COMPARING CELLULAR AUTOMATA

Comparing CA: Motivation

Often the case is that as the number of iterations increases two CA with different updating rules can start to look very similar. In attempt to classify CA it is useful to determine how similar to two different CA are. To solve this problem, we implemented the ability to plot the time evolution of two different CA in our animation (as opposed to just a single CA that the animation had originally) with the first CA live cells being colored in green and the second's in blue. All squares in which both CA had a live cell were colored in red. Additionally, the number of common squares was kept track of in a counter. Prior to adding this, the applet had already been displaying the iterations in GUI. With the counter keeping track of the common squares, we could now see the number of common squares versus the number of iterations for two CA. We now can define function f(CA1, CA2, iteration_number) which takes in two different CA and a particular iteration number and returns the number of common squares that they share. From a mathematical standpoint studying these functions for differing kinds of CA and for large iteration intervals could give some interesting results. For examples these functions might be analytical or might themselves have chaotic behavior.

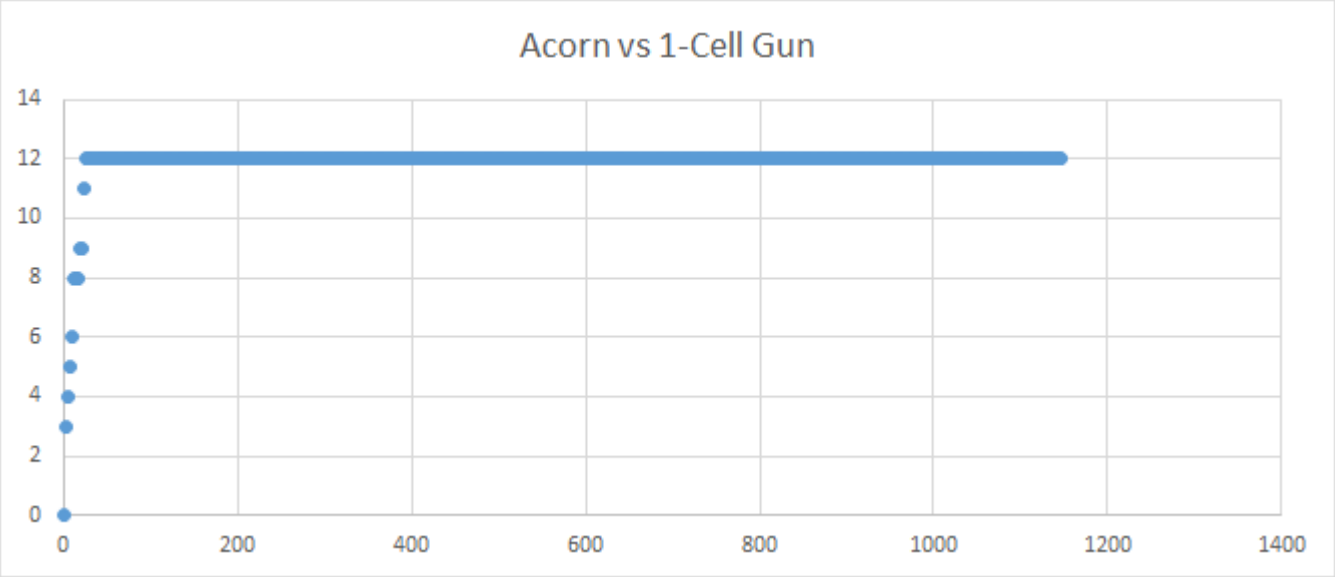
Comparing CA: Implementation

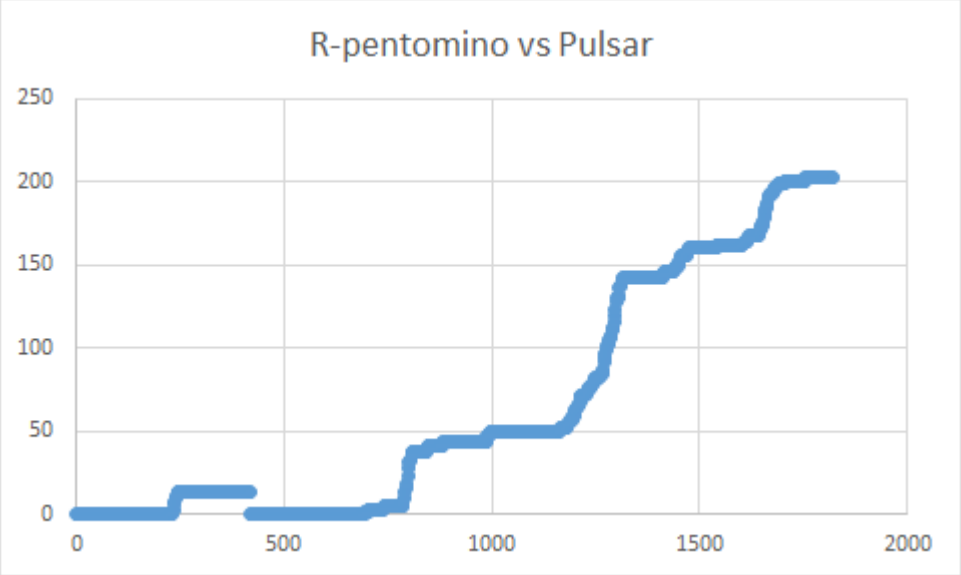
To implement the CA comparison, we had to add another mode to the mode selection unit and we had to add a way to select the two different CA. Basically the now and next matrices which govern the evolution of a single CA had to be duplicated to include a now2 and a next2 matrices. All the code associated with the now and next matrices had to be replicated to support the now2 and next2 matrices as well. Although this approach seemed straightforward initially, it took quite a few versions in order to get the code to appropriately display the correct time evolutions for both the CA. One issue that we encountered was that the one of the two CA always seemed to die after a certain number of iterations. However, we did not introduce any sort of coupling to the two CA that would lead to that type of behavior (introducing coupling is interesting idea though and we would like to analyze this is greater detail in future works if possible). Due to this many combinations of CA were unable to produce meaningful statistics, since one would always die out. In most cases when there was this die out, the two CA did not even have the chance to interact at all. The solution to this problem was to add an additional variable that would check the stability of the CA that was dying out. If this variable was not true, then only were we to apply the updating condition for the now2 and next2. Although this seemed to work a fair amount of CA combinations, we are still not completely sure if the problem is fixed.

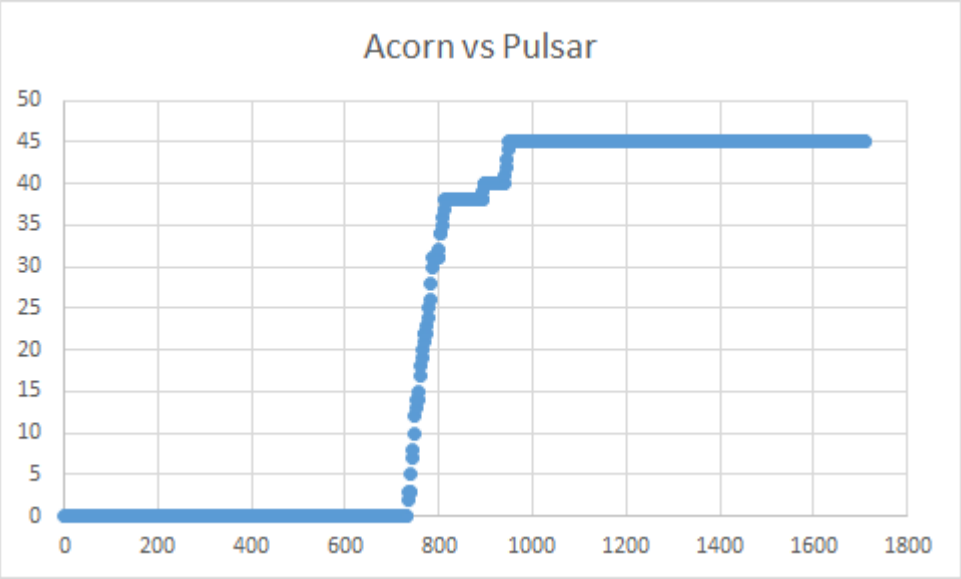
Comparing CA: Results

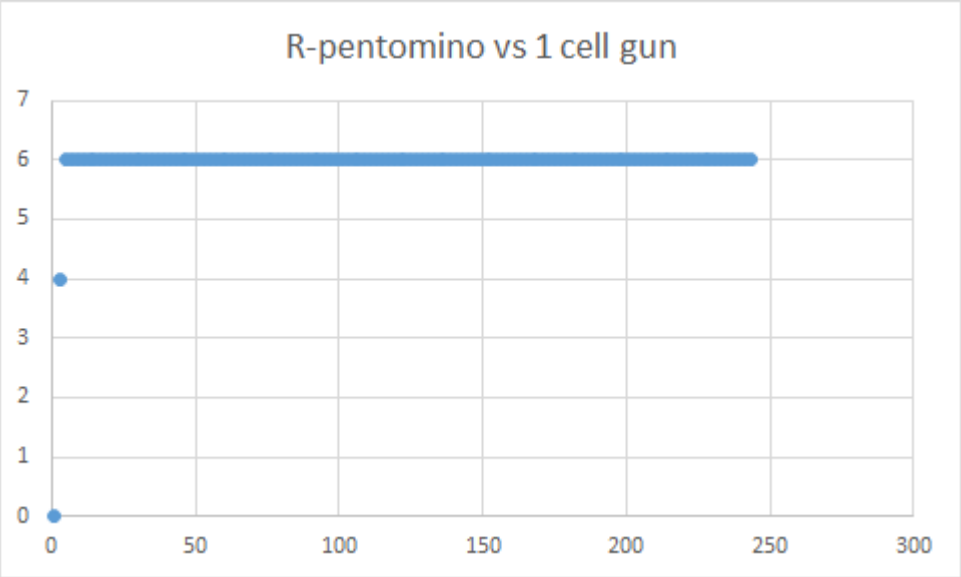
Due to the large sample space of possible combinations for CA1 and CA2 and time considerations, we were not able to plot every combination of CA1 and CA2 although we did run and observe the behavior of many other different combinations. The data shown below shows the number of common squares on the y axis and the iteration number of the x axis. During our observation of the sample space, we determined that the number of

squares for the large majority of times increase discontinuously and are always at least an order of magnitude smaller than the iteration number (except for the case when both CA are the same). It would be great to have regression equations that for each of these graphs, but due to technical issues we were not able to obtain these.









Comparing CA: Future Directions

Introducing two CA (or any number of CA) all evolving on a single board introduces can produce very interesting behavior when they become coupled. For example if there two cellular automata so happened to have a common square, then one of the CA (or both) would have all of its nearest neighbors to that cell would become alive (or dead). In this sense each CA can represent species interactions in biology or even chemical reactants in chemistry.

This is an example of implementing coupling between the two CA.

```
                    else if (now[i][j] && now2[i][j]) {
                      //now[i+1][j+1]= true;
                            //now[i-1][j-1]= true;
                            //now[i][j+1] = true;
                            //now[i][j-1] = true;
                            //now[i+1][j] = true;
                            //now[i-1][j]=true;

                            commonsquares++;
                            commonsquaresPrompt.setText("common squares
        #"+commonsquares);
              bufferGraphics.setColor(Color.red);
                                    }
```