Regularity in Conway's Game of Life

Research · October 2015

DOI: 10.13140/RG.2.1.5098.5686

1 author:

Caleb A Koch
Cornell University
5 PUBLICATIONS **0** CITATIONS

SEE PROFILE

Contents

List of Figures

List of Tables

1 Introduction

In the 1940s, the eminent physicist and mathematician John von Neumann began producing work related to solving the mystery of self-reproduction as employed in biological systems[1]. He was primarily interested in the use of computing devices to model the complex behavior of organisms. The development of this idea came after a suggestion by Stanislaw Ulam who introduced the possibility of using cellular automata as a means of achieving von Neumann's goals. Thus, von Neumann was determined to answer the question, "What kind of logical organization is sucient for an automaton to be able to reproduce itself [2]?" He proposed an elegant solution to this question in his book Theory of Self-Reproducing Automata (completed and published posthumously in 1966), which successively revolutionized the discipline of computer science by introducing a robust outline of how cellular automata can be employed to model a universal Turing machine[2] (a universal Turing machine is an abstract idea of a device that can perform any computable task[3]).

Since then, cellular automata have been used in a variety of ways and have taken on multiple definitions, but it is generally agreed upon that all cellular automata consist of "a collection of 'colored' cells on a grid of specified shape that evolves through a number of discrete time steps according to a set of rules based on the states of neighboring cells[4]." As such, every cellular automata consists of a certain type of cellular space and a transition rule. The cellular space can be described in terms of any d-dimensional regular lattice of cells with finite boundary conditions. Each cell has k number of states where k is most commonly used as a positive integer $(k : k \in \mathbb{Z}^+)$ [3]. The set of states is denoted Σ, making $k = |\Sigma|$. The state of cell C located at index i and at time t is denoted C_i^t. (Note that $C_i^t \in \Sigma$). The neighborhood of C consists of the state of the adjacent cells or $C_{i\pm1}^t$ and is used to determine the state of C at time $t+1$. The function φ, known as the transition rule, is used to compute this state. Thus, $C_i^{t+1} = \varphi(C_{i-1}^t, C_i^t, C_{i+1}^t)$. It should be noted that different cellular automata use different neighborhoods. Namely, a 2-dimensional cellular automaton uses either the Moore neighborhood or the von Neumann neighborhood shown in figure 1.

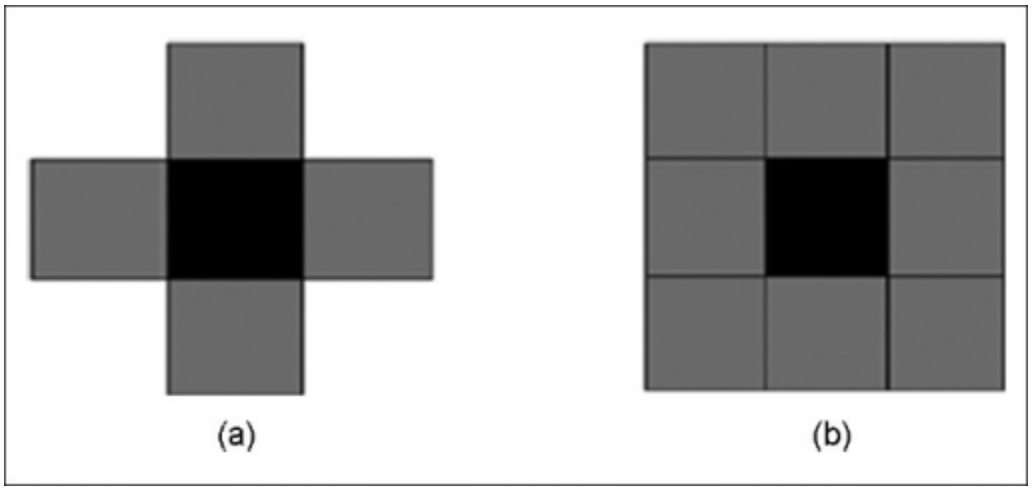


Figure 1: *Two different neighborhoods in a 2-dimensional cellular automaton. (a) gives a depiction of the von Neumann neighborhood with the black cell in the center being updated. In contrast, (b) represents the Moore neighborhood, with the center black cell being updated.*

2 Statement of Task

This paper will seek to further explore the notion of cellular automata with respect to their mathematical basis. Specifically, we will investigate a certain 2-dimensional cellular automata represented on an integer-lattice, $\mathbb{Z}^2$, known as Conway's Game of Life by proposing an original pattern. We will then investigate a certain aspect of the cellular automata that remains consistent for various systems.

3 Game of Life

In the 1970s, Cambridge Professor John Conway invented a 2-dimensional cellular automaton consisting of a Moore neighborhood that he called "life." In doing so, he was hoping to be able to study the macroscopic behaviors of a population [5]. As such, he experimented with various transition rules and eventually decided on this setup [3]:

- $\Sigma = \{0, 1\}$ where a state of 0 represents a "dead" member of the population and 1 represents an "alive" member of the population
- The transition rule consists of either the death of a member (going from $1 \to 0$), birth of a member ($0 \to 1$), or no change ($0 \to 0$ or $1 \to 1$)
- With zero or one bordering member alive in a cells neighborhood, death occurs due to loneliness
- With two or three members alive in the neighborhood, there is no change to an alive member
- If a dead members neighborhood consists of three live members, a new member is "born" ($0 \to 1$)
- If a live members neighborhood consists of more than three live members, it will die due to overpopulation

To propose a formal definition of φ, let the state of a cell C at position (m, n) in the 2-dimensional Cartesian grid (i.e. $\mathbb{Z}^2$ integer lattice) such that $m, n \in \mathbb{Z}$ at time t be given as $C^t_{m,n}$. Additionally, let the neighborhood N of C equal
$\{C^t_{m+1,n}, C^t_{m,n+1}, C^t_{m+1,n+1}, C^t_{m-1,n}, C^t_{m,n-1}, C^t_{m-1,n-1}, C^t_{m+1,n-1}, C^t_{m-1,n+1}\}$

We can represent the summation of the elements of N as

$$S = \sum C^t_{g,h} : g \in \{m-1, m, m+1\}, h \in \{n-1, n, n+1\} \wedge (g \neq m \vee h \neq n)$$

It follows that if

$$((C^t_{m,n} = 1 \oplus C^t_{m,n} = 0) \wedge (S = 3)) \to C^{t+1}_{m,n} = 1$$

OR if

$$((C^t_{m,n} = 1) \wedge ((S \neq 2) \wedge (S \neq 3)) \to C^{t+1}_{m,n} = 0$$

In any other scenario $C^t_{m,n} = C^{t+1}_{m,n}$.
We can combine this to form a piecewise function:

$$C^{t+1}_{m,n} = \begin{cases} 1 & : S = 3 \wedge C^t_{m,n} = 0 \\ 0 & : S \neq 3 \wedge C^t_{m,n} = 0 \\ 1 & : (S = 2 \vee S = 3) \wedge C^t_{m,n} = 1 \\ 0 & : (S < 2 \vee S > 3) \wedge C^t_{m,n} = 1 \end{cases}$$

3.1 Example Simulation

From these rules, we can run a simple simulation. Suppose we start out with three "alive" cells lined up in a row. If we differentiate the states of the cells by assigning the color black to cells with the value of 1 (members that are alive) and the color white to all other cells (in this case, dead cells), our automaton at $t = 0$ would look like figure 2.

Further, assume that the middle black cell is at position $(0, 0)$. Thus, $C^0_{0,0} = 1$ and $C^0_{0,1} = 1$ and $C^0_{-1,0} = 1$. In order to determine how the automaton will evolve from $t = 0$ to $t = 1$, we must look at each individual cell and its neighborhood, then apply the transition rule. For the cell at position $(0, 0)$, we know that

$$N = \{1, 0, 0, 1, 0, 0, 0, 0\}$$

and hence

$$S = 1 + 0 + 0 + 1 + 0 + 0 + 0 + 0 = 2$$
$$\therefore \varphi(N) = C^{t+1}_{0,0} = 1$$

If we repeat this process for the cell, $C^0_{0,1}$, we find that $\varphi(N) = 0$. Similarly, for the cell at $(-1, 0)$, we find that $\varphi(N) = 0$. In contrast, the cells at $(0, 1)$ and $(0, -1)$, become alive, due to the fact that both of their neighborhoods consist of exactly three live cells. Thus, the automaton at $t = 1$ looks like figure 3.

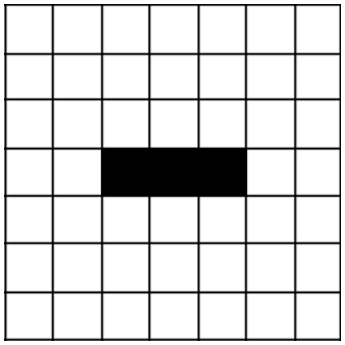


Figure 2: *Initial configuration.*

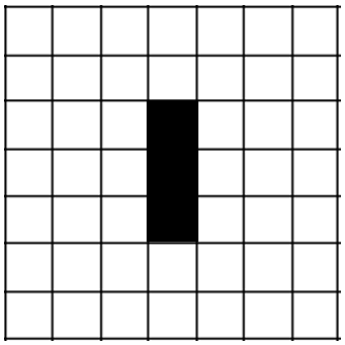


Figure 3: *Cellular automaton at t=1.*

If we rotate figure 3 90°, we end up with the same configuration as that of figure 2. Thus, we know that at $t = 2$ the automaton will be identical the automaton at $t = 0$. Furthermore, we know that the automaton at $t = 3$ will be identical to the automaton at $t = 1$, and so continues the cycle for as long as t is allowed to increase. Thus, this shape oscillates with a period of 2 generations. Many other oscillators have been identified in Conway's game of life with varying periods (the vast majority of which repeat themselves after two generations). This particular oscillator depicted in figures 2 and 3 is known as a blinker.

With a basic understanding of these cellular automata, one naturally begins to pose questions such as *what kinds of patterns can arise in Conway's Game of Life and how do they behave as t increases? How do basic polygonal shapes evolve in Life? Can we predict their evolution?*

4 Definitions

To explore these questions, we start with an elementary, regular polygon: a square. Specifically, we want to understand how a square of varying sizes behave if implemented as the start configuration of a cellular automaton simulation. We can define our start configuration more formally by the following:

If we let $J = \{m|C^0_{m,n} = 1\}$, $I = \{n|C^0_{m,n} = 1\}$, $\mu = \sqrt{|J|}$, and $\eta = \sqrt{|I|}$, then the following properties must be true for the start configuration to meet our guidelines (i.e. being a square).

$$\mu \in \mathbb{Z} \text{ and } \eta \in \mathbb{Z} \tag{1}$$

$$\forall a \in J,\ \exists\ r_1, ..., r_\mu \in J : \forall i,\ r_i = a. \text{ The same applies for } I. \tag{2}$$

$$\forall a \in J,\ \exists\ b : b \in J \wedge b \pm 1 = a. \text{ The same applies for } I. \tag{3}$$

Lemma 4.1. *If the above conditions are true for a given start configuration, then the resulting shape at $t = 0$ made by the "live" cells is a square.*

Proof. If (1) is true, we know that the number of live cells on the grid is a perfect number. For a square, this characteristic would certainly hold as its area would be a product of two side lengths, and in the case of an integer-lattice, $\mathbb{Z}^2$ (upon which the square is represented), both side lengths would be integers, and therefore, their products would be integers as well. But (1) can be fulfilled by starting with any perfect number of live cells regardless of the pattern in which they are arranged. Hence, (2) is a necessary condition. If (2) is true, then for every element of J, there exists μ cells which have the same value of m (the same holds for set I with the value of n). Thus, because $|J| = |I|$, the configuration fulfilling conditions (2) and (1) exhibits reflectional symmetry and rotational symmetry. But the first two conditions do not restrict the number of dead cells between the live cells or the spread of the live cells (in a single square, the start configuration needs to be a single aggregate of cells), thus condition (3) is necessary. If (3) is true, then for every live cell at position (m, n) there exists another live cell in the neighborhood of the cell at (m, n). Hence, all the live cells are connected to each other in some manner. If all three conditions are true, then the resulting shape is a single mass of live cells with an equal number of rows and columns. Moreover, it exhibits reflectional and rotational symmetry. Therefore, the resulting shape must be a square. □

Corollary 4.1.1. *If a start configuration fulfills (1),(2), and (3), the cellular automatons configuration at any value of t exhibits rotational and reflectional symmetry.*

Proof. Suppose there is a cell A which has a value of 1 at $t = 0$ and is located at position (m, n). Further, because the collection of cells forms a square, there exists a horizontal and vertical line of symmetry. Let the horizontal line of symmetry be defined as $y = a$ and the vertical line be defined as $x = b$. Then, there exists cells B, C, and D at positions $(b-m, n), (m, n-a)$, and $(b-m, a-n)$, respectively, which all have the same value of A. Moreover, we know that the value of S is the same for cells A, B, C and D. Therefore, the states of both cells will equal each other when $t = t + 1$. Because φ is applied uniformly to each cell, this property holds for all cells at position (m, n). Consequently, it can be concluded that the cellular automaton exhibits the same symmetry for any value of t. □

5 Systematic Methodology

In this section, the definitions established above will be used to develop a method of analyzing a cellular automaton's evolution with mathematical techniques.

Now that the start configuration has been explicitly outlined, we can continue with our original intentions. We define our square as having side length ρ such that ρ is greater than 0. Furthermore, we define the lower left corner of the $\rho \times \rho$ square as being located at position $(0, 0)$. Thus, we can define the cells located along each of the four borders as:

$$\begin{aligned} B_1 &= \{C_{0,1}; C_{0,2}; ...; C_{0,\rho-2}\} \\ B_2 &= \{C_{1,0}; C_{2,0}; ...; C_{\rho-2,0}\} \\ B_3 &= \{C_{\rho-1,1}; C_{\rho-1,2}; ...; C_{\rho-1,\rho-2}\} \\ B_4 &= \{C_{1,\rho-1}; C_{2,\rho-1}; ...; C_{\rho-2,\rho-1}\} \end{aligned}$$

Note that we exclude the four corners of the square from its borders. This is done intentionally because the value of S for the corners is different than the value of S for the other cells along the border (this difference will be important in later calculations). Hence, in the case that $\rho = 1$ or 2, every set $B_1...B_4 = \{\emptyset\}$. Also note that the above definitions only apply to values of ρ greater than 4 (in the case $\rho = 3$, $|B_i| = 1$ while in the case $\rho = 4, |B_i| = 2$ for $i \in \{1, 2, 3, 4\}$). In general, $|B_i| = \rho - 2$.

Using this construction, we can find the configuration of the automaton at $t = 1$. To do this, we first need to separate all the live cells into three groups based on their respective S values. The four corners of the square make up one group. They are located at

$$(0, 0), (0, \rho - 1), (\rho - 1, 0), \text{ and } (\rho - 1, \rho - 1).$$

For each corner located at these points, there are three live cells contained in its neighborhood (one either below or above, one to the side, and one in the diagonal). Thus, $S = 3$ and the value of each corner at $t = 1$ is 1. The cells represented by sets $B_1...B_4$ (located in the border of the square) each have an S value of 5 (one live cell above and below, one to the side, and two along the diagonal of the respective border cell) and hence will equal 0 after the transition rule is applied. The last group consists of all the cells within the borders of the square. For these cells, $S = 8$. Thus, in the next generation they will take on a value of 0. So far we have only considered cells a part of the initial square. To define all of the live cells at $t = 1$, we need to consider the dead cells bordering the square. Namely, we should observe the dead cells in the von Neumann neighborhood (figure 1) of the cells contained in each set $B_1...B_4$. Because these cells will have exactly three live cells in their neighborhoods, they will become live cells in the next generation. Combining these changes, we have the following live cells at $t = 1$:

$$\begin{gathered} C_{0,0}; C_{0,\rho-1}; C_{\rho-1,0}; C_{\rho-1,\rho-1} \\ C_{1,-1}; C_{2,-1}; ...; C_{\rho-2,-1} \\ C_{-1,1}; C_{-1,2}; ...; C_{-1,\rho-2} \\ C_{\rho,1}; C_{\rho,2}; ...; C_{\rho,\rho-2} \\ C_{1,\rho}; C_{2,\rho}; ...; C_{\rho-2,\rho} \end{gathered} \tag{4}$$

To better understand this progression, consider an example where $\rho = 5$. The initial configuration would be identical to the shape depicted in figure 4, if we let the black cells represent those with the value of 1 and the white cells represent those with the value of 0.

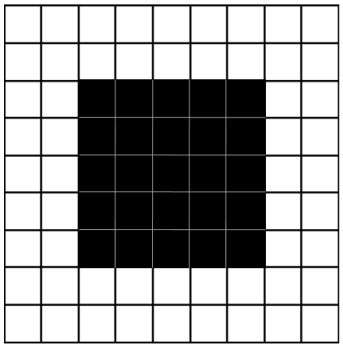


Figure 4: *Initial Configuration.*

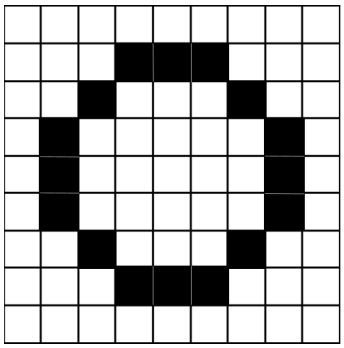


Figure 5: *Configuration at t=1.*

At $t = 1$, the system changes according to the transition rule (figure 5). Notice how the four corners of the square remain "alive" while the borders and the interior cells of the square "die". Also, notice how the cells in the von Neumann neighborhood of the squares border cells take on the value of 1. This generation corresponds with the changes outlined in (4). We can use this method to find changes in its configuration as t gets larger. In doing this, we find that at $t = 11$ the automaton takes on the configuration in Figure 6.

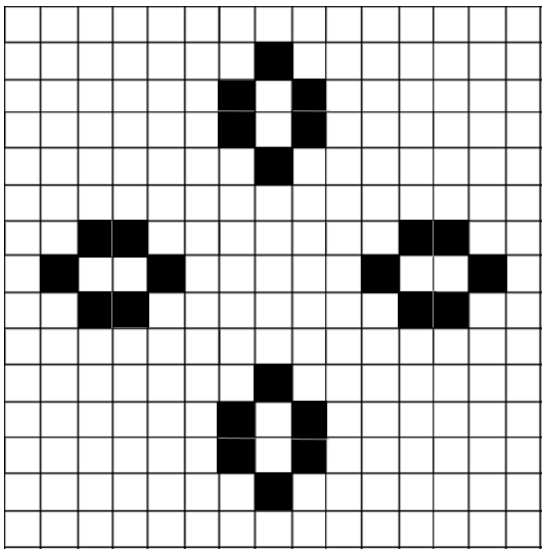


Figure 6: *Configuration at t=11*

These four shapes created by the black cells are known as "beehives". When four beehives are arranged in two groups of two beehives on the same columns with the two groups perpendicular to each other, the resulting shape is known as a "honey farm". Hence, Figure 6 depicts a honey farm (more at Appendix C). At this point, the cellular automaton has reached a still life or a point in its evolution where the configuration either remains static for all subsequent values of t (as with the cellular automaton in Figure 6) or the configuration oscillates between states for all subsequent values of t (as with the blinker shown in figures 2 and 3). Knowing that when $\rho = 5$, the configuration reaches a still life at $t = 11$, we can run a few simple simulations to see how different systems behave at different values of ρ.

6 Simulations and Findings

Using Edwin Martin's Conway's Game of Life program, we simulate each cellular automaton created by a respective value of ρ for all values of ρ over the interval [1, 25]. Our results of this simple exercise are recorded in Table 1 where t represents the value of t at which the cellular automaton reaches a still life.

ρ	t	Final State Description[1]	ρ	t	Final State Description[1]
1	1	nothing	14	9	pond
2	1	nothing	15	22	nothing
3	5	Traffic lights	16	11	4 ponds inside 4 hives
4	4	nothing	17	33	tub in center surrounded by four bi-blocks with 3 blinkers
5	11	honey farm	18	17	pond surrounded by 4 bi-blocks
6	5	Pond	19	20	Traffic lights surrounded by 4 traffic lights
7	5	Traffic lights	20	12	nothing
8	6	nothing	21	26	tub surrounded by 4 traffic lights
9	16	tub surrounded by 4 blinkers	22	13	pond surrounded by 4 blocks
10	17	pond	23	48	Traffic lights surrounded by 8 blocks surrounded by 4 traffic lights
11	32	4 blinkers	24	15	4 ponds surrounded by 4 blocks
12	9	nothing	25	46	tub surrounded by 4 blinkers surrounded by 4 blinkers
13	18	tub			

Table 1: Simulation of cellular automata at different values of ρ

Also, the final state description gives a brief explanation of the shapes that compose the still life. A reference of terminology for the various shapes can be found in Appendix C. From these simulations, we find that each cellular automaton reaches a still life at some value of t. Moreover, it seems that, with a few exceptions, as ρ gets larger so does the value of t. This observation leads to the question *do all systems derived from conditions (1), (2), and (3) eventually terminate or reach a still life?* To answer this question, we need to be able to understand the underlying mechanisms relating ρ to t and then use this understanding to make a general statement about all positive integer values of ρ. First, though, we need more data so as to better grasp this intricate behavior. Because manually inputing start configurations into the simulation program is tedious and time consuming, we create our own programs to iterate through different values of ρ and output the necessary values. The basis for these programs can be found in Appendix A. Running individual simulations for ρ over the interval [1,79], data is collected and documented in tabular form in appendix B. This data is graphed as a scatter plot to represent it visually as depicted in Figure 7. Just looking at the figure there are a few observations that can be made. Most notably, there is an obvious trend in the data as predicted earlier. Furthermore, by observing the dotted trendline, we can see that the t values increase as ρ increases. The strength of this correlation can be calculated by finding the

[1]More at Appendix C

correlation coefficient using the the formula:

$$r = \frac{\sum xy - \frac{(\sum x)(\sum y)}{n}}{\left(\sqrt{\sum y^2 - \frac{(\sum y)^2}{n}}\right)\left(\sqrt{\sum x^2 - \frac{(\sum x)^2}{n}}\right)} \tag{5}$$

Substituting appropriate values, we find that $r = \frac{513242 - 377845.7532}{(200.94)(1124.1925)} \approx 0.599$. Thus, there is a moderate correlation between the values.

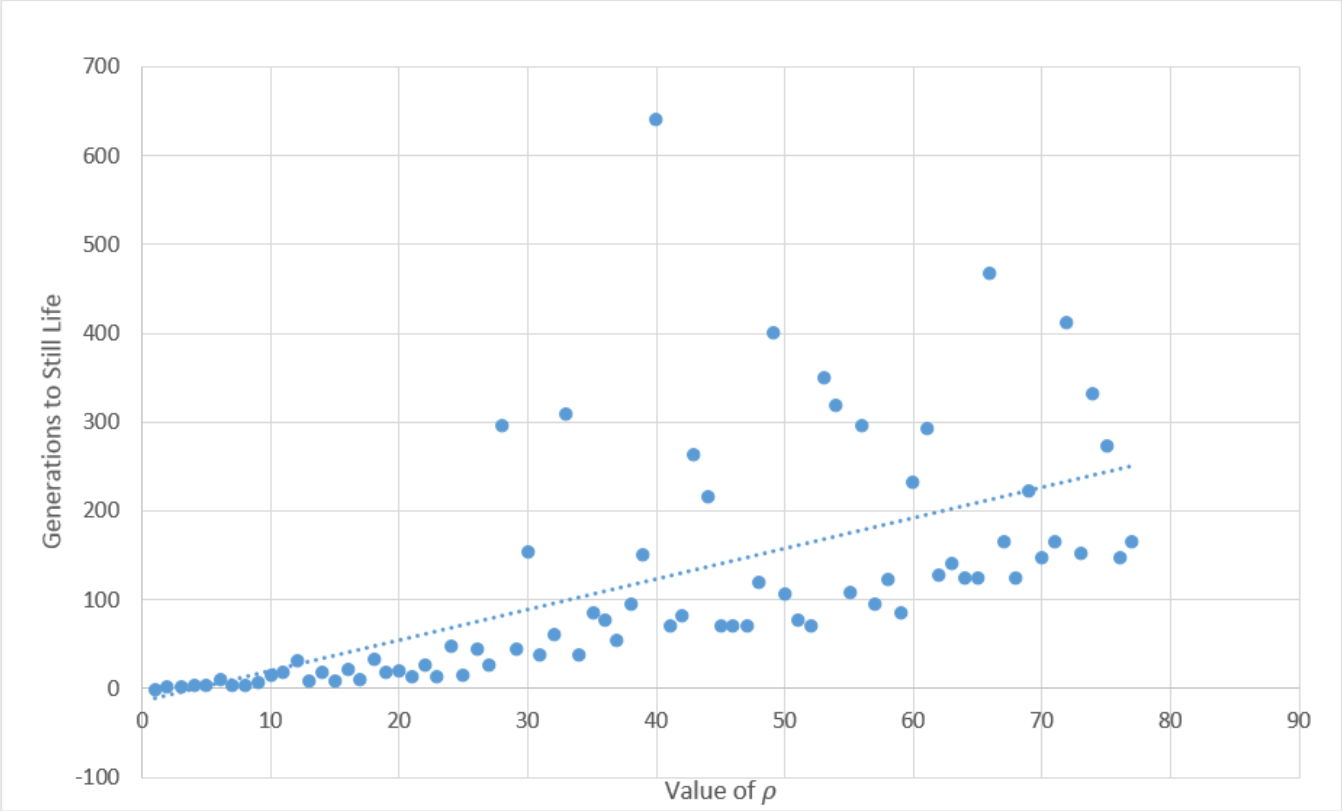


Figure 7: *Scatter plot of data found in Appendix B*

Secondly, several outliers are noticed that seem to appear almost at random. This observation is expected though because of past descriptions of the cellular automata. According to Stephen Wolframs classification of cellular automata systems, there are four categories of behavior each of increasing complexity [6]. The Game of Life has been described as exhibiting Class IV behavior[7] which is characterized by the involvement of "a mixture of order and randomness: localized structures are produced which on their own are fairly simple, but these structures move around and interact with each other in very complicated ways"[6].

Nevertheless, while these observations are interesting and give testament to the complex nature of cellular automata systems in Conway's Game of Life, we are now ready to address the original question. On the graph and in the table (appendix B), two points were omitted: those formed when $\rho = 58, 59$. The reason for this omission was originally a result of what was speculated to be an infinite loop because at these values the simulation program did not terminate, even after more than 12 hours of runtime. To solve this problem, the program was modified to graphically display the evolution at large values of t. In doing so, we were able to identify the reason for the unending simulation. Namely, we found the following for $\rho = 58$ at $t = 227$:

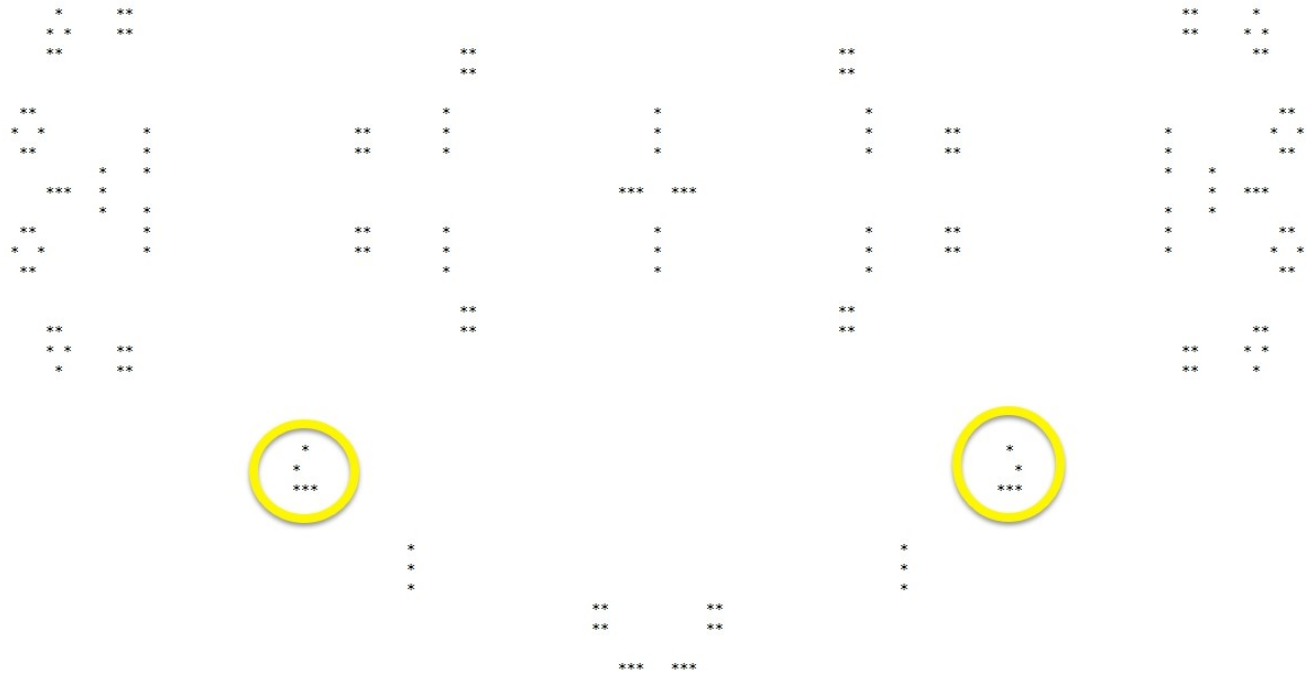


Figure 8: *Configuration at* $t = 227$

Although this picture is a section of the cellular automaton's configuration, it demonstrates the problem. First, note that the asterisks represent a cell with the value of 1 while a white space represents that with a value of 0. Now, notice the circles drawn around two different shapes. These are known as "gliders" which are described in appendix C and are essentially 5 block shapes that perpetually move across the grid in a diagonal direction[7]. Hence, the reason the simulation never came to an end is because these gliders were generated which prevented the system from reaching a still life; the same thing happened when $\rho = 59$. We are now ready to address the original question in the form of a theorem.

Theorem 6.1. *If a cellular automaton is generated following conditions (1), (2), and (3), then the resulting system will eventually terminate at a finite value of* t*.*

Proof. This statement can be disproved by offering a counter example. We have already shown that when $\rho = 58$, the cellular automaton eventually has two gliders that are a part of its configuration. From the corollary proven earlier, we also know that there are two other gliders not shown in the picture (they would be a reflected version the two gliders shown). Furthermore, we know that gliders only move along diagonal lines. These two lines can be defined by the equations $y = x$ and $y = -x$. By looking at the configuration of the cells, it can be observed that the glider on the right is moving along the line $y = -x$ at 315° while the glider on the left is moving along the line $y = x$ at 225°. The only point at which these two lines would intersect is at the point $(0, 0)$ which is located at the center of the automaton. Because these two gliders are beyond this point and heading away from it, they will clearly never intersect. The same applies to the two gliders not shown. One will be moving along the line $y = -x$ at 135° away from the origin while another will be moving along the line $y = x$ at 45°. Thus, neither of the four gliders will intersect. Additionally, it can observed that there are no other cells blocking the path of the gliders, thereby allowing them to continue along their paths for all subsequent values of t. Hence, this cellular automata will never reach a still life because we can take any value of t and show that $t + c$ yields a different configuration for

all integer values of c. Consequently, the theorem is not valid. □

Although the system may not terminate, there is an obvious trend in the data depicted in Figure 7. Namely, it seems that as ρ increases, so does the number of steps needed for the system to reach a still life. Of course, due to the Class IV behavior of the automata, there are outliers that deviate from this trend, but it seems that the majority of cases exhibit this property.

7 Conclusion and Further Work

To formally study the 2-dimensional cellular automata known as Conway's Game of Life, a formal definition of the transition rule had to be provided, initially. In doing so, we were able to establish several conventions which were utilized later in the paper. We then continued to present the main focus of this exploration which was to investigate the evolution of a cellular automaton with a start configuration of a square (a square shape as formed by the cells that start as live cells). We provided three conditions that should be met by a cellular automaton so as to start with the desired configuration. We proved that the only cellular automaton fulfilling the conditions are those with a square start configuration. Additionally, we were able to prove as a corollary that any cellular automaton fulfilling the conditions remains symmetric at any value of t. With these definitions in place, we were able to show mathematically the configuration of a cellular automaton at $t = 1$. We provided an example of this transition and showed how the system eventually reaches a still life. To continue the investigation, we simulated the first 25 values of ρ and found that all reach a still life. After noticing this similarity, we went on to try and prove its universality among all values of ρ. To generate more data, we ran our own simulations and eventually found that our original suspicion was incorrect. We were able to use a counterexample to disprove the original suspicion.

From our work in this paper, we have shown that from simple discrete rules, there can arise intricate, complex behavior. Although this observation has been previously made with other cellular automata (see reference 6), we provided a unique perspective through the analysis of a simple polygon and were able to disprove the regularity of such a structure implemented in the simulation. We were also able to use the defined configurations to show that the cellular automaton exhibit Class IV behavior. For further studies, we could investigate the nature of this class IV behavior by better understanding the cause for the outliers found in Figure 7. With this understanding, we could thereby predict the occurrence of such an abnormality. Other than understanding the Class IV behavior, we could also focus on other aspects of the cellular automaton fulfilling the defined conditions. For example, we could investigate why certain systems spawn a collection of gliders while others do not. Also, we could search for an answer to the question, Are the gliders the only structures preventing the simulation from reaching a still life? Lastly, we could address the question that originally inspired this exploration by analyzing other fundamental shapes and their behavior in the Game of Life. Ultimately, it is through the answering of these questions and problems that we can better grasp the underlying nature of cellular automata and similar discrete models of computation.

References

[1] Toffoli, T., & Margolus, N. (1987). Cellular automata machines: a new environment for modeling. MIT press.

[2] Mitchell, M. (1996). Computation in cellular automata: A selected review.Nonstandard Computation, 95-140.

[3] Solomon, R. (2008). The Little Book of Mathematical Principles, Theories, & Things. Metro Books.

[4] Weisstein, Eric W. Cellular Automaton. 2014. Web site. 4 November 2014.

[5] Sarkar, P. (2000). A brief history of cellular automata. ACM Computing Surveys (CSUR), 32(1), 80-107.

[6] Wolfram, S. (2002). A new kind of science (Vol. 5). Champaign: Wolfram media.

[7] Martnez, G. J., Seck-Tuoh-Mora, J. C., & Zenil, H. (2013). Wolframs classification and computation in cellular automata Classes III and IV. In Irreducibility and Computational Equivalence (pp. 237-259). Springer Berlin Heidelberg.

[8] Elkies, N. (1997). The still-Life density problem and its generalizations.

8 Software Used

Graphs come from Microsoft Excel

Basic Simulations come from Conway's Game of Life, written by Edwin Martin

Figures were made using MS PowerPoint, Skitch Touch, and GIMP

All C++ programs were written in the Dev C++ integrated development environment

Statistical data comes from user created simulator programs. The code for the program concept is included in Appendix A.

This paper was prepared in LaTeX using TexMaker.

A Program Concept

```
#include<vector>
#include<fstream>
#include<iterator>
#include<algorithm>
#include<cmath>
using namespace std;

vector< vector <int> > updateVec(vector< vector <int> > grid);//generates automaton at
    t+1
vector< vector <int> > checkVec(vector< vector <int> > grid);//check to see if the grid
    needs to be expanded. If it does make sure to increase the size of prev. grids as
    well
bool test(vector< vector <int> > grid); //make sure to take into account the increased
    size of vector. THis tests to see if the automaton has reached still life

//create a new automaton (once val_n is updated)
vector< vector <int> > newAutomaton();

void out2file(void);//outputs data to two separate text files

int rows=0;

int num_gens=0;//represents the number of generations of a specific automata
int val_n=300;//n represents the intitial configuration of the nxn square. Namely, it
    specifies the square root of the number of cells turned on

//storage grids to compare with current in order to determine if still life has been
    reached
vector< vector <int> > prev2grid;
vector< vector <int> > prevgrid;



int prelim=0;
int counter=0;
//create a program to generate output representing the number of
//generations it takes for an nxn grid of cells to reach still life
int main()
{

   vector< vector <int> > grid;
   grid=newAutomaton();
   while(!test(grid))
   {
      grid=updateVec(grid);
      grid=checkVec(grid);
   }
```

```
   out2file();
   grid.clear();
   return 0;
}

vector < vector<int> > updateVec(vector< vector <int> > grid)
{
   //Takes grid and applies transition rule to each cell
   //creates new automaton and returns it
}

bool test(vector< vector <int> > grid)
{
   //checks to see if the grid is the same as either of the two storage grids
   //if so, returns true
   //otherwise updates storage grids and returns false

}

void out2file()
{
   //outputs num_gens and val_n to file
   //resets num_gens and updates val_n
}


vector< vector <int> > checkVec(vector< vector <int> > grid)
{
   //checks the grid for any live cells in the outer two layers
   //If test is true, updates the grid with one or two quiescent layers
   return grid;
}


vector< vector<int> > newAutomaton()
{
   //creates a new grid based in val_n
   return grid;
}
```

B Raw Data

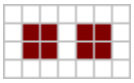
ρ	t
1	1
2	1
3	5
4	4
5	11
6	5
7	5
8	6
9	16
10	17
11	32
12	9
13	18
14	9
15	22
16	11
17	33
18	17
19	20
20	12
21	26
22	13
23	48
24	15
25	46
26	26

ρ	t
27	295
28	45
29	154
30	38
31	62
32	309
33	38
34	87
35	78
36	53
37	96
38	150
39	641
40	69
41	82
42	265
43	216
44	70
45	70
46	70
47	120
48	401
49	107
50	78
51	70

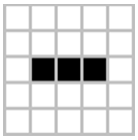
ρ	t
52	351
53	318
54	109
55	297
56	95
57	122
60	85
61	232
62	294
63	127
64	140
65	125
66	124
67	467
68	165
69	124
70	223
71	148
72	165
73	412
74	152
75	332
76	273
77	147
78	165
79	277

C Terminology[1]

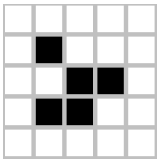
Bi-block: Two blocks next to each other, separated by a pair of dead cells.



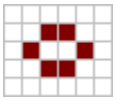
Blinker[2] : Three live cells directly adjacent to each other.



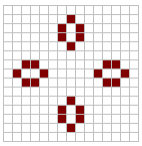
Glider[2]: A 5 block shape that moves across the grid in a diagonal direction.



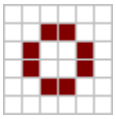
Hive: A 3×4 rectangle of live cells where the interior is made of dead cells and the corners are also made of dead cells.



Honey Farm: Four hives positioned at most 5 dead cells from the other. There are two groups of two hives each and each group is perpendicular to the other.



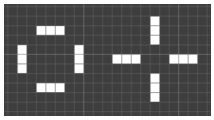
Pond: A 4×4 square of live cells where the interior is made of dead cells and the corners are also made of dead cells.



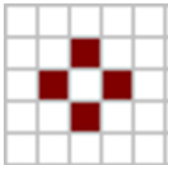
[1] All Pictures come from `http://en.wikipedia.org/wiki/Still_life_%28cellular_automaton%29` unless otherwise indicated

[2] Comes from `http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life`

Traffic Lights[1]: Four blinkers positioned so they are always separated by at least one dead cell.



Tub: The same as a pond but with a 3×3 square of live cells.



[1]Comes from `http://conwaygamelife.wikia.com/wiki/Traffic_Light`