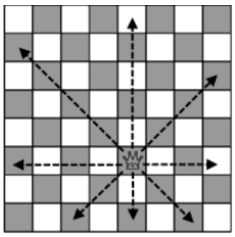
N-Queen

June 3, 2023

• Introduction

The N-Queens problem is a classic puzzle in computer science and mathematics. It involves placing N queens on an N X N chessboard in such a way that no two queens threaten each other. This can be achieved by positioning the queens such that two queens do not share the same row, column and diagonal.

The problem's name originates from the chess piece "Queen," which can move horizontally, vertically, and diagonally across the board. The challenge lies in finding a solution for any given N, determining all possible arrangements of queens that satisfy the problem's constraints.



Solving the N-Queens problem can be approached using various techniques, such as backtracking, constraint satisfaction.Researchers have devised numerous algorithms and strategies to tackle the problem efficiently and find solutions for larger values of N.

Indeed, the N-Queens problem is a captivating and challenging combinatorial problem that has intrigued researchers and enthusiasts in the field of problem-solving.

The N-Queens problem serves as a benchmark for evaluating the efficiency and effectiveness of different algorithms and heuristics. Researchers have explored various techniques, such as backtracking, constraint satisfaction, genetic algorithms, and simulated annealing,

to tackle this problem.

• Methodology

Backtracking is finding the solution of a problem whereby the solution depends on the previous steps taken. The basic idea behind backtracking is to explore all possible options systematically, while keeping track of the choices made so far. If the current path violates some constraint, the algorithm backtracks to the previous point and tries a different option.

Backtracking uses recursion to solve it's problems. It does so by exploring all the possibilities of any problem, unless it finds the best and feasible solution to it. Recursion occurs when a function calls itself repeatedly to split a problem into smaller sub-problems, until it reaches the base case.

The problem's combinatorial nature and exponential search space make it an ideal candidate for applying backtracking. The backtracking algorithm systematically explores the solution space, intelligently pruning branches that lead to invalid configurations. By employing backtracking, it becomes possible to efficiently find all valid solutions to the N-Queens problem.

The backtracking algorithm for the N-Queens problem works as follows:

➢ Algorithm:

1. Start with N X N empty chessboard.
2. Start from the first row (0th row) and iterate through each column (0 to N-1).
3. Check if queen attacks on any previously placed queen.
4. If the place is not safe then check for the next column.
5. If place is safe then repeat the step for next row.
6. If all queens are placed successfully then our solution is ready. print the chessboard .

7. If there are no more columns to try the current row ,backtrack to the previous row.Remove the queen from the current cell.Move back to previous row and try the next column. Repeat the steps until all possible arrangements explored.

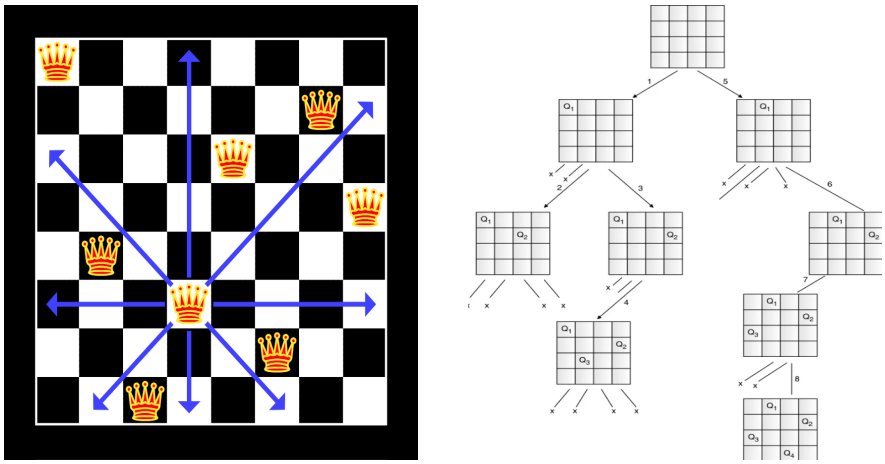


Figure 1: Backtracking Sequence

Backtracking allows the algorithm to explore different possibilities by undoing incorrect placements and trying alternative options.

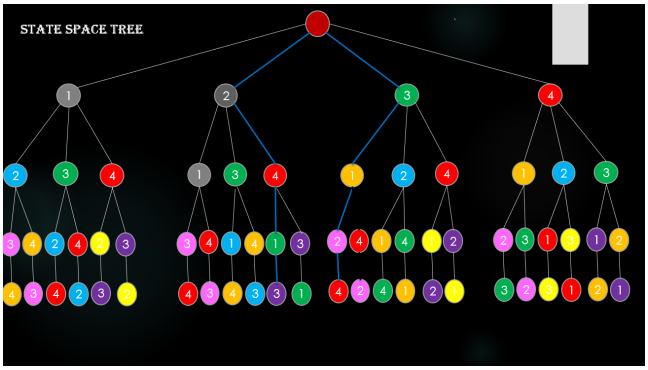


Figure 2: State Space Tree

- **Code**

```
#include <iostream>
using namespace std;

bool isSafe(char** board,int n,int row,int col);
void print(char** board,int n);
void nqueen(char** board,int n,int row);

bool isSafe(char** board,int n,int row,int col){
    //vertical up
    for(int i=row-1;i>=0;i--){
        if(board[i][col]=='Q'){
            return false;
        }
    }

    //left diagonal

    for(int i=row-1,j=col-1;i>=0 && j>=0;i--,j--){
        if(board[i][j]=='Q'){
            return false;
        }
    }

    //right diagonal
    for(int i=row-1,j=col+1;i>=0 && j<=n;i--,j++){
        if(board[i][j]=='Q'){
            return false;
        }
```

```
    }

   return true;

}
void print(char** board,int n){
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            cout<<board[i][j]<<" ";
        }
        cout<<endl;
    }
    cout<<endl;
}
void nqueen(char** board,int n,int row){
    if(row==n){
        print(board,n);
        return ;
    }

    for(int j=0;j<n;j++){
        if(isSafe(board,n,row,j)){
            board[row][j]='Q';
            nqueen(board,n,row+1);
            board[row][j]='X';
        }
    }

}
```

```
int main() {
    int n;
    cin>>n;
    char** board= new char*[n];
    //Initialize
    for(int i=0;i<n;i++){
        board[i]=new char[n];
        for(int j=0;j<n;j++){
            board[i][j]='X';
        }
    }

    nqueen(board,n,0);



    return 0;
}}
```

• For small values of N, it is feasible to calculate the number of solutions directly. Here are the numbers of solutions for N-Queens problems with N ranging from 1 to 5:

No. and No. of Solutions

N = 1:- 1 solution
N = 2:- 0 solutions (no solution exists)
N = 3:- 0 solutions
N = 4:- 2 solutions
N = 5:- 10 solutions

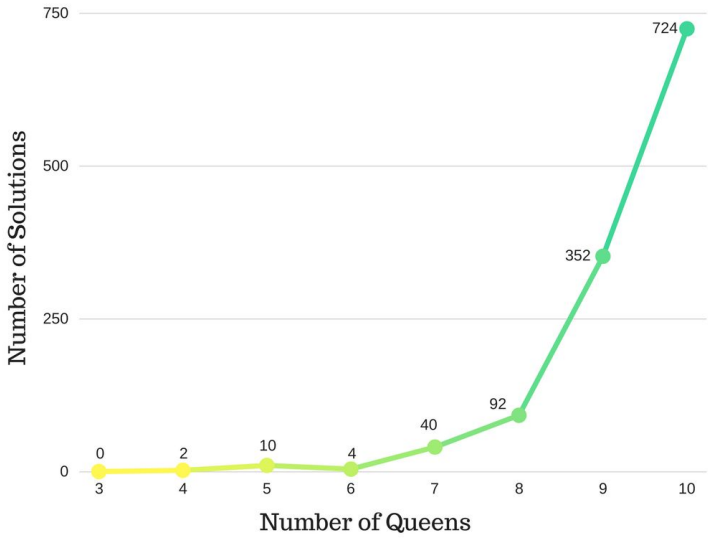


Figure 3: No of Solutions

For larger values of N, the number of solutions grows exponentially, making it computationally expensive to find and count all solutions.

• Mathematics in N-Queen

In the context of the N-Queens problem, mathematics plays a crucial role in understanding the problem, formulating constraints, and developing algorithms to solve it.

1. **Chessboard Representation:** Maths helps in representing N×N chessboard used in the N-Queens problem. Each cell of the board can be represented by coordinates (x, y), where $1 \leq x, y \leq N$. This mathematical representation enables efficient manipulation and tracking of the board's state during the solution process.

2. **Constraint Formulation:** The main constraint is that no two queens should be able to attack each other. By mathematically

defining these constraints, it becomes possible to evaluate the validity of a particular queen placement.

3. **Combinatorics and Permutations:** Mathematics provides combinatorial techniques to count and generate permutations of queen placements. Techniques such as factorial calculations, permutations, and combinations helps to explore the search space and create valid solutions.

4. **Algorithm Design and Analysis:** Mathematics plays a crucial role in designing and analyzing algorithms for solving the N-Queens problem. Techniques like backtracking, recursion, and dynamic programming can be applied to efficiently search through the solution space.

5. **Complexity Analysis:** Mathematics helps in evaluating the time and space complexity of algorithms.By evaluating the worst-case time and space complexity, mathematicians and computer scientists can assess the efficiency of different algorithms and determine their scalability for larger problem instances.

   • **Time Complexity:** The time complexity of the N Queen problem can be examined using Big O notation. In the worst case, when we need to consider all possible arrangements of queens,the time complexity can be approximated as **O(N!)**, where N is the size of the chessboard.

   The reason for this time complexity is that for each row, we need to consider all possible columns to place a queen. In the first row, there are N options, in the second row N-1 options (excluding the column where the queen is already placed), and so on. This leads to N*(N-1)*(N-2)*...*2*1 = N! possible configurations.

   • **Space Complexity:** The space complexity of N Queen problem is **O(N)**.This is determined by recursive stack used during backtracking algorithm.

In conclusion, the N-Queen problem is a fascinating puzzle with a rich history in computer science and mathematics.To solve it we can use different algorithms.

• References

1) Figure 1: https://images.app.goo.gl/F1jyC6UWcNvfweSeA
2) Figure 2: https://images.app.goo.gl/Zhtz7bU5s57PCZSo6
3) Figure 3: https://images.app.goo.gl/SX7Lrjnj2yigG3i38
4) Figure 4: https://heroic-sfogliatella-19d7ea.netlify.app/graph1.jpg
5) Book Reference : Discrete Mathematics and its Applications - Author: Kenneth H.Rosen
6)Our reference for understanding:
1.https://www.ijcaonline.org/archives/volume174/number6/buchade-2017-ijca-915417.pdf
2.https://youtu.be/xFv$_H l4B83A$