Assignment No: 4 (Constraint Satisfaction Problem)

Objective: In this experiment, we will be able to do the following:

- How to place N queens on board with non-attacking mode using backtracking.
- CSP for Graph Coloring Problem

Prerequisite: Basic knowledge of CSP, Backtracking

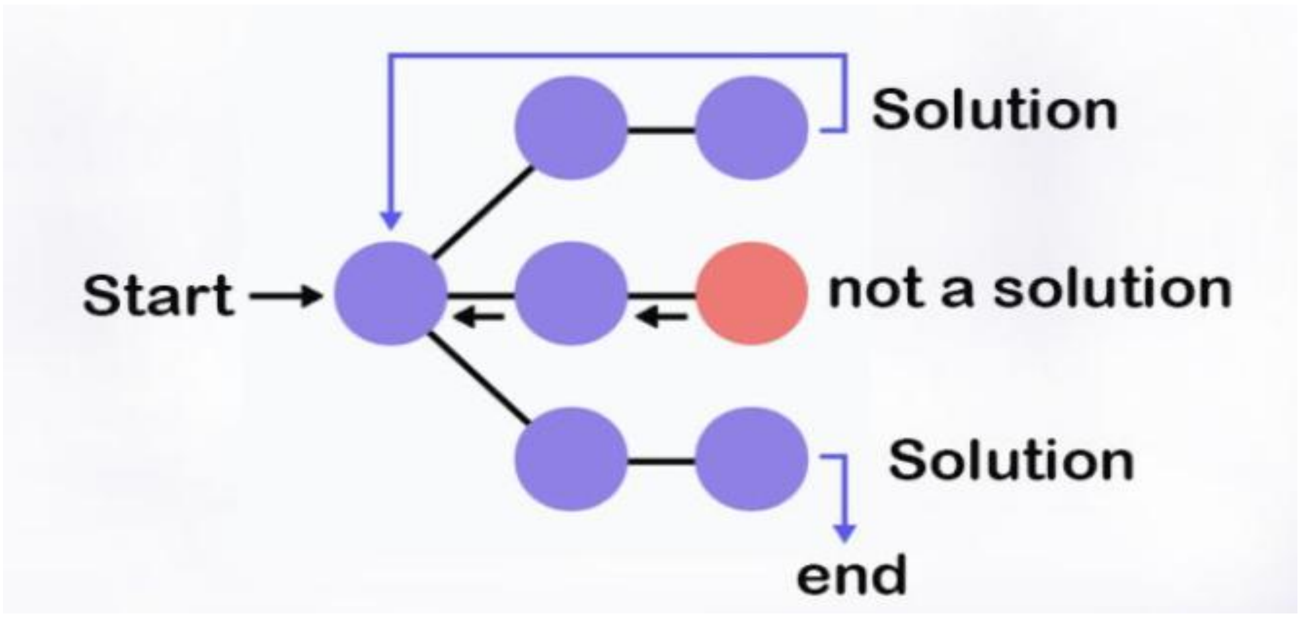
Outcome: Successfully able to place N queens on board mode using backtracking.

Theory: CSP means solving a problem under certain constraints or rules.
CSP depends on three components

X: It is a set of variables
D: It is a set of domains
There is a specific domain for each variable.
Domain is possible values each variable can take.
C: It is a set of constraints which are followed by the set of variables. Rules that restrict how variables can be assigned.

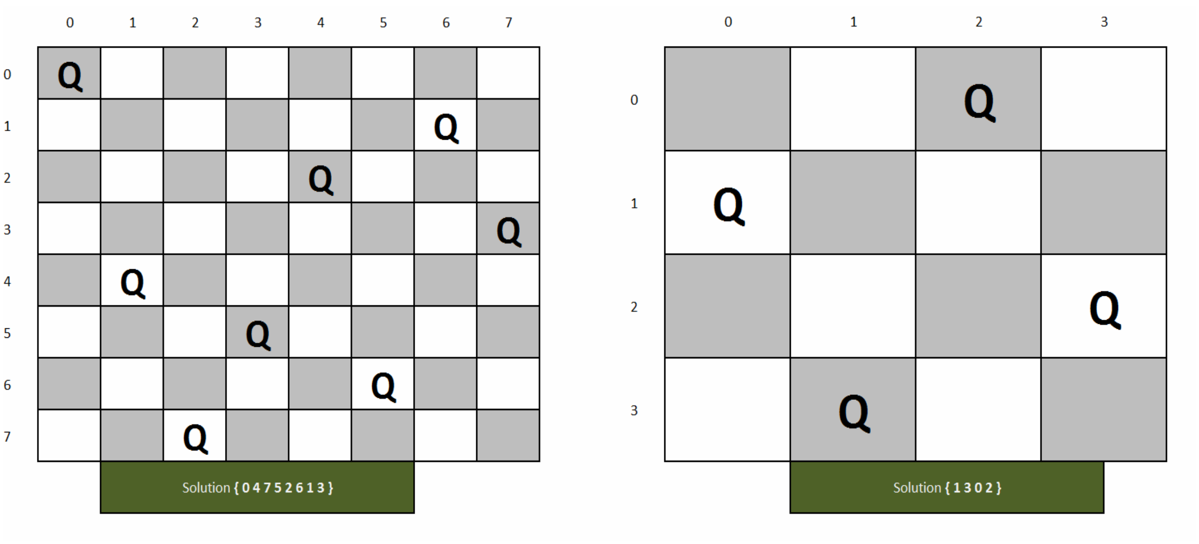
Backtracking Algorithm

Backtracking is an algorithmic-technique for solving problems recursively by trying to build a solution incrementally, one piece at a time, removing those solutions that fail to satisfy the constraints of the problem at any point of time (by time, here, is referred to the time elapsed till reaching any level of the search tree).



N-Queens problem:

The n-Queens problem is a puzzle of placing exactly N queens on an NxN chessboard, such that no two queens can attack each other in that configuration. Thus, no two queens can lie in the same row, column or diagonal. The idea is to place queens one by one in different columns, starting from the leftmost column. When we place a queen in a column, we check for clashes with already placed queens. In the current column, if we find a row for which there is no clash, we mark this rowand column as part of the solution. If we do not find such a row due to clashes then we backtrack and return false.



N-Queens problem Algorithm

1) Start in the leftmost column
2) If all queens are placed return true
3) Try all rows in the current column. Do the following for every tried row.
   a) If the queen can be placed safely in this row then mark this [row, column] as part of the solution and recursively check if placing the queen here leads to a solution.
   b) If placing the queen in [row, column] leads to a solution then return true.
   c) If placing queen doesn't lead to a solution then unmark this [row, column] (Backtrack) and go to step (a) to try other rows.
4) If all rows have been tried and nothing worked, return false to trigger backtracking.

Graph-Coloring Problem

Graph Coloring is a classic Constraint Satisfaction Problem (CSP) where the goal is to assign colors to vertices of a graph such that:

1. No two adjacent vertices share the same color (edge constraint).
2. Minimize the number of colors used (optional optimization).

Formulating Graph Coloring as a CSP

1. Variables (V): Each vertex of the graph is a variable.
2. Domain (D): A set of available colors (e.g., {Red, Green, Blue}).
3. Constraints (C): If two vertices are connected by an edge, they must have different colors.

Example: Coloring a Graph

Consider the following graph:

```
A------B
|     /  |
|   /    |
C-------D
```

Step 1: Define the CSP Components

1. Variables (V): A, B, C, D
2. Domain (D): {Red, Green, Blue}
3. Constraints (C):
    - A ≠ B
    - A ≠ C
    - B ≠ C
    - B ≠ D
    - C ≠ D

Step 2: Solve the Graph Coloring CSP

We will use the Backtracking algorithm to assign colors:

1. Assign A = Red.
2. For B, valid options: {Green, Blue}. Choose B = Green.
3. For C, valid options: {Green, Blue}. Choose C = Blue.
4. For D, valid options: {Red, Blue} (since B = Green). Choose D = Red.

Solution:

A = Red, B = Green, C = Blue, D = Red.

Step 3: Techniques for Solving Graph Colouring CSP

1. Backtracking: Explore all possibilities and backtrack if a conflict occurs.
2. Forward Checking: After assigning a colour, eliminate invalid options for adjacent vertices.
3. Minimum Remaining Values (MRV): Choose the vertex with the fewest legal colour option

Branch and Bound

Branch and bound is an algorithm design paradigm which is generally used for solving combinatorial optimization problems. These problems are typically exponential in terms of time complexity and may require exploring all possible permutations in the worst case. The Branch and Bound Algorithm technique solves these problems relatively quickly.

Parameter	**Backtracking**	**Branch and Bound**
Approach	Backtracking is used to find all possible solutions available to a problem. When it realizes that it has made a bad choice, it undoes the last choice by backing it up. It searches the state space tree until it has found a solution for the problem.	Branch-and-Bound is used to solve optimization problems. When it realizes that it already has a better optimal solution that the pre-solution leads to, it abandons that pre-solution. It completely searches the state space tree to get an optimal solution.
Traversal	Backtracking traverses the state space tree by DFS (Depth First Search) manner.	Branch-and-Bound traverse the tree in any manner, DFS or BFS.
Function	Backtracking involves a feasibility function.	Branch-and-Bound involves a bounding function.
Problems	Backtracking is used for solving Decision Problems.	Branch-and-Bound is used for solving the optimization Problem.
Searching	In backtracking, the state space tree is searched until the solution is obtained.	In Branch-and-Bound as the optimum solution may be present anywhere in the state space tree, so the tree needs to be searched completely.
Efficiency	Backtracking is more efficient.	Branch-and-Bound is less efficient.
Applications	Useful in solving N-Queen Problem, Sum of subset.	Useful in solving Knapsack Problem, Traveling Salesman Problem.

Conclusion: In this way we have studied how to solve the CSP problem and how to place N queens on board with non-attacking mode using backtracking. Also studied how to solve Graph Coloring problem.