

A Case Study on Cellular Automata

Rafael Enrico C. Hugo

2401 Taft Avenue

Manila, Philippines

rafael_hugo@dlsu.edu.ph

Abstract

The case study focuses on John von Neumann's and Edgar Codd's description of a Cellular automata. It is a Type 0 machine which means that it is similar to a Turing machine that interprets unrestricted grammar. Its input is a two-dimensional data tape filled with cells and its output is an overwritten two-dimensional data tape. It can evaluate all cells at once and transform them simultaneously. It is primarily used to simulate complex systems like entropy, evolution, and self-reproduction.

Machine Definition

Cellular automaton is a mathematical model utilized to design and simulate systems that seemingly have unrecognizable patterns unless observed from a smaller degree or divided into parts. John von Neumann thought of the idea for the model when he was working on biological evolution and self-reproduction. His concept was to create a self-reproducing machine.

He described the model as having the following characteristics:

1. A plane that is divided evenly into squares
2. Each square has a copy of a finite automaton. It is only called a cell if it is one of the squares in the plane and if it also contains a finite automaton.
3. Each cell has a neighborhood, which is itself and the cells above, below, left, and right to itself.
4. A cell's state at time $t + 1$ is determined by its neighborhood state at time t as well as the transition function f of the finite automaton.
5. The finite automaton possesses a distinguished state such that

$$f(v_0, v_0, \dots, v_0) = v_0$$

6. During each time step, all but a finite number of cells are in the distinguished state.
7. There are 29 distinct states for each cell.
8. A transition function f is specified and is also shown to have a certain computation.

Several others expounded on von Neumann's ideas and sought to provide the computation and construction universality of the 29-state cellular space. But it was

Edgar Codd who introduced the idea of 2-state cells; these states were either 1 (on) or 0 (off). Through his observation of the propagation patterns with these cells, it disproved the universality of von Neumann's model. Codd's 2-state cell model will be the central focus of discussion of this case study as it is proven to be a more universal abstraction of simulating complex systems.

To describe Codd's model mathematically, a cellular space must first be defined. A cellular space is a set of integers $I \times I$ on a plane or to give a visualization, it is quite similar to a Cartesian plane. Each cell is connected to n neighboring cells and in all of them, there is an assumption that one of them is a state similar to the state we are evaluating. Codd then described the neighborhood state function as

$$h^t(a) = (v^t(a), v^t(a + \delta_2), v^t(a + \delta_n))$$

It is simply saying that to determine a cell's state, it makes use of the values of other cells. This state function is crucial in determining the state of a cell at time $t + 1$. That said, if a neighborhood state is all off then so is cell being evaluated in the neighborhood. It will remain off forever unless there is a change.

A realization of the machine was also worked on by Codd through von Neumann's description of the self-reproducing machine. The automata makes use of a control unit and a data tape similar to those used in Turing machines. There would have to be a distinction as Turing machines can only interpret one dimension and cellular automata makes use of two dimensions so it is assumed that the Turing machine used is able to evaluate a two-dimension data tape. If the automata was to be used as a Turing machine, then the data or output will be placed on the data tape. But cellular automata is also capable of constructing so if this was taken into consideration and the machine was used as is, the data tape would be useful for storing specifications of the output. Therefore, its total state is the state and the contents of the infinite two-dimensional tape.

A cellular automaton's input is a data tape with the set of states, including the initial state, that extends infinitely

to the right. Stephen Wolfram utilized a simple one-dimensional array to demonstrate the input and output of the machine.

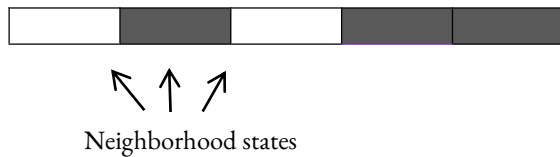


Figure 1. Wolfram's neighborhood states

Since there are three states in a neighborhood, there is a total of 2^3 possible patterns of neighborhood states. A general formula for the states is 2^n where n is the number of states in a neighborhood. Given the example from Figure 1, Wolfram gave the output for this neighborhood as follows:

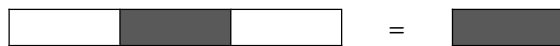


Figure 2. Initial state output

We can then follow onto the array evaluating the next three.

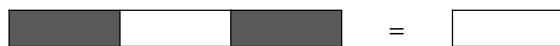


Figure 3. Next state output

Completing the evaluation, we shall get this figure as the output of Figure 1.1:



Figure 4. Output

Its output is also a one-dimensional array similar to its input. A data tape can be compared with every subsequent generation so that the output and pattern can be seen.

As earlier stated, cellular automata has the capability of checking several cells at once with its neighborhood state function. It is then possible to predict the outcome of an input or to get the next generation based on certain rules that the automata follows. Since the automata will continue to evaluate all the neighborhoods, it is possible that the is ever-changing except when it is evaluated at a certain time t . A great example is John Conway's Game of Life. It utilizes Moore's neighborhood of an initial cell and all the cells around it. The game follows these rules:

1. If a cell has two to three neighbors that are alive, it will survive.
2. If a cell has one or no neighbors alive or if it has four or more neighbors alive, it will die.
3. If an empty cell has exactly three living neighbors, then it will become alive.

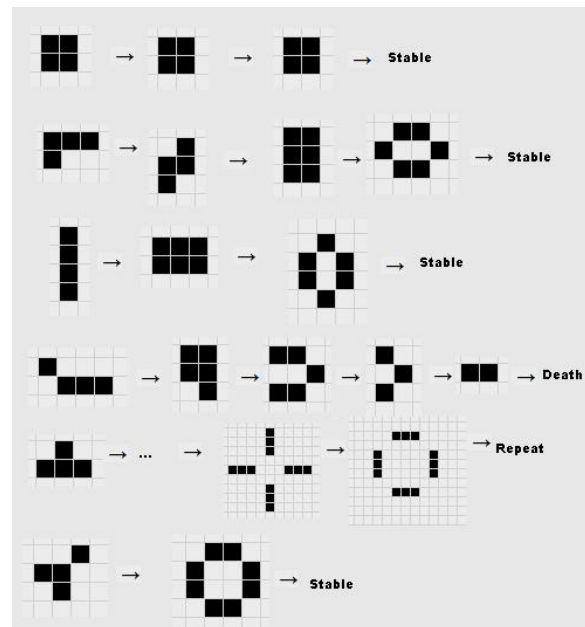
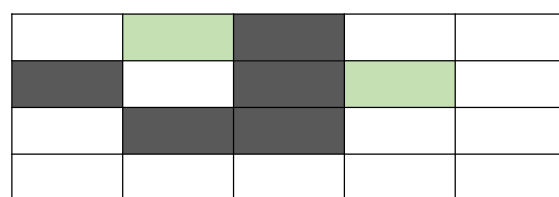
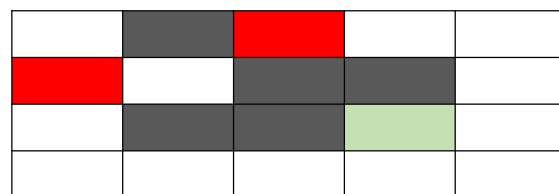


Figure 5. Cornell University's visualization of the Game of Life patterns

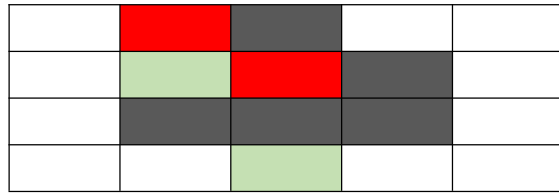
A well-known pattern is the glider which moves across the plane forever as long as it doesn't reach the boundary. Below is a visualization of the repeating pattern that allows the glider to move across the plane.



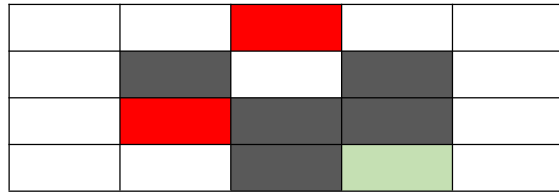
Initial Glider Pattern



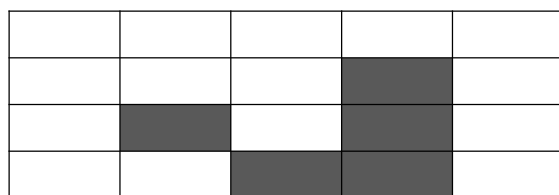
Pattern 2



Pattern 3



Pattern 4



Back to initial glider pattern

Figure 5. Conway's glider pattern in the Game of Life

A rough tracing is realized to see which cells have died (red) and which cells will live (green). The cellular automata is capable of changing the data that it is fed since it evaluates several cells at once and then outputs a similar data tape based on rules. Likewise, in Conway's example, it is capable of drastically changing the cellular plane.

Formal Language and Computational Power

To place cellular automata in Chomsky's hierarchy, it is insufficient to interpret von Neumann's ideas and conclude that it is indeed a Type 0. A more proper way would be to first construct it and then compare it to other machines in each type.

Aristid Lindenmayer provided an explanation to the language of cellular automata. He distinguished that all cells in the array must be transformed at the same time and there is also no distinction between terminal and nonterminal symbols. This would already eliminate Type 3 and Type 2. The former, being Regular Grammar and Finite State Machines, are not able to evaluate the whole array at once. On the other hand, Type 2 utilizes a data stack which also means that it is not able to evaluate the whole array at once. This leaves us with Type 1 and Type 0.

Let's first assume that cellular automata is a Type 1. It is a form of Turing machine that is linear bound although

this type is unable to rewrite or overwrite the data tape it is fed. According to Lindenmayer, all cells in the array or data tape must be transformed at the same time meaning that there is overwriting over the data tape. Therefore, cellular automata cannot be a Type 1.

Although Type 0 is left, it cannot be assumed that cellular automata is indeed a Type 0. It must be first constructed to be evaluated. Through von Neumann's description of the cellular automata, it is known that the automaton is a Turing machine. In reference to what Codd said, this Turing machine is able to overwrite the two-dimensional tape it is being fed. This two-dimensional tape lacks context or is context-free similar to Chomsky's description of Type 0 of unrestricted grammar. It can be concluded that the Turing machine evaluating this two-dimensional tape is a Type 0; cellular automata is a Type 0.

Applications

In 1992, Kai Nagel and Michael Schreckenberg published their paper *A cellular automaton model for freeway traffic* wherein they made use of cellular automata as a foundation for simulating of freeway traffic. This later became known as the Nagel-Schreckenberg model.

With relation to cellular automata, their model is described as a one-dimensional array of L sites wherein these sites can be occupied by a vehicle or left empty. Each of the vehicles also have an integer velocity between 0 and the max value realistically possible. Every state update takes note of four variables: (1) acceleration, (2) slowing down, (3) randomization, and (4) car motion.

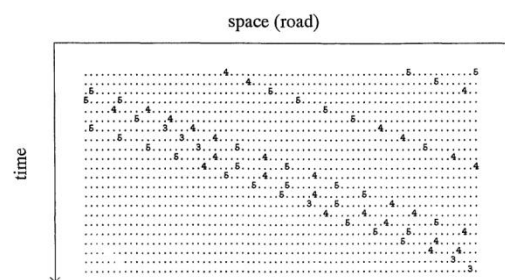


Figure 3.1 Nagel-Schreckenberg's visualization of a closed traffic system

Each horizontal line is a road and each one below it is another road but in a different time $t + 1$. To predict the next state, the previously stated four variables must be taken note of and it must follow these rules:

1. If the velocity v is lower than the max value realistically possible and if the distance of the next vehicle is larger

than $v + 1$, a value of one would be added to the acceleration.

2. If a vehicle a is at a site and another vehicle b is in front of a such that the velocity of a and the its distance d behind b is $a > d$, a will reduce its acceleration by one.

3. There is a random chance p that a vehicle's acceleration is decreased by one.

4. A vehicle is advanced some quantity of sites.

Taking a look at the time t , the first horizontal from the top, there is a vehicle a with acceleration value 4 and in front of it are two vehicles b and c with acceleration value 5. There are 32 sites between a and b and 11 sites between b and c . Following the rules, a will then move 4 sites because the distance between a and b is greater than the acceleration of a . Same goes for b .

There might be confusion as to how cellular automata is implemented in the experiment. Below is another model used by Nagel-Schreckenberg but it has been manipulated to showcase a much more realistic traffic flow.

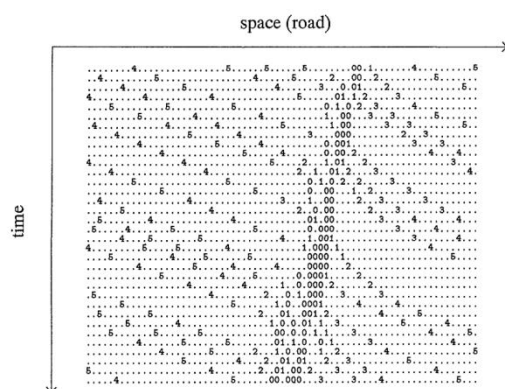


Figure 3.2 Nagel-Schreckenberg's visualization of a manipulated closed traffic system

It is much more cluttered and there also exists vehicles with 0 acceleration. This means that a vehicle has stopped on the road for a certain time t and it would also mean that it would have to accelerate for it to move. For a vehicle to stop, it would have to evaluate the distance of the vehicle in front of it. Likewise, it would also have to evaluate if it wants to accelerate. A vehicle is a cell that looks into the cells around it to determine what its acceleration value should be. Every time frame is another generation of cells.

The German physicists compared this manipulated closed traffic system to real life data and it showed that it 'imitates' the latter's case.

References

[1] Codd, E. (1968). Cellular automata, Academic Press Inc. New York.

[2] Cornell Math Explorer's Club. (n.d.). Conway's Game of Life. pi.math.cornell.edu
<https://pi.math.cornell.edu/~lipa/mec/lesson6.html>

[3] Damerow, J. (2010). John von Neumann's Cellular Automata, *Embryo Project Encyclopedia*, Arizona State University. School of Life Sciences. Center for Biology and Society.
<https://hdl.handle.net/10776/2009>

[4] Lindenmayer, A. (1973). Cellular automata, Formal Languages and Developmental Systems. *Studies in Logic and the Foundation of Mathematics*, 677-691.
doi:10.1016/s0049-237x(09)70394-5

[5] Mitchell, M. (1996). Computation in Cellular Automata : A Selected Review. *Non-Standard Computation*. <https://doi.org/10.1002/3527602968.ch4>

[6] Nagel, K., & Schreckenberg, M. (1992). A cellular automaton model for freeway traffic. *Journal de Physique I*, 2(12), 2221-2229. doi:10.1051/jp1:1992277

[7] Thatcher, J.W. (1964). Universality in the von Neumann cellular model. University of Michigan.
<https://hdl.handle.net/2027.42/7923>


[8] Wolfram, S. (1983). Statistical mechanics of cellular automata, *Reviews of Modern Physics*, 55(3), 601-644.
doi:10.1103/revmodphys.55.601

Appendix A: Record of Contribution

Group members:

P1: Rafael Enrico C. Hugo

Activity	P1
Topic formulation	100.0
Machine definition	100.0
Formal language and Computational Power	100.0
Applications	100.0
Raw Total	400.0
TOTAL	100


Rafael Enrico C. Hugo
March 30, 2023