Sequential Dynamical System Game of Life

Mi Yu

March 2, 2015

We have been studied sequential dynamical system for nearly 7 weeks now. We also studied the game of life. We know that in the game of life, everything is updated simultaneously. Our group has been thinking about applying the sequential rules on the game of life. We found out many cool things that will happen if you apply sequential rules on the game of life. First of all, let me define the rules for the SDS game of life.

Definition 1. *The universe of the SDS game of life is an infinite two-dimensional orthogonal grid of square cells, each of which is in one of the two possible states: alive or dead. The boundary we observe and update will contain every live cell and its eight neighbors. We will only update the cells in this defined boundary. In order to update this boundary, we will look at the alive cells and neighbors to those cells. We have the local function on each cell's neighbors, which are the calls that are horizontally, vertically, or diagonally adjacent, that follows the following rules:*

1. *Any live cell with fewer than two live neighbors dies, as if caused by loneliness.*
2. *Any live cell with 2 or 3 live cells survive.*
3. *Any live cell with more than 3 live cells dies, as if caused by overpopulation.*
4. *Any dead cell with 3 live neighbors live, as if caused by reproduction.*
5. *The boundary we observe changes whenever a cell in our boundary dies or a dead neighbor comes alive when updating the grid locally.*

The update order is from the very left to the very right, and then if it reaches the very right, then it moves from one down to the next row, and starts the entire process again.

Now, let us play an example of the SDS game of life. Look at the following initial state:

Figure 1

The readers might recognize that this is a block in the game of life. The block is a still life in Conway's game of life. The normal question to ask is that if a block is still a still life in the SDS game of life. Now let us do the update. If we start from the top. We know that there is nothing that will come alive on the first row since there are only totally two live cells in the second row. Let us look at the second row. The first cell will not be alive since it only has two live neighbors. The second cell and the third cell will still be alive since they have three neighbors. The fourth cell will still be dead since it only has two live neighbors. Repeating the process, we will have the original graph as our output after one update. Then, we have the following proposition:

Proposition 1. *Any still-life shapes in Conway's game of life is still a still life in the SDS game of life.*

Proof. This is trivially true. If a shape in Conway's game of life is a still life, it means that if we apply the rules to every cell at the same time, every cell will still be the same as before. Now, let us put this shape in the SDS game of life. If we look at each cell sequentially, we know that there will not be any change for any cell since the local rules of SDS game of life is exactly the same as the rules for Conway's game of life. We can prove this by induction, but since it is trivial, we will save this for the readers. Then, even if we run the still lives in the SDS game of life, since there is nothing changed in any step in the update order, the final outcome will not be changed, which means that the shape is still a still life. □

Now, we might wonder if an oscillator or a spaceship in Conway's game of life is still an oscillator and a spaceship. We can try the following initial states. First of all, look at the following initial state:

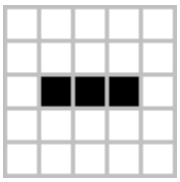

Figure 2

As the readers might recognize, this is a blinker, which is an oscillator in the original game of life. However, if we do the sequential update based on the update order and rules we defined earlier, we will have the following graphs as the local configurations.

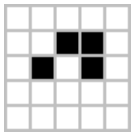

Figure 3: After one update

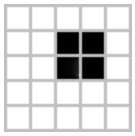

Figure 4: After two updates

We can see that after two updates, the blinker becomes the block, which is a still life. Thus, we know that oscillators will not still be an oscillator. This is also true if we think about the rules. This time we are updating sequentially, thus all the local changes will affect all the later local changes. We know that oscillator probably will not be able to maintain as an oscillator. Let us do the glider to make sure that spaceships cannot maintain as spaceships either. In case you forget the shape of a glider, the following initial state is a glider:

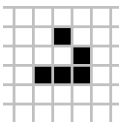

Figure 5

If we do the updates, we will have the following states as the results:

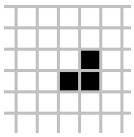

Figure 6: After one update

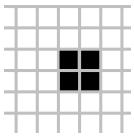

Figure 7: After two updates

Again, the glider in Conway's game of life turns into a block. We can be ascertain that spaceships and oscillators in Conway's game of life behave differently in the SDS game of life. Actually, all three examples we had earlier all end up in a block. People might guess that if all the shapes in the SDS game of life will become a block. Then, the SDS game of life will be extremely boring. Luckily, this is not true. We made an algorithm that randomly generates living cell and run the rules. Surprisingly, although it seems that everything will stabilize at first, almost all the randomly generated shapes actually explode. However, if all the shapes just explode or stop growing, the SDS game of life will also be boring. After randomly trying several examples, we have that this is also not true. We found two spaceships in the SDS game of life. The following two shapes are spaceships:

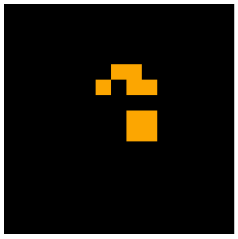

Figure 8: Spaceship 1

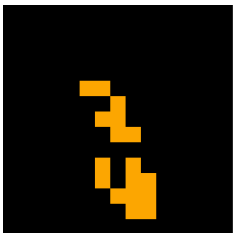

Figure 9: Spaceship 2

If we update the two figures, we will find out that the first spaceship moves towards upright cover at speed $\frac{1}{2}$. Since the spaceship will become the following shape and then change back:

Figure 10: Spaceship 1 after 1 update

Spaceship 2, however, moves to the bottom right corner at speed 1, which is the fastest speed a shape can move without any support. We used the random generating algorithm to find out the two spaceships. We do not know if there are still other spaceships out there , and using our current method will be very limited to find other spaceships since slow spaceships are not able to escape the giant living cell groups. Same thing will happen to the oscillators. We have no luck finding any oscillators yet. We cannot prove that there is none, either. We will continue the research and try to find some oscillators if it is possible. However, we have already notices many things here. First, although it seems like the SDS game of life can be entirely a death or a chaotic region, we can certainly find very organized life forms such as still lives and spaceships. This makes the SDS game of life hard to be classified into any classification of cellular automata. Also, the speed of spaceships are also very interesting in the SDS game of life. The speed limit seems to vary based on the directions. Also, there seems to be no spaceships moving left based on our update order. Next we will talk about the classifications of cellular automata and some proofs about the speed limits in the SDS game of life.

1 Classification of Cellular Automata

If you recall back to week 2, we quickly went over a classification system for cellular automata in general. These are the four basic classes:

1. Class 1: Nearly all initial patterns evolve quickly into a stable, homogeneous state. Any randomness in the initial pattern disappears.

2. Class 2: Nearly all initial patterns evolve quickly into stable or oscillating structures. Some of the randomness in the initial pattern may filter out, but some remains. Local changes to the initial pattern tend to remain local.

3. Class 3: Nearly all initial patterns evolve in a pseudo-random or chaotic manner. Any stable structures that appear are quickly destroyed by the surrounding noise. Local changes to the initial pattern tend to spread indefinitely.

4. Class 4: Nearly all initial patterns evolve into structures that interact in complex and interesting ways, with the formation of local structures that are able to survive for long periods of time. Class 2 type stable or oscillating structures may be the eventual outcome, but the number of steps required to reach this state may be very large, even when the initial pattern is relatively simple. Local changes to the initial pattern may spread indefinitely. Wolfram has conjectured that many, if not all class 4 cellular automata are capable of universal computation.

Now, the line between Class 3 and Class 4 is quite vague. SDS Life does display similarities to both. It seems to evolve in a very chaotic manner and the surrounding noise does eliminate all existing stable structures so that may make it seem that it is Class 3. However, with the discovery of spaceships this may mean that SDS Life is actually a Class 4 Cellular Automata. So as of right now, we will classify this as a Class 3.5 Cellular Automata until we can prove or disprove other properties that would make it a Class 4.

2 Open Problems

This week we stumbled upon many questions that had us stumped. Due to the limitation of only having randomly made grids in our program that runs the game, we had to rely strictly on that (which required many trials and much luck) and our brains in attempt to create the objects we wanted to find. These are the following open problems we have:

1. Do oscillators exist?

2. What is the speed limit for each direction a spaceship can move?

   Conjecture: We believe that moving down-right is the fastest direction a spaceship can travel, then up-right, next is down-left, and finally up-left is the the slowest direction.

3. Are there any directions that a spaceship cannot travel?

These questions we will attempt to tackle for next weeks presentation and possibly continue to look at them occasionally even after the quarter concludes.