Assignment 1: N-Queen's Problem

By: Alex Darcovich, Aliya Conrad, Devon Gough, Flora Diep & James (Donghao) Wang

Artificial Intelligence

CISC 352 | Winter 2021

Queen's University

Group Contribution:		
Group Member	**Student Number**	**Contribution**
Alex Darcovich	20070104	➔ Code
Aliya Conrad	20080290	➔ Report
Devon Gough	20130120	➔ Code/Testing
Flora Diep	20055924	➔ Report
James (Donghao) Wang	20119632	➔ Report/Testing

Algorithm:

Given an integer N (represented as a value between four and ten million) from the input file "nqueens.txt", the program will construct an N x N chess board (a sequence of elements from four to ten million) that distributes N queens across the board. The MIN-CONFLICTS algorithm can be implemented with the assistance of helper functions to solve and validate safe tile positions to place each individual queen such that each queen is placed on different rows, columns, and diagonals of the board.

We use a function that initializes the chessboard, taking the size of the board as an argument and placing N queens in optimal positions on an initialized N x N grid. Such positions are found using the following analytical solution where n is the number of inputs and A(n) is a function that maps n to a set of 2-tuples which correspond to the non-conflicting placements of the n queens with the first entry of a 2-tuple corresponding to the row and the second entry of a 2-tuple corresponding to the column. [1][2]

If n is even and $\exists m \in \mathbb{N}$ such that $n = 6m + 2$, then:

$$A(n) = \left\{ \begin{matrix} \left(i, 1 + \left[\left(2(i-1) - 1 + \frac{n}{2}\right) \text{ (mod n)}\right]\right) \\ \left(n + 1 - i, n - \left[\left(2(i-1) - 1 + \frac{n}{2}\right) \text{ (mod n)}\right]\right) \end{matrix} \;\middle|\; i \in \left[1, \frac{n}{2}\right] \right\}$$

If n is even and $\nexists m \in \mathbb{N}$ such that $n = 6m + 2$, then:

$$A(n) = \left\{ \begin{matrix} (i, 2i) \\ \left(\frac{n}{2} + i, 2i - 1\right) \end{matrix} \;\middle|\; i \in \left[1, \frac{n}{2}\right] \right\}$$

If n is odd and $\exists m \in \mathbb{N}$ such that $n = 6m + 3$, then:

$$A(n) = \left\{ \begin{matrix} \left(i, 1 + \left[\left(2(i-1) - 1 + \frac{n-1}{2}\right) \text{ (mod n)}\right]\right) \\ \left(n - i, n - \left[\left(2(i-1) + \frac{n-1}{2}\right) \text{ (mod n)}\right]\right) \\ (n, n) \end{matrix} \;\middle|\; i \in \left[1, \frac{n-1}{2}\right] \right\}$$

If n is odd and $\nexists m \in \mathbb{N}$ such that $n = 6m + 3$, then:

$$A(n) = \left\{ \begin{matrix} (i, 2i) \\ \left(\frac{n-1}{2} + i, 2i - 1\right) \\ (n, n) \end{matrix} \;\middle|\; i \in \left[1, \frac{n-1}{2}\right] \right\}$$

Figure 1. The analytical solution of a N-Queens Problem for any n >= 4 used in the algorithm.

A for loop is then iterated eight times, generating random indices and random rows to shuffle and create conflicts in our initial board to be solved using the MIN-CONFLICTS algorithm. This is done each time by re-assigning a random index of the chess board to a random row, thereby changing the row of the queen initially corresponding to that index. The row of each moved queen is appended to a list of conflicts for use by our algorithm. Initializing our board in this way allows for precise testing of our algorithm by giving us relative control over the difficulty of the solution as we increase or decrease the number of conflicts. This means that if the number of iterations in the for loop is increased or decreased, the running time of the algorithm will respectively increase or decrease with greater or fewer respective conflicts to be resolved within the chessboard configuration.

With an efficient algorithm, the program is able to improve the positioning of our queens. With each iteration, the objective is to reconstruct the chessboard with the least number of conflicts possible.

While conflicts exist, our solver function selects a random conflicting queen from the list created during the board initialization and applies the MIN-CONFLICT algorithm to move the piece.

Starting with the first row, the rows are iterated through to search for the positions with the least number of conflicts per row. Subsequently, diagonals are iterated until the position with the least number of conflicts is found. Then, by sequentially checking each tile of the column corresponding to the randomly chosen conflicting queen and determining the number of conflicts for each tile, a list of tiles containing no conflicts can be made. This enables the program to strategically move each queen to a unique row within its column.

Our solver implements the solution function to check whether conflicts exist before applying MIN-CONFLICT. The solution function examines the length of the board and the length of the set of the board, which is equal to the board list with duplicates removed. The lengths should be equivalent if there is exactly one queen in every row, ensuring that there are no duplicate pieces. It then determines whether the pieces are unique across the board diagonals in the same manner. This function will return True if there are no conflicts and will return False if there are any conflicts. If the solution found satisfies all conditions of the constraints, the program will return a single list of integers where each entry corresponds to a row of a queen.

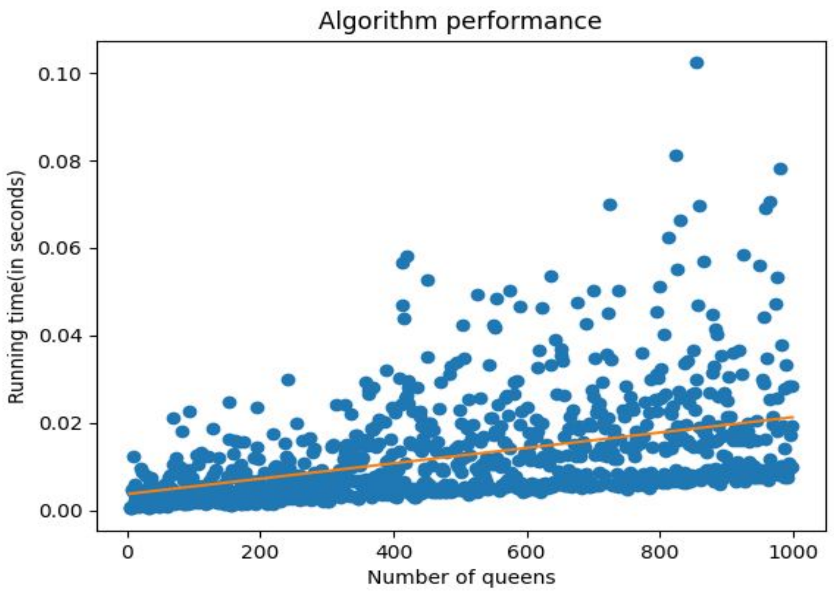

Figure 2. Algorithm performance illustrated by linear time complexity O(n).

Figure 2 manifests the effect of increasing the number of queens (>= 4) on the running time of the algorithm. Omitting some outliers, most of the data points in the graph construct a linear line of best fit. In the actual code, the only portions of instructions that are dependent on the data size are for loops, none of which are nested and all of which are sequential. Since each for loop takes O(n) time to run, the algorithm has a total time complexity of O(n).

References

[1] *Constructions for the Solution of the m Queens Problem.* (1969). Mathematics Magazine. https://www.tandfonline.com/doi/abs/10.1080/0025570X.1969.11975924?journalCode=umma20

[2] *Explicit Solutions to the N-Queens Problem for All N.* (1991). Mathematics Magazine. https://dl.acm.org/doi/10.1145/122319.122322