CSC 831: ARTIFICIAL INTELLIGENCE

N QUEEN PROBLEM USING BEST FIRST SEARCH

OGUNJINMI OYERONKE PRISCILLIA

179074021

ABSTRACT

Finding solution to N Queen problem is a difficult problem as there are 4,426,165,368 possible arrangements of eight queens on an 8 x 8 board, but with only 92 solutions. However N Queen problem is a solvable problem, applying the right algorithm helps make solving the problem easier and faster. Specific algorithms in artificial intelligence can be used to solve this problem. In this work best first search algorithm was used to sole the N queen problem.

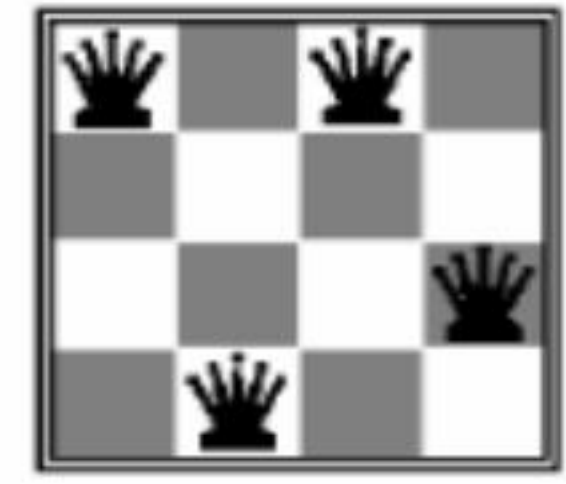
INTRODUCTION

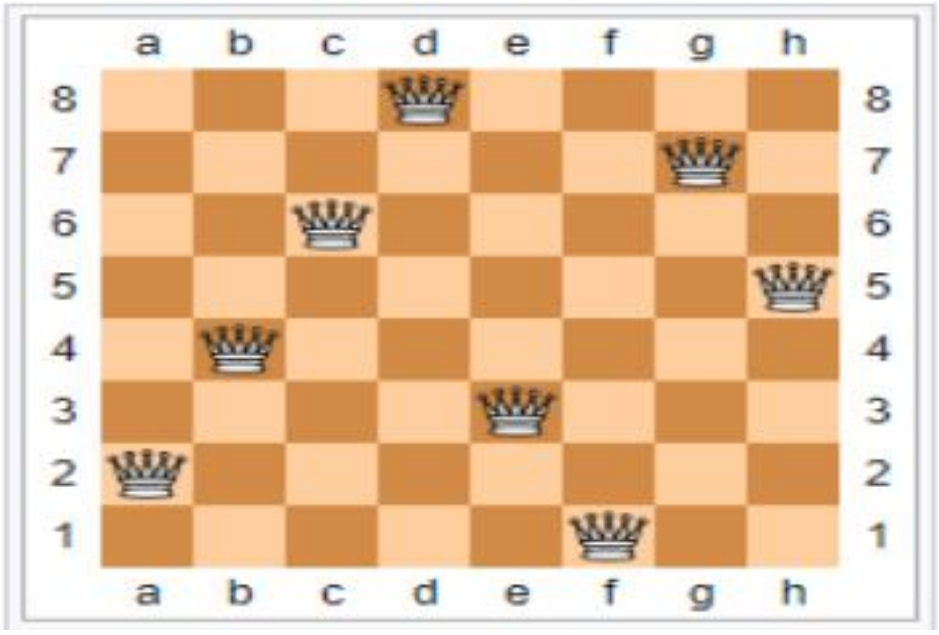
N Queen problem is the problem of placing N number of Queens on an N x N board so that no queens threaten each other. Thus a solution requires that no two queens share the same row, column or diagonal. Two queens are attacking each other if one of them can travel horizontally, vertically, or diagonally and hit the square the other queen is on. The problem is to place the queens such that no two queens are attacking each other.

An object-oriented solution to the eight-queens problem differ from a solution written in a conventional imperative programming language. In a conventional solution, some sort of data structure would be used to maintain the positions of the pieces. A program would then solve the problem by systematically manipulating the values in these data structures, testing each new position to see whether it satisfied the property that no queen can attack any other.

Problem Definition

The n-queens problem, is a search problem whereby you want to place N queens on an N × N board (square grid). Each queen occupies one square on a grid and no two queens share the same square. Two queens are attacking each other if one of them can travel horizontally, vertically, or diagonally and hit the square the other queen is on. The problem is to place the queens such that no two queens are attacking each other. For instance, the board below is not a legal solution: two queens are on the same vertical line.





One solution of eight queen problem.

As with many similar problems, the solution to the eight-queens puzzle involves two interacting steps: generating possible partial solutions and filtering out solutions that fail to satisfy some later goal. This style of problem solving is sometimes known as generate and test paradigm.

To find a solution it is clear that the queens will need to communicate with each other. Realizing this, a second important observation can be made that will greatly simplify our programming task namely, each queen needs to know only about the queens to her immediate left. Thus, the data values maintained for each queen will consist of three values: a column value, which is immutable; a row value, which is altered in pursuit of a solution; and the neighboring queen to the immediate left.

Initialization

We will divide the task of ending a solution into parts. The method initialize establishes the initial conditions necessary for a queen object, which in this case simply means setting the data values. This is usually followed immediately by a call on find Solution to discover a solution for the given column. Because such a solution will often not be satisfactory to subsequent queens, the message advance is used to advance to the next solution. A queen in column n is initialized by being given a column number, and the neighboring queen. At this level of analysis, we will leave unspecified the actions of the leftmost queen, who has no neighbor.

Finding a Solution

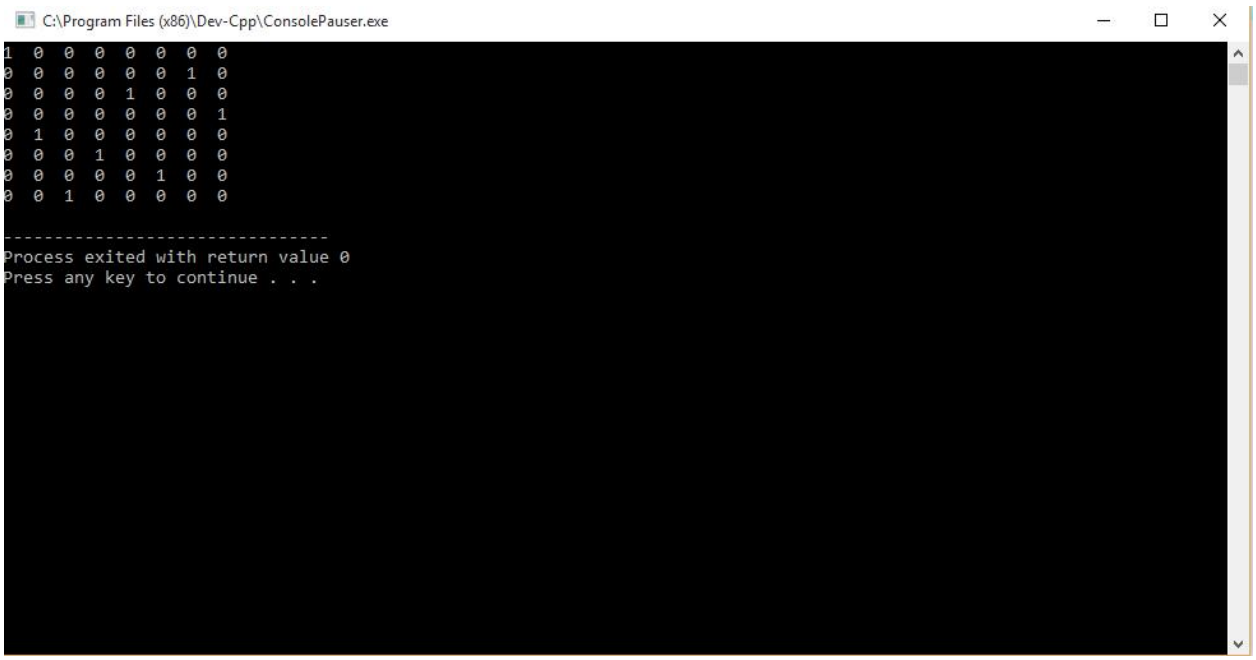
To find a solution, a queen simply asks its neighbors if they can attack. If so, then the queen advances herself, if possible (returning failure if she cannot). When the neighbors indicate they cannot attack, a solution has been found

Advancing to the Next Position

The procedure advance divides into two cases. If we are not at the end, the queen simply advances the row value by 1. Otherwise, she has tried all positions and not found a solution, so nothing remains but to ask her neighbor for a new solution and start again from row 1.

Best First Search Algorithm

1. OPEN = [Initial State]
2. While OPEN is not empty
3. Do
4. Remove the best node from OPEN, call it n
5. If n is the goal state, back trace path to n and return path
6. Create n's successors
7. Evaluate each successor, add it to OPEN and record its parent.


C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe
1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 0

Process exited with return value 0
Press any key to continue . . .

References

http://web.engr.oregonstate.edu/~budd/Books/oopintro3e/info/chap06.pdf

https://www.researchgate.net/figure/A-solution-to-the-8-queens-problem-presented-as-5-1-8-4-2-7-3-6_fig1_257549155