GLA UNIVERSITY MATHURA

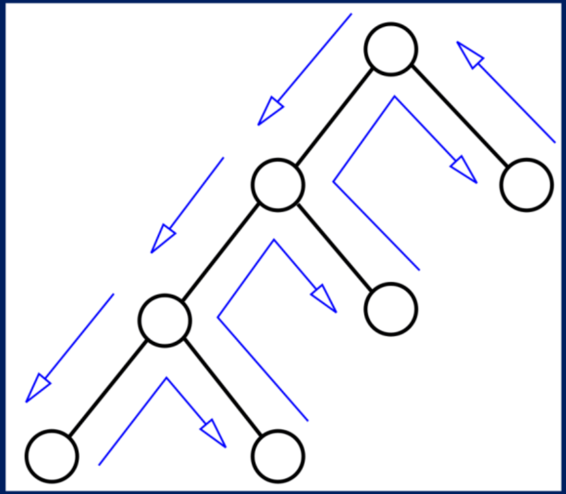
Recognised by UGC Under Section 2(f)

Accredited with A Grade by NAAC

DESIGN & ANALYSIS OF ALGORITHM (BCSC0012)

Chapter 13: Backtracking

8 – Queens Problem



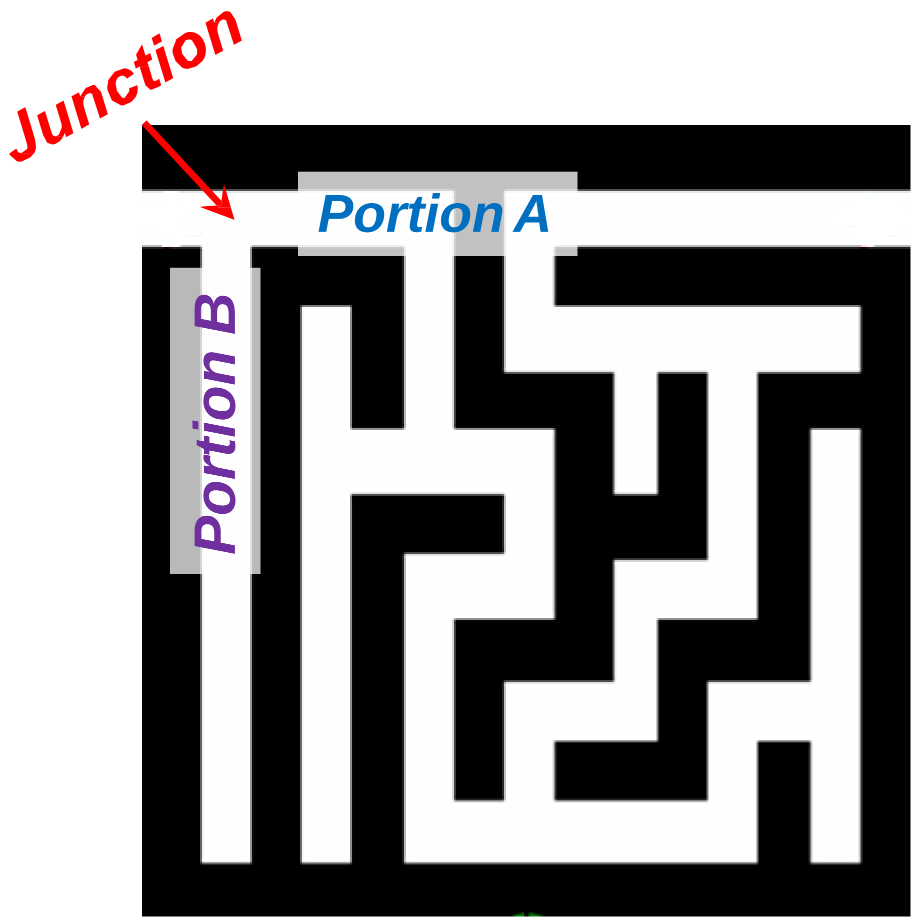
Prof. Anand Singh Jalal

Department of Computer Engineering & Applications

Backtracking

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

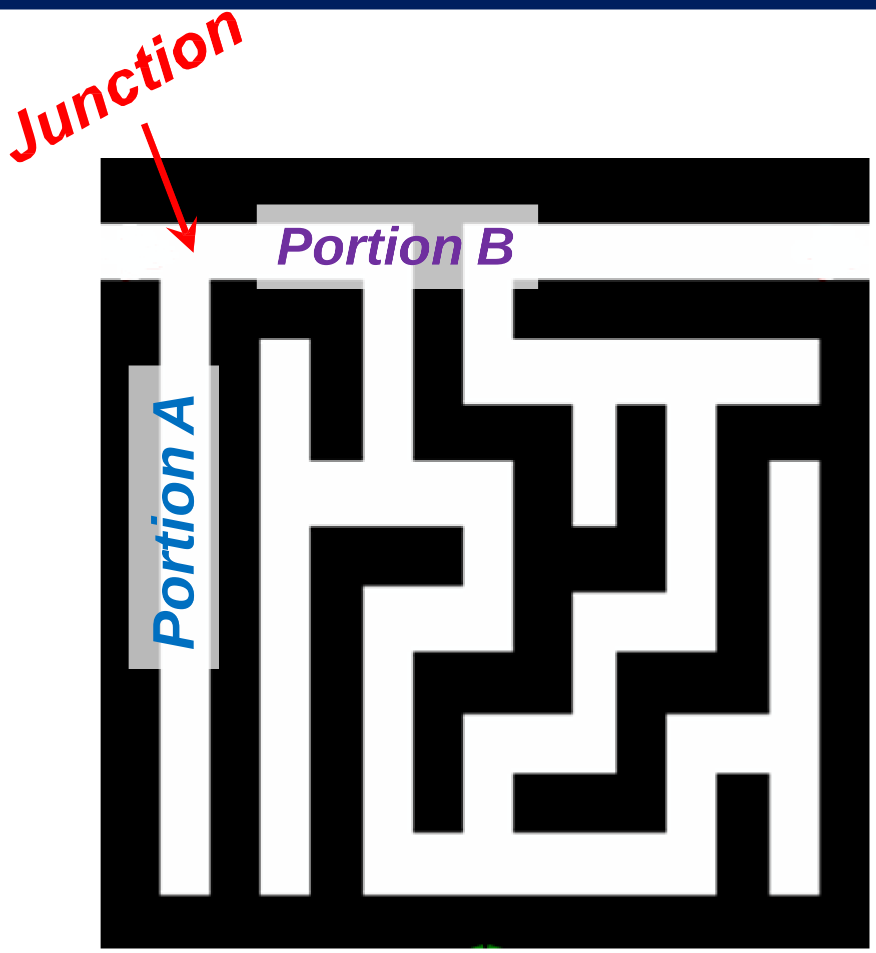
A standard example of backtracking would be going through a maze.
At some point in a maze, you might have two options of which direction to go:



Backtracking

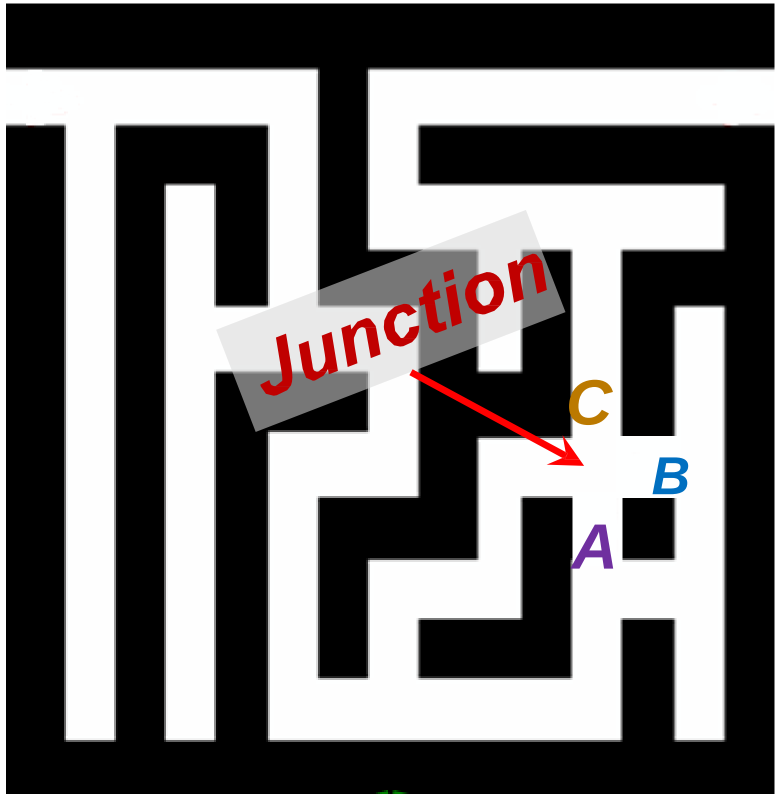
- One strategy would be to try going through **Portion A** of the maze.
  - If you get stuck before you find your way out, then you ***"backtrack"*** to the junction.

- At this point in time you know that **Portion A** will ***NOT*** lead you out of the maze,
  - so you then start searching in **Portion B**



Backtracking

- Clearly, at a single junction you could have even more than 2 choices.
- 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 maze.



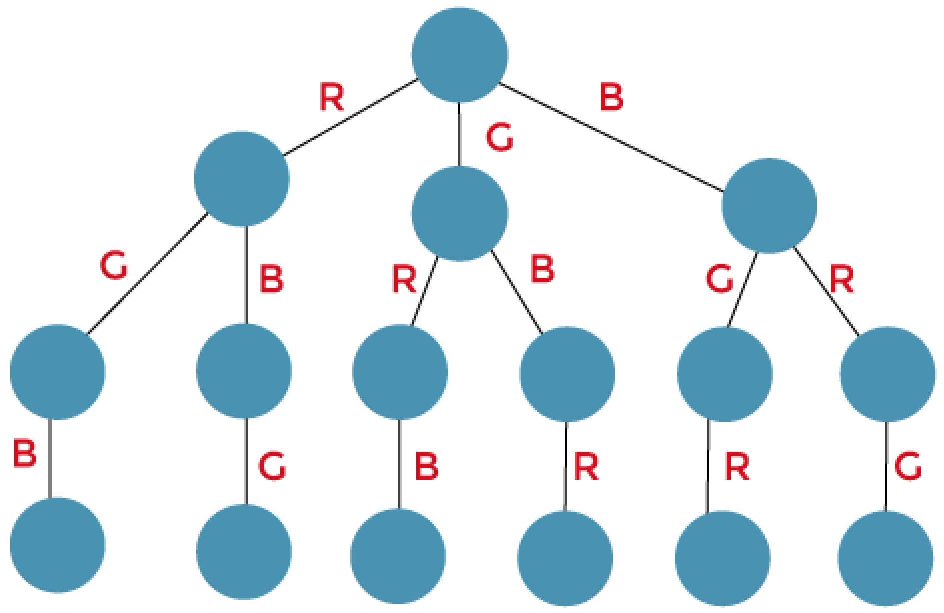
Backtracking can reduce a NP compete problem to linear problem by only going through selected branches of the global solution.

Backtracking ...

Backtracking is nothing but the modified process of the Brute force approach. It is a technique where multiple solutions to a problem are available, and it searches for the solution to the problem among all the available options.

Example:

Consider the box, and we have three objects of three different colors. One object is of red color, the second object is of green color, and the third object is of blue color. Now, we have to keep these objects inside the box, and we have multiple options.

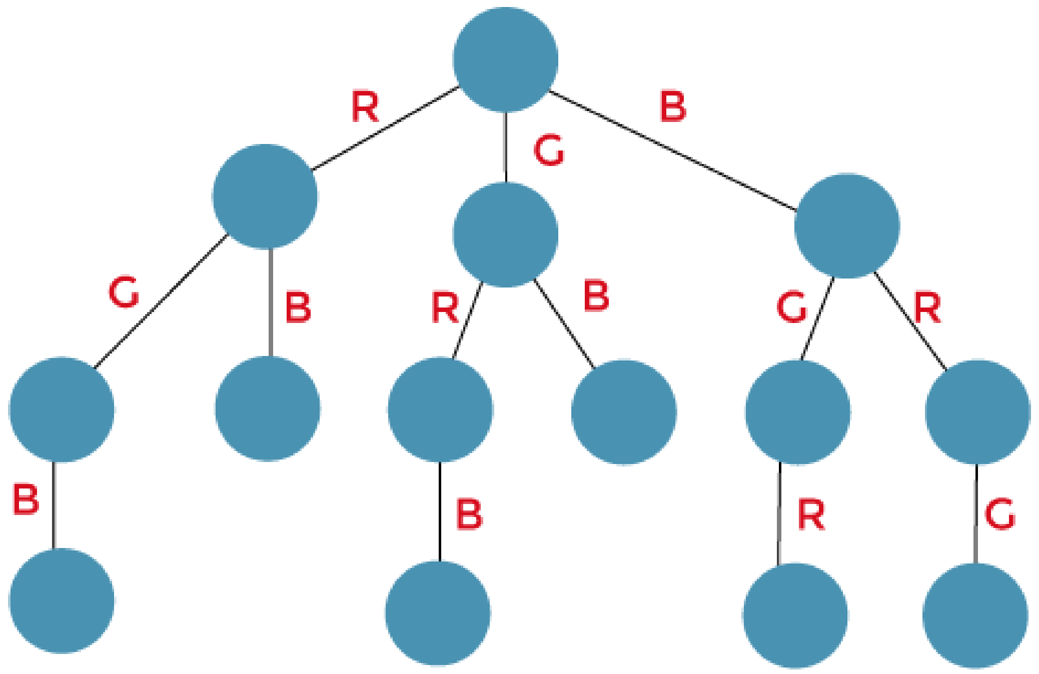


The Solution is represented by a state space tree

The above is the state space tree that shows all the possible solutions related to the problem.

Backtracking ...

Using backtracking, we can solve the problem that has some constraints and we can find the solution based on these constraints. Suppose in the above example; the constraint is blue color object must not be in the middle (bounding function).

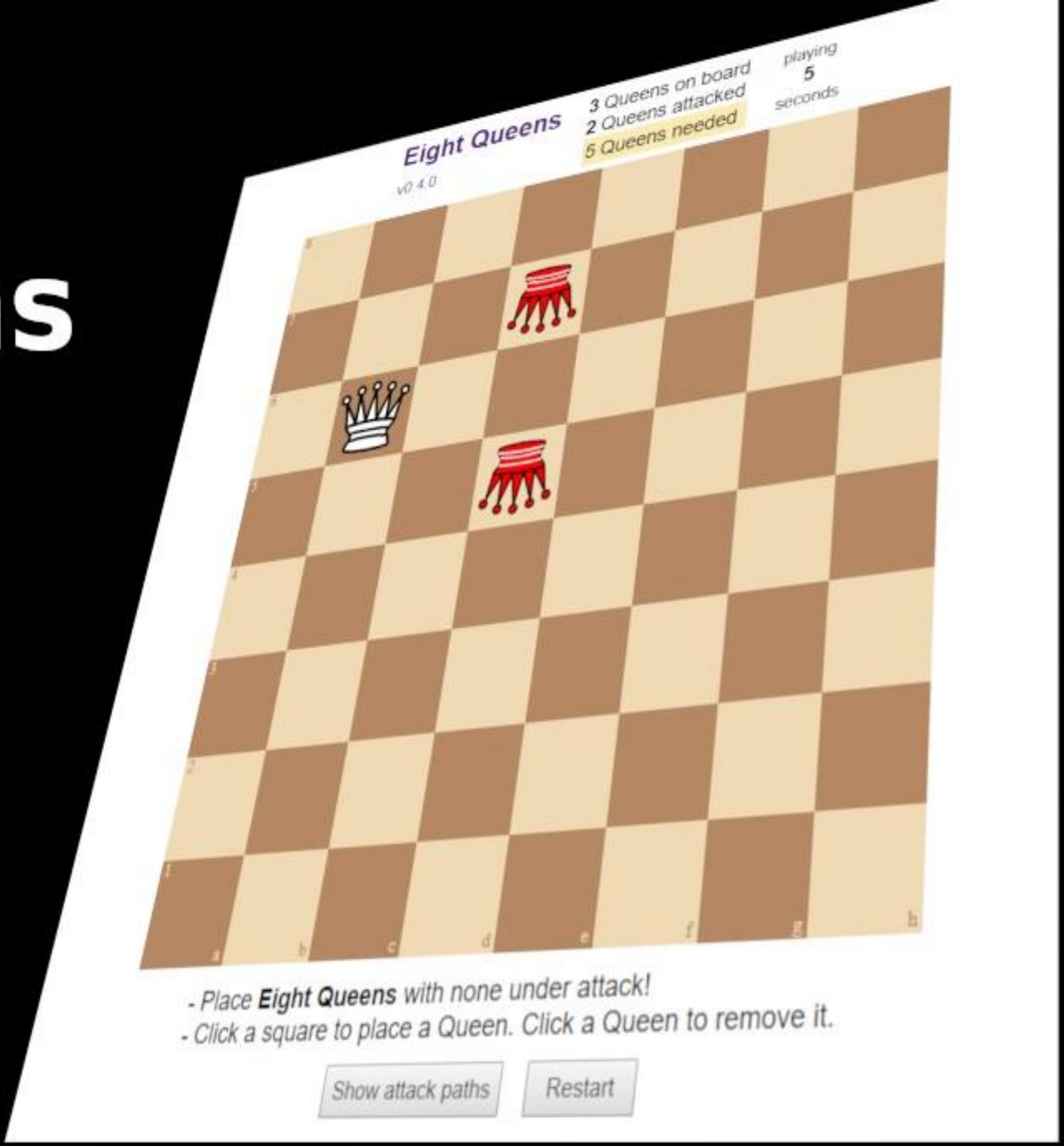


The above is the state space tree that does not have blue object in the middle of the solution.

Examples of backtracking method

- 8 queens problem
- Graph-coloring problem
- 4 knight problem
- Perfect hash function problems

Eight Queens



Eight Queens

v0.4.0

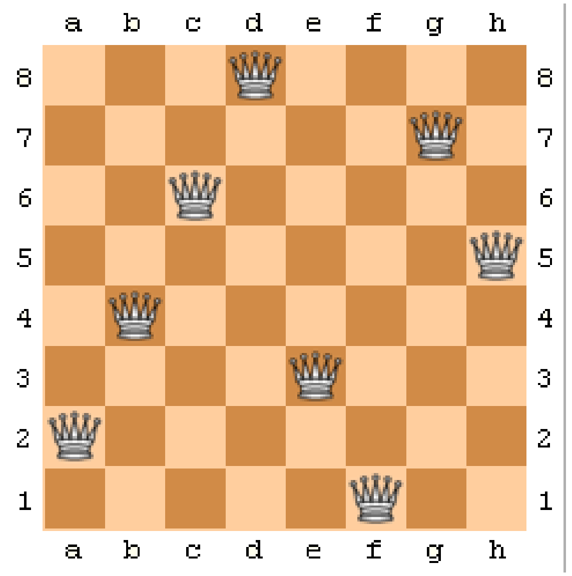
3 Queens on board
2 Queens attacked
5 Queens needed

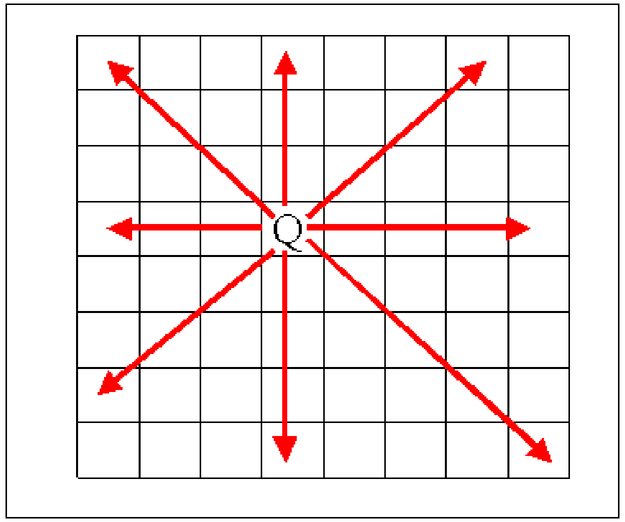
playing
5
seconds

- Place **Eight Queens** with none under attack!
- Click a square to place a Queen. Click a Queen to remove it.

The 8 Queens Problem

- A classic chess puzzle
  - Place 8 queen pieces on a chess board so that none of them can attack one another
  - In chess, queens can move all the way down any row, column or diagonal (so long as no pieces are in the way).
  - Due to the first two restrictions, it's clear that each row and column of the board will have exactly one queen.

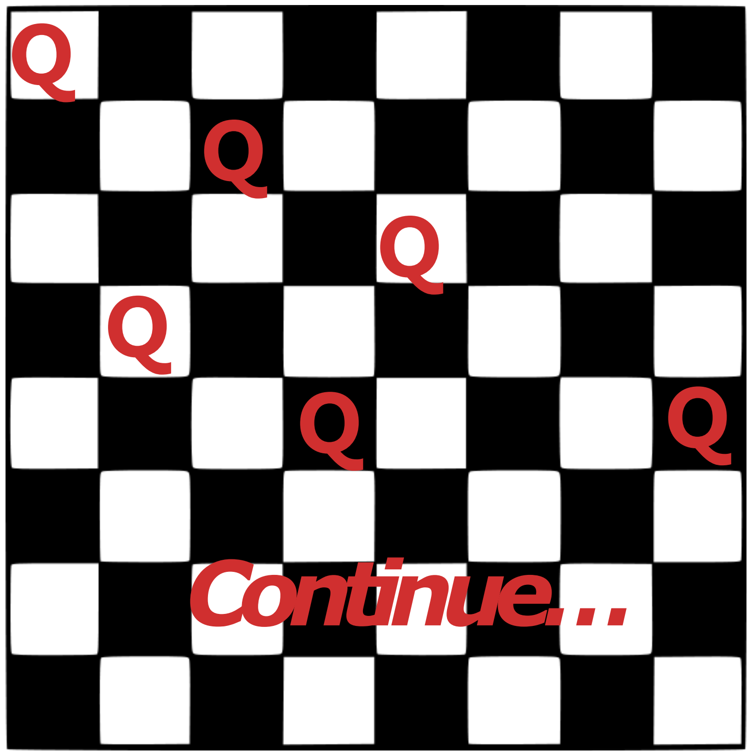




GLA UNIVERSITY MATHURA
Accredited with A Grade by NAAC

The 8 Queens Problem

- 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.



Animated Example: http://www.hbmeyer.de/backtrack/achtdamen/eight.htm#up

The N Queens Problem …

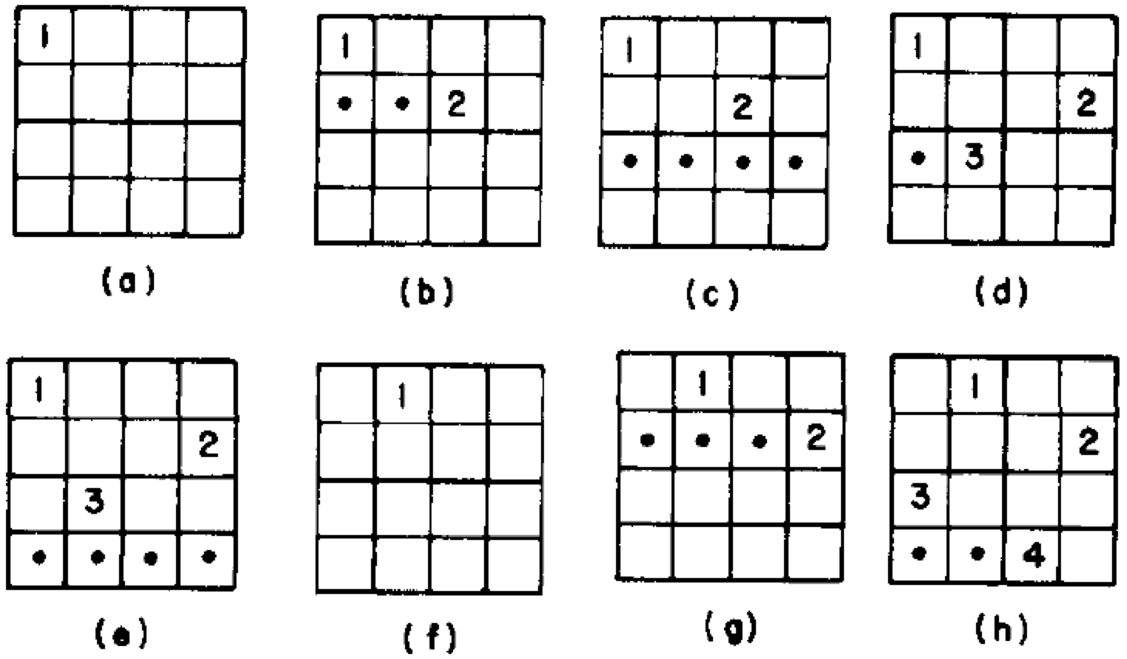
- The n-queens problem is a generalization of the 8-queens problem of n queens are to be placed on a n x n chessboard so that **no two attack, i.e., no two queens are on the same row, column or diagonal.**

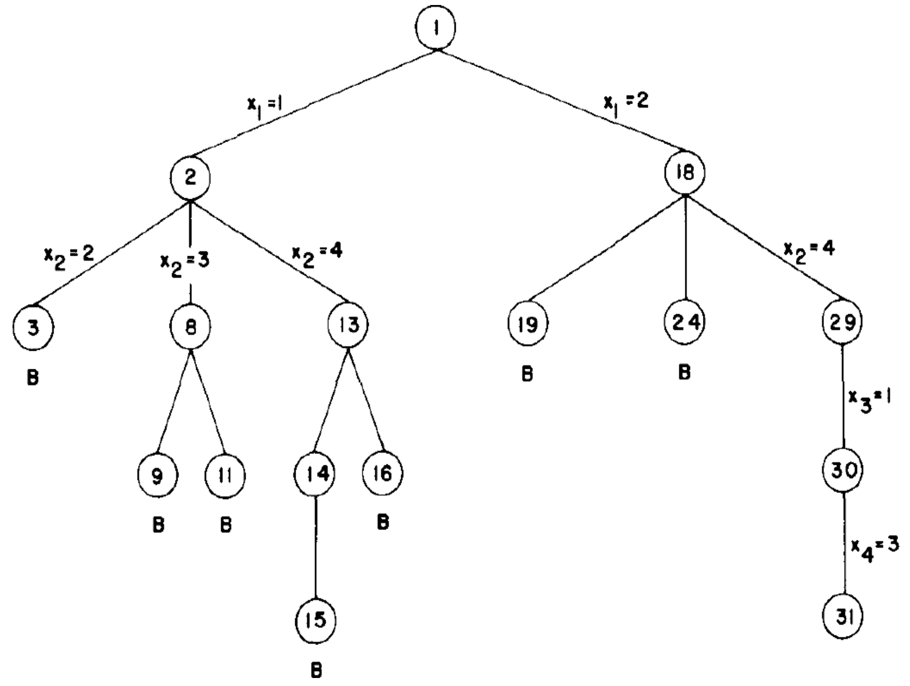
```
Nqueens(k,n){
for i=1 to n{
      if(place(k,i)){
          x[k]=i
          if (k==n){
              for j=1 to n
                 print x[i]}
          else
             Nqueens(k+1,n)}}}
```

```
place(k,i){
for j=1 to k
      if(x[j]==i || abs(x[j]-i ==abs(j-k))
             return false
      return true
             }
```

The 4 Queens Problem …

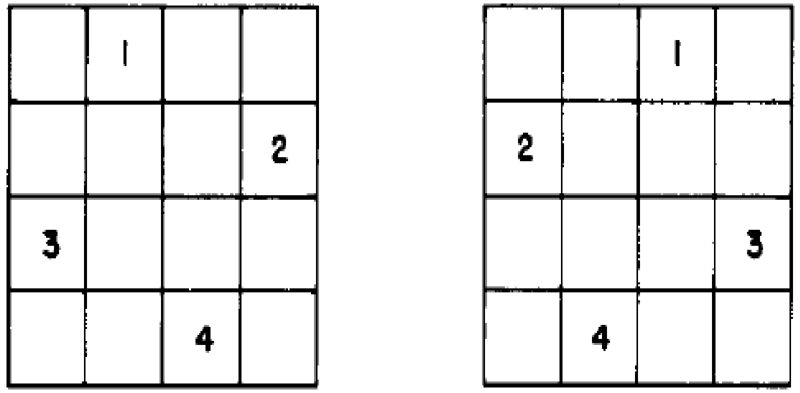
- The 4-queens problem is a generalization of 4 queens are to be placed on a 4 x 4 chessboard so that no two attack, i.e., no two queens are on the same row, column or diagonal.





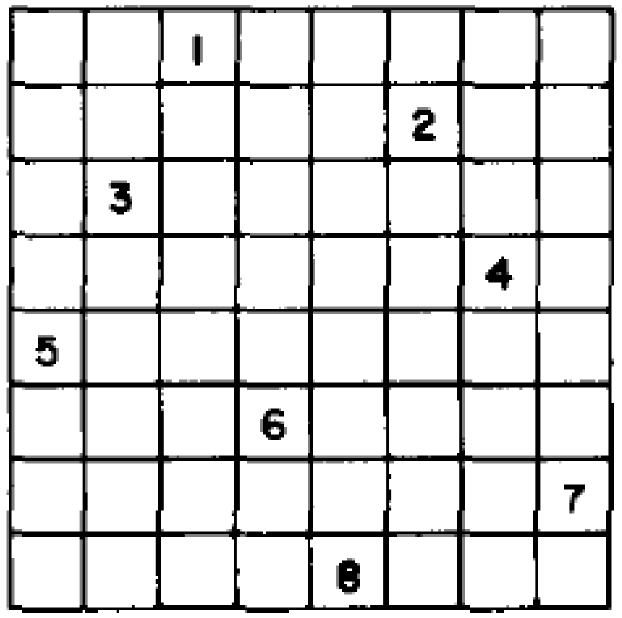
The 4 Queens Problem ...

- The 4-queens problem is a generalization of 4 queens are to be placed on a 4 x 4 chessboard so that no two attack, i.e., no two queens are on the same row, column or diagonal.

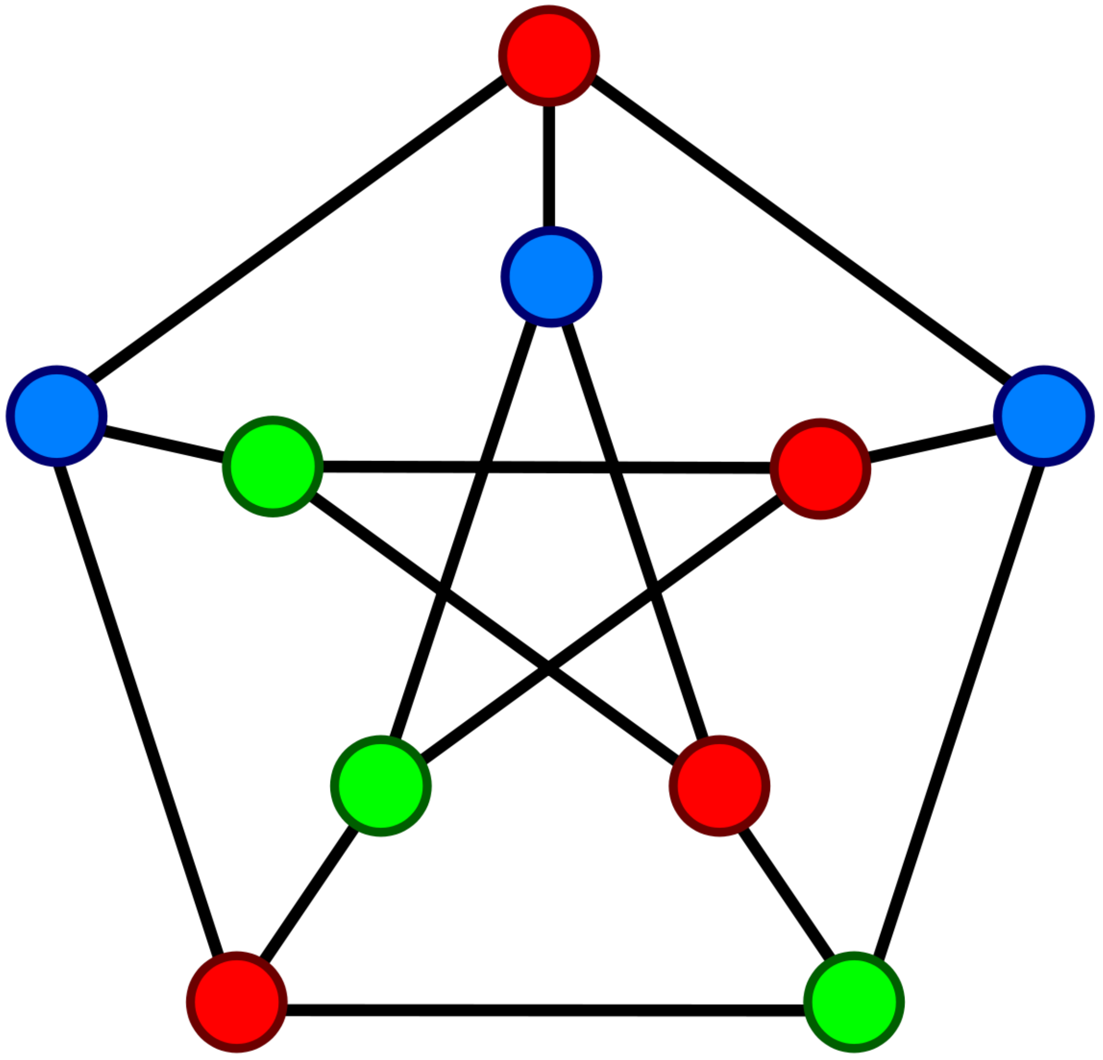


The 8 Queens Problem …

- The 8-queens problem is a generalization of 8 queens are to be placed on a 8 x 8 chessboard so that no two attack, i.e., no two queens are on the same row, column or diagonal.



Graph Coloring Problem

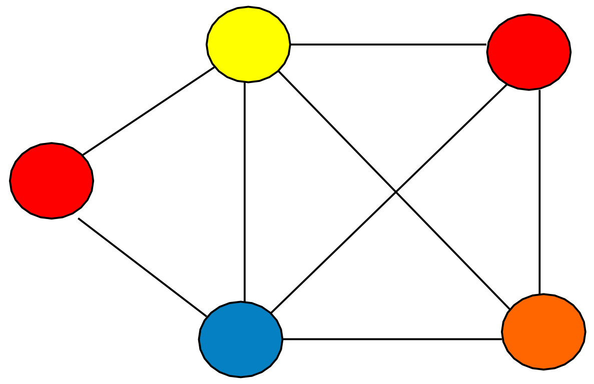


Graph Coloring Problem

- Graph Coloring is an assignment of colors (or any distinct marks) to the vertices of a graph. Strictly speaking, a coloring is a proper coloring if no two adjacent vertices have the same color.

- Direct approach builds the tree of ALL possibilities in exponential time.

$$f : V(G) \rightarrow S$$



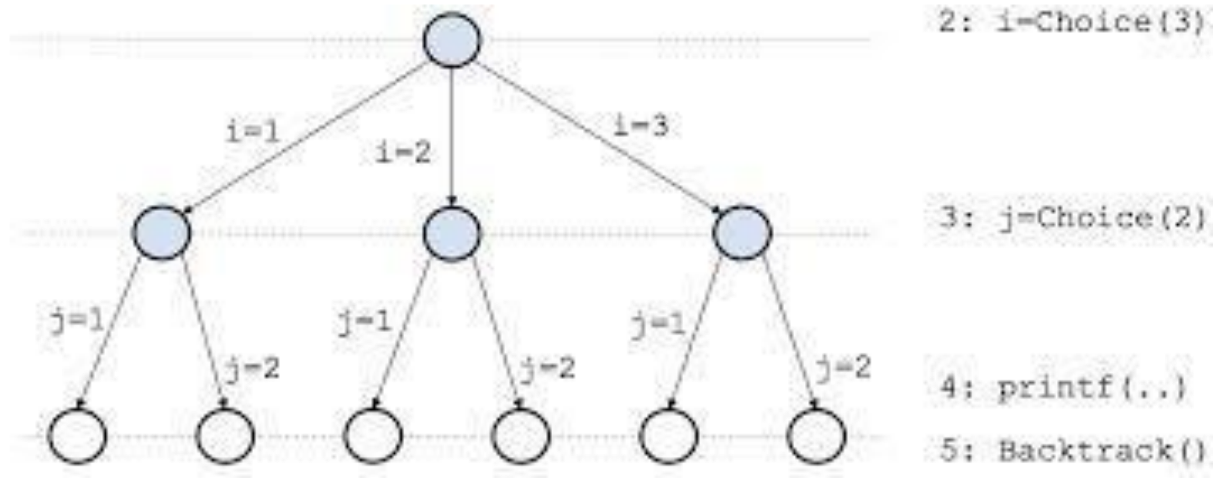
Graph Coloring Problem

- Explore possibilities; backtrack when doesn’t work.
- Start by assigning an arbitrary color to one of the vertices.
- Continue coloring while maintaining the constraints imposed by the edges
- If reach a vertex that can’t be colored, backtrack -- go back up the recursion tree and explore other children

Graph Coloring Problem

Partial 3-coloring (3 colors) is solved by the following method:

- Color first vertex with 1st color, color next vertex with the next color, check if those two vertices are adjacent, if not - coloring is legal, proceed to next vertex, if yes and color is the same – coloring is illegal, try next color for second vertex. If all colors tried and all colorings are illegal, backtrack, try next color for previous vertex etc.
- Note: sometimes solution is impossible.
- Exponential O(3^n) complexity is reduced to O(n) on average.



Any Questions ?

Dr. Anand Singh Jalal
Professor
Email: asjalal@gla.ac.in