Project #1 - Cella Rule 90

Mauricio Macias

Gabriel Rivera

Mitchell Norseth

**Complexity Order of our Algorithm**

For this project we wrote a program to display the generational progress of Wolfram’s Rule-90 cellular automaton. We used the brute force algorithm that we wrote in javascript that initializes a 2d array it checks each cell in the in the array individually, row by row, in order to check if a cell needs to be shaded as an ‘X’ (1) or passed (0). It does this by referencing the previous row in the grid to check if certain cells are shaded. It depends on the pattern of shaded/non shaded cells in the previous row. This is what tells each sequential cell if it needs to be shaded or not.

When it comes to the Big-O running time of our algorithm it is pretty easy to see that the breadth of our algorithm comes down to a single nested for loop:

for( var r = 10; r < 400; r = r + 10) {

        for ( var c = 10; c < 400; c = c + 10) {

        . . .

}

}

This gives our algorithm the time O(N^2) which means it is quadratic. The running time of these two loops is proportional to N^2. When N doubles, the running time increases by N^2 -> N \* N.

Our algorithm takes in no input, but rather just loops through the array after it’s initialized, so we don’t have to worry about the size of our input.