Joseph Rowley
CS 101
Winter 2013
Manfred Warmuth

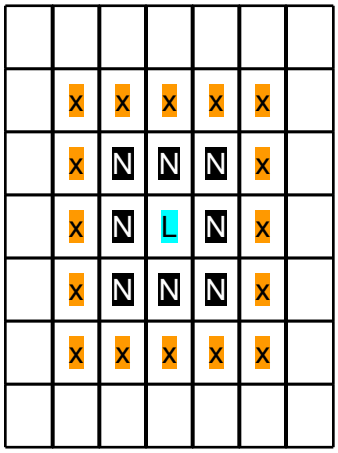
Game of Life in HTML 5

I choose to create the game of life in HTML 5 because I had no prior real experience with HTML5 and wished to try my hand at it. As a result I had some challenges with implementing the algorithm exactly as they were laid out in the instructions but here is how my program works.

1. I parse inputs and add them to a 2D grid and liveArray. This requires a constant amount of time for each element imputed, so is in the O(n) time.
2. For each living cell, I look for neighboring cells that aren't living and add them to a broArray. Again, I am processing each living cell, so that in itself takes constant time. Then calculating a given cells neighbors is another constant operation, so O(n) time again.
3. I then calculate the number of living cells around each given cell in the liveArray and broArray. For this step I have to iterate through both the liveArray and broArray. This isn't the most efficient method because I could have each live cell just increase the count of the neighboring cells, however, in practice this became ugly and wasn't working for me so I opted for this less efficient but easier to implement system. This means that each neighboring cell also has to scan it's neighbors for live cells.

L is a single living cell.
N is a neighboring cell
X are extra dead cells that are counted unnecessarily due to the less efficient algorithm

Sample Grid



Although this is the most efficient way to do this step, it still O(n) time as for each live cell there are at most 8 neighbors and for that cell there are at most another 8 neighbors. So this is a more intensive operation but still a constant time directly linearly proportional to the number of cells added initially.

4. I then apply the rules to both the liveArray and broArray, deciding if a given cell should be continue in it's existing state or change states. This operation is O(n) as it as simple as iterating through 2 arrays and populating another array.
5. Finally, I update the display by clearing it, and painting all living cells in the liveArray. The display is updated by iterating through an array again, this O(n) time.

This process loops each time a user progresses to a new step.

As you can see nothing in this process is over O(n) time thus, my algorithm runs in O(n) time.