**BACKTRACKING**

Backtrack(x)

if x is not a solution

return false

if x is a new solution

add to list of solutions

backtrack(expand x)

**N-Queen Problem**.

In N-Queen problem, we are given an NxN chessboard and we have to place n queens on the board in such a way that no two queens attack each other. A queen will attack another queen if it is placed in horizontal, vertical or diagonal points in its way. Here, we will do 4-Queen problem.

Here, the solution is −

Here, the binary output for n queen problem with 1’s as queens to the positions are placed.

{0 , 1 , 0 , 0}

{0 , 0 , 0 , 1}

{1 , 0 , 0 , 0}

{0 , 0 , 1 , 0}

For solving n queens problem, we will try placing queen into different positions of one row. And checks if it clashes with other queens. If current positioning of queens if there are any two queens attacking each other. If they are attacking, we will backtrack to previous location of the queen and change its positions. And check clash of queen again.

Step 1 − Start from 1st position in the array.

Step 2 − Place queens in the board and check. Do,

   Step 2.1 − After placing the queen, mark the position as a part of the solution and then recursively check if this will lead to a solution.

   Step 2.2 − Now, if placing the queen doesn’t lead to a solution and trackback and go to step (a) and place queens to other rows.

   Step 2.3 − If placing queen returns a lead to solution return **TRUE.**

Step 3 − If all queens are placed return TRUE.

Step 4 − If all rows are tried and no solution is found, return FALSE.