Project 2

Jonathan Gregory

CS 461

September 4, 2017

Report

This program is able to solve the n-Queens problem by implementing a hill climbing search. The program first asks the user for the desired value for n. Then it computes an acceptable solution. It does this by first creating a vector of Queen struts that is placed on a double array. That double array holds the current attack value of the queen, highest potential attack, the current row, along with the number of conflicts that can be removed when moving to the optimal row.

Initially, the queens are placed on a random row while only one queen is present per column. There is a separate double vector that holds the value of every cell in regard to the attackable value. These are the sum of all the queens that could "take" that cell.

It will now traverse each column seeing if there is a value that has less queens attacking it and will move the queen to that desired location. It will then perform a validation on the board to confirm that it fits the parameters stated in the directions. If it is and there are no better spots for the board to be adjusted, it will either result in a local minimum or a plateau. This goes on until 200 exact plateau results.

Success is determined if there is a board that has all the other queens not attacking each other or the lowest attack positions. Then it will print out the matrix. After that, the program asks the user if they want to try another. If they are so inclined, the user can then enter a new parameter for the value of n. It should be noted that the program will rarely (like 1 out of 20) run into out of bounds when a plateau is reached but it will correct itself and print out the correct matrix after it restarted calculating it. There is only 1 file for this solution. It is all in main.cpp.