CELLULAR AUTOMATA

How Cellular Automata Work

# Introduction

Cellular automata form the basis of a simple mathematical model that consists of the following elements:

1. An array of cells, each of which is in a particular state at a given time. A state might be a certain color, numeric value, or other attribute. The set of available states is the same for all cells.
2. A set of rules that determines the next state of each cell when the cells change states. The rules are also the same for all cells.
3. A neighborhood of cells that the rules look at to determine the next state of each cell. For example, with one-dimensional cellular automata the rules might look at the current state of a cell and the states of its two nearest neighbors to determine that cell's next state.
4. Generations. An array of cellular automata all change their states at the same time. Each simultaneous change of state can be thought of as the creation of a new generation of cellular automata. (Biological analogies pervade the study of cellular automata.)

The notion of cellular automata originated in the 1940s with the work of Stanislaw Ulam and John von Neumann on self-reproducing machines. The study of cellular automata has continued in more recent times with such scientists as Stephen Wolfram, John Holland, and Christopher Langton, and is central to the emerging science of complexity.

Two remarkable features of cellular automata are the surprisingly complex patterns that they can exhibit based upon simple rules, and their ability to model many natural biological and physical systems.

Looking closely at elementary cellular automata, the simplest type of one-dimensional cellular automata. It then discusses the most famous example of two-dimensional cellular automata, the Game of Life. After presenting these two specific examples, the paper explains how an infinite variety of different types of cellular automata can be created by varying the basic parameters. Finally, it describes the different categories of cellular automata behavior and explains what is required for cellular automata to exhibit complex, life-like behavior (behavior that has been described as being "on the edge of chaos.")

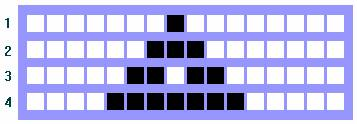
# Elementary Cellular Automata

A grid of elementary cellular automata consists of a one-dimensional row of cells, where each cell can be in one of two states, and the rules for the transformation of a cell are based on the current state of the cell and its two closest neighbors. The neighborhood thus consists of three cells.

The following is an example of a set of rules for elementary cellular automata where the two states are called "white" and "black:"

1. If all three cells are white, the cell remains white.
2. If all three cells are black, the cell becomes white.
3. In any other case (that is, if there is a mixture of black and white cells in the neighborhood), the cell becomes (or remains) black.

The following diagram shows four successive generations of cellular automata conforming to these rules. An initial generation consisting of one black cell with all other cells white, and three subsequent generations created by three applications of the rules:



This example illustrates three basic properties of cellular automata described by Rennard:

* Parallelism: The cells all change state independently and simultaneously (that is, in parallel).
* Locality: The new state of a cell is dependent upon only the state of the cell itself and the states of its neighbors.
* Homogeneity: All cells have an identical set of possible states and follow the same rules.

# II (A). Implementing Elementary Cellular Automata on a Computer

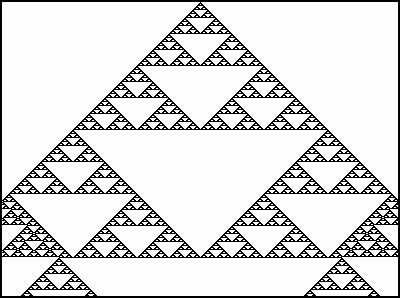
Cellular automata themselves constitute an abstract mathematical model. This model is realized in many natural systems and can also be artificially implemented with pen and paper, colored wooden blocks, and so on. Of course, cellular automata are most commonly implemented on a computer.

A computer implementation of elementary cellular automata could simply display a single row of cells and have each new generation replace the old. More commonly, however, a computer displays each successive generation immediately under the prior generation. Because the cellular automata themselves have only one dimension, we can use the second dimension to represent time, thus revealing the entire evolution of the cellular automata from generation to generation, and also creating complex and interesting patterns. When the new generations reach the bottom of the viewing window, the window contents can be scrolled so that the process can continue indefinitely.

In a computer simulation, it is useful to have a separate array for constructing each generation of cells, which can be displayed on the screen when complete. (The problem with attempting to update a single array is that this method would change cell states that are still needed to determine the new states of other cells.)

Another implementation detail is that the cells on either end of the row, which are each missing one neighbor, can be handled by assuming that the two ends are connected to each other, conceptually forming a circle rather than a simple line. An alternative solution is to assume that the missing neighbors always have a certain state. For instance, in the example given in the previous section, we might assume that the missing neighbors always have the "white" state.

The figure below shows the output of a Java applet written by Eck that implements the example elementary cellular automata described in the previous section. The program automatically produces new generations at regular intervals, and displays each new generation below the previous one, scrolling the output once it reaches the bottom of the applet window. It uses the option of conceptually joining the two ends of the row when applying rules to the cells at the left and right ends.



# The Game Of Life

Elementary cellular automata, including the example that was given, are arranged in a single dimension. This section describes the Game of Life, also known simply as "Life," which is probably the best-known example of two-dimensional cellular automata. The Game of Life was developed by John Horton Conway in 1970 and soon left the laboratory to become a popular mathematical game.

The Game of Life cellular automata are arranged in a simple two-dimensional grid of cells representing a universe or world. Cells can be in one of two states: "alive" or "dead." The following is a universe consisting of only 3 rows and 5 columns in which cells 11, 12, and 13 are alive and the rest are dead:

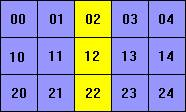


In the Game of Life, a cell's neighborhood consists of 9 cells: the cell itself plus its 8 closest neighbors. Thus, the neighborhood of cell 12 would be cells 01, 02, 03, 11, 12, 13, 21, 22, and 23.

The rules for each cell are as follows:

1. If the cell is dead but is surrounded by 3 alive neighbors, it becomes alive (that is, it is "born").
2. If the cell is alive and is surrounded by 2 or 3 alive neighbors, it remains alive (that is, it "survives")
3. In all other cases, the cell dies or remains dead.

Following these rules, the next generation for the example universe would be as shown here:



These rules are a metaphor for biological life: Each cell is a potential location for a life form. For a life form to be born in a cell requires the coming together of several life forms (in the Game of Life a cell has 3 parents). If a life form is surrounded by too many neighbors (4 or more) it dies (due to "overcrowding"), or if it is surrounded by too few neighbors (0 or 1) it also dies (due to "loneliness").

Certain patterns reappear frequently in the Game of Life. Many of these patterns have been named and studied extensively. For example, a pattern that remains the same from one generation to the next is termed a still life, and a pattern that cycles through a set of configurations is known as an oscillator.

Also mimicking real life, some entire patterns in the Game of Life can self-reproduce (in addition to the reproduction of individual cells). One of best known self-reproducing patterns is termed a glider, which is a five-cell pattern that self-reproduces every four generations, one cell away. A glider gun is a set of cells that generates gliders.

In addition to the obvious beauty and fascination of the patterns generated by the Game of Life, this particular form of cellular automata has theoretically interesting properties. For example, the Game of Life has been shown to computationally universal, as explained in "5. Behavior of Cellular Automata". In the Game of Life's role as a universal computer, a glider can serve as a medium of communication that is analogous to the electrical pulses in a conventional computer.

The Game of Life can be run at the following web site: <http://www.ibiblio.org/lifepatterns/> (click the "Enjoy Life" button). The user can create a custom initial configuration or use a predefined initial state, such as one that demonstrates glider guns. Also, a Game of Life program (Life32 for Windows, written by Johan Bontes) can be downloaded from <http://psoup.math.wisc.edu/Life32.html>.

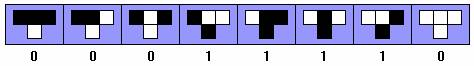
# Other Types of Cellular Automata

The cellular automata described so far in this paper are relatively simple ones in one or two dimensions. It is possible to create an infinite variety of different cellular automata types by varying the following parameters:

* The number of dimensions and arrangement of the cells. The cells in a grid of cellular automata can exist in one, two, three or more dimensions, and the cells can be arranged in many different ways. For example, in 2-dimensional cellular automata the cells can be squares arranged in rows and columns, hexagons packed in a hexagonal mesh, or triangles in a triangular mesh.
* The neighborhood of cells that the rules look at in determining the next state of each cell. The neighborhood usually includes the current cell, but does not need to.
* The number of states that a cell can have. Typically, a cell can be in one of an integral number of discrete states. Alternatively, however, there can be a continuous range of possible state values.
* The rules governing the transition from one state to the next state.

If the cellular automata have K possible states and each cell has a neighborhood of N cells that it depends upon for its next state, then there must be KN different rules to fully specify the cellular automata. This value obviously grows rapidly as the cellular automata become more complex than the simple examples that have been described.

Because elementary cellular automata have 2 states and a neighborhood of 3 cells, they require a total of 23 = 8 rules. (In the example elementary cellular automata described in "2. Elementary Cellular Automata", there are in fact 8 rules; 6 of these rules are lumped together in the "catch-all" rule at the end.) Since each of the 8 rules can specify one of 2 different states, there are 28 = 256 possible types of elementary cellular automata. Each of these possible cellular automata types can be specified by a unique binary number. For example, the following set of rules has the binary number 00011110:



Wolfram refers to the elementary cellular automata determined by these particular rules as "Rule 30," since 30 is the decimal equivalent of the unique binary number.

Changing parameters permits the creation of cellular automata of great complexity. However, complex cellular automata are hard to implement on a computer and are difficult to analyze. Also, there may be little need to create complex cellular automata, since very simple ones can generate patterns of surprising complexity. In fact, possibly the most interesting feature of cellular automata is their demonstration of the emergence of highly complex patterns from very simple objects.

# Behavior of Cellular Automata

Christopher Langton, a founder of the field of artificial life, studied cellular automata in order to model the evolution of life. Accordingly, he classified one cellular automata state as "dead" and all other states as "alive," and also restricted the rules so that if a cell and all its neighbors are dead, then that cell remains dead in the next generation . (In other words, for a cell to be alive, either it must have survived from a previous generation, or it must have been "born" due to the presence of a live neighbor, following Pasteur's dictum that life comes only from life.) Although for simplicity most of the following analysis was done using one-dimensional cellular automata, the categorizations apply to 2-dimensional cellular automata as well.

Langton observed his cellular automata falling into three basic categories:

1. Highly ordered. Cellular automata in this category quickly die out or exhibit only simple, repeating patterns. These cellular automata are stagnant, predictable, and generally considered boring.In prior studies, Wolfram placed these cellular automata in what he terms Class 1 and Class 2, which are the first two of his four cellular automata classes. Class 1 cellular automata reach a homogeneous state, and Class 2 cellular automata soon display only stable or periodic patterns.
2. Chaotic. These cellular automata exhibit few detectable patterns. They are completely random and unpredictable and therefore also boring. Wolfram placed these cellular automata in Class 3, which consists of cellular automata that soon display only chaotic patterns.
3. Complex—on the edge of chaos. These cellular automata exhibit complex patterns and interesting, often lifelike, behavior. Christopher Langton described them as being "on the edge of chaos," because they exhibit behavior that is a hybrid between ordered and chaotic.Cellular automata in this category belong to Wolfram's Class 4, which consists of cellular automata that continue to display complex and often long-lived structures. According to Wolfram, some of the cellular automata in this category also have the theoretically interesting (and possibly practical) property of computational universality, meaning that given a proper initial configuration they are in theory capable of executing arbitrary algorithms. The property of computational universality is normally associated with a general-purpose computer. However, Wolfram explains that a grid of cellular automata can be regarded as a computer: The initial states of the cells are analogous to the initial data stored in a computer memory as a sequence of 0's and 1's. And, as cellular automata evolve from one generation to the next, the states of cells are changed according to deterministic rules built into the cells, just as the initial data in a computer memory is modified according to the rules of the computer architecture and program. A well-known implementation of cellular automata in the third category, which possesses the property of computational universality, is the Game of Life.

How can we predict which of the three behavior categories a given set of cellular automata will demonstrate? In answer to this question, Langton defined a parameter that he named lambda, which refers to the probability that a given cell will be alive in the next generation. (Lambda equals the fraction of rules that lead to the alive state.) As the rules are gradually adjusted so that lambda changes from 0 to 1, the cellular automata transform from category 1 (highly ordered), to category 3 (complex), to category 2 (chaotic). The specific value of lambda that results in complex behavior depends upon the particular set of rules and how these rules are changed to adjust lambda. However, according to Rennard, for many cellular automata lambda values around .3 to .5 tend to produce complex behavior. Eck has written an applet that displays one-dimensional cellular automata and allows the user to interactively adjust lambda to discover the value that creates complex behavior. This applet is located at <http://math.hws.edu/xJava/CA/EdgeOfChaosCA1.html>.

# Conclusion

Cellular automata form the basis of a simple mathematical model that involves of a grid of cells, each of which changes state according to a common set of rules that is "built into" each cell. To determine the next state of a given cell, the rules look only at the current states of the cells in the immediate neighborhood of that cell. The simplest one-dimensional cellular automata are known as elementary cellular automata, and the most popular example of two-dimensional cellular automata is the Game of Life.

By adjusting the parameters, it is possible to generate cellular automata of great complexity. However, it is usually unnecessary to do so, because by appropriately adjusting the lambda value of the rules, very simple cellular automata can exhibit highly complex patterns and behavior, and—for some cellular automata—can even perform universal computation.

The ability of cellular automata to exhibit complex patterns and behavior from simple local rules makes them theoretically interesting. However, as explained in the next paper, they are of more than theoretical interest due largely to their ability to model many natural systems—specifically, those natural systems that also exhibit complex behavior based on simple local interactions, including, according to the field of artificial life, biological life itself.

# Reference List

1. <http://math.hws.edu/xJava/CA/>.
2. <http://www.rennard.org/alife/english/acgb.html>.
3. <http://mathworld.wolfram.com/CellularAutomaton.html>. <http://en.wikipedia.org/wiki/Cellular_automaton>
4. <http://www.mirekw.com/ca/whatis_life.html#How> does it work.