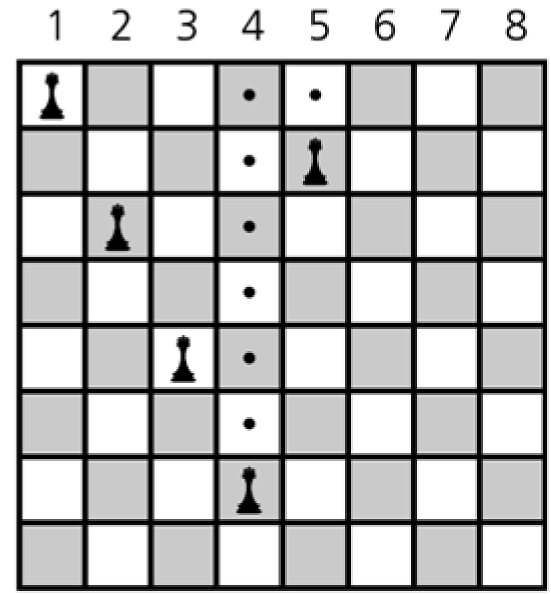
(a)

Five queens that cannot attack each other, but that can attack all of column 6.



(b)

Backtracking to column 5 to try another square for the queen

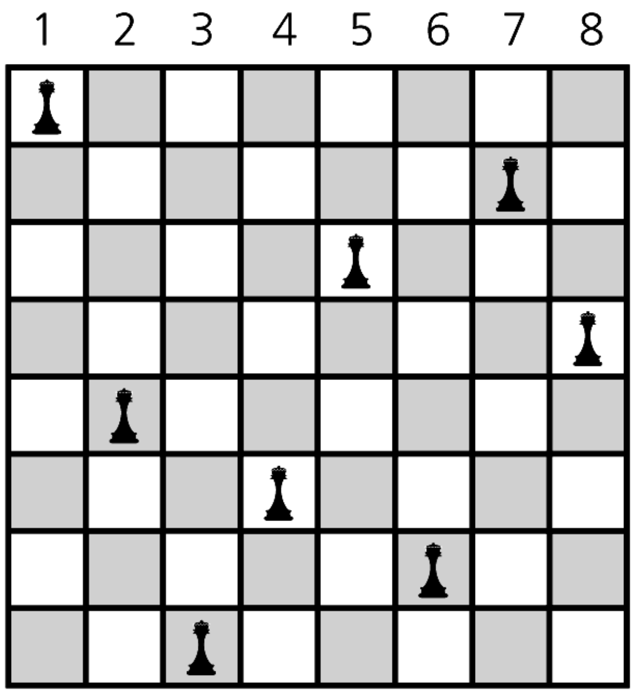


(c)

backtracking to column 4 to try another square for the queen and then considering column 5 again.

Backtracking : Example

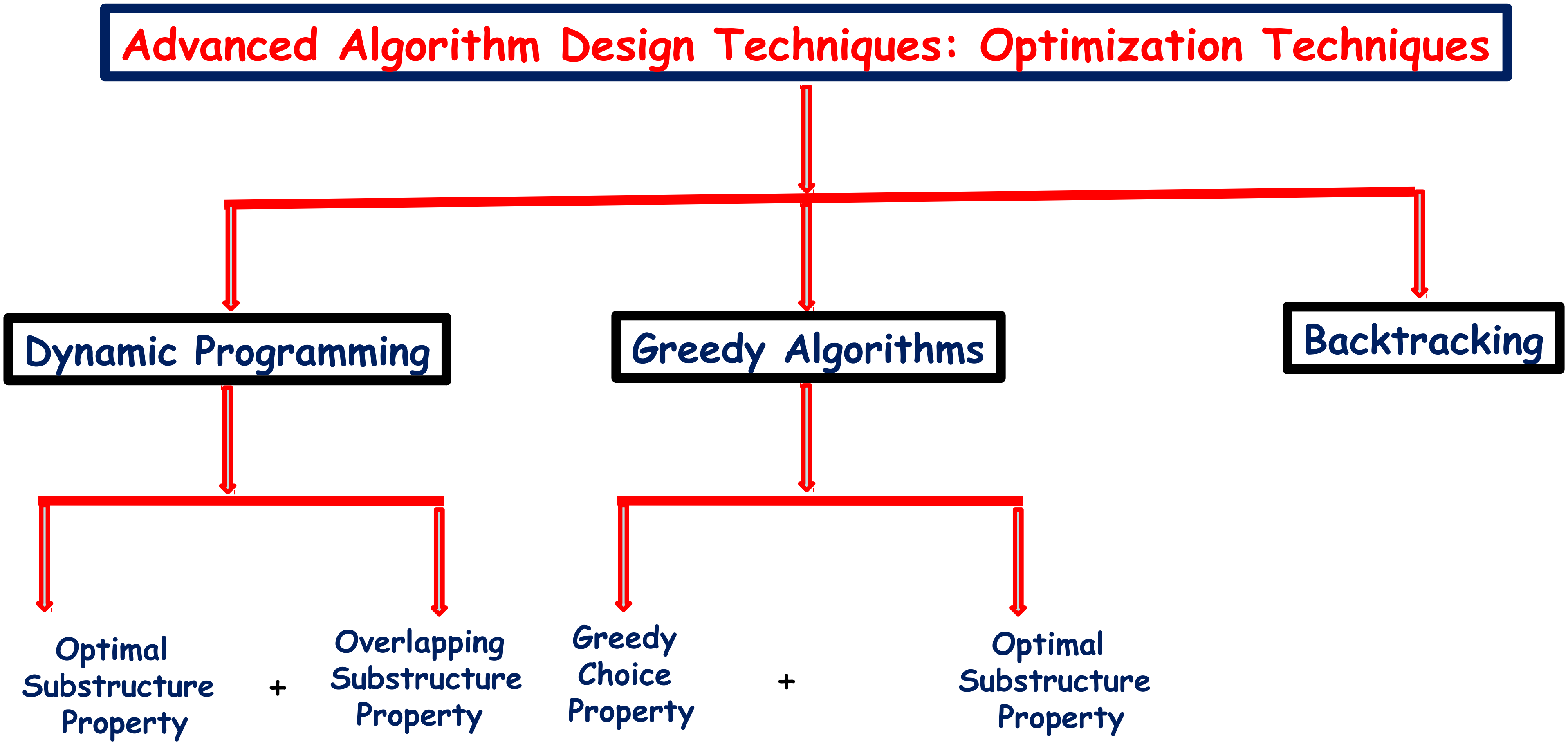
n-queens Problem



A solution to the Eight Queens problem

For 8-queens problem, there are 92 distinct solutions.

Learn DAA: From B K Sharma



Dynamic Programming

Programming means "Tabular Method", not computer programming.