# Chapter 4

# Two-Dimensional Automata

*The chessboard is the world, the pieces are the phenomena of the universe, the rules of the game are what we call the laws of Nature.*

Thomas Henry Huxley

Two-dimensional cellular automata exhibit some of the same characteristics as do one-dimensional automata. There are two fundamental types of neighborhood that are mainly considered. First there is the *von Neumann* neighborhood (the 5-cell version of which was used in the construction of his self-replicating machine), consisting of the 4 or 5 cell array depending on whether or not the central cell is counted:

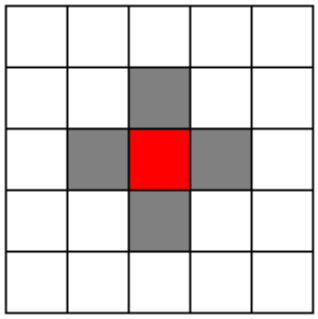

Figure 4.1: The von Neumann neighborhood surrounding a central cell.

The *Moore* neighborhood consists of the 8 or 9 cell array depending on whether or not the central cell is counted (Figure 4.2).

In both cases $r = 1$ and each is useful depending on the context. The extended Moore neighborhood has the same form as the preceding but with $r > 1$.

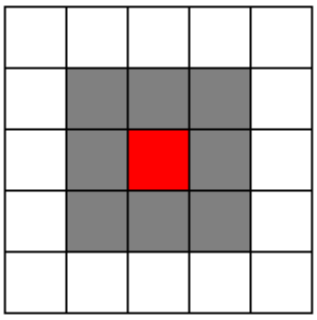

Figure 4.2: The Moore neighborhood with $r = 1$.

Typically, in a rectangular array, a neighborhood is enumerated as in the von Neumann neighborhood illustrated below (Figure 4.3) and analogously for the Moore neighborhood. The state of the $(i, j)th$ cell is denoted by $c_{i,j}$.

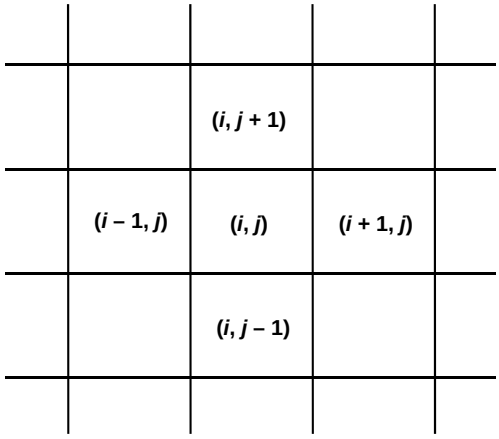


Figure 4.3: The enumeration of the cells of the von Neumann neighborhood.

In the 1-dimensional case with $k = 2$, $r = 1$, there were just $2^3 = 8$ possible neighborhood-states. Now however, with just two states 0 and 1 and a 9-cell Moore neighborhood (again, $k = 2$, $r = 1$), there are $2^9 = 512$ possible neighborhood-states ranging from all white to all black with all the various 510 other combinations of white and black cells in between. A given transition function would tell us how the central cell of each of the 512 neighborhood-states should change at the next time step. How many such transition functions are there? A staggering $2^{512} \approx 10^{154}$, more than the number of all the atoms in the universe! Even with a 5-cell neighborhood,

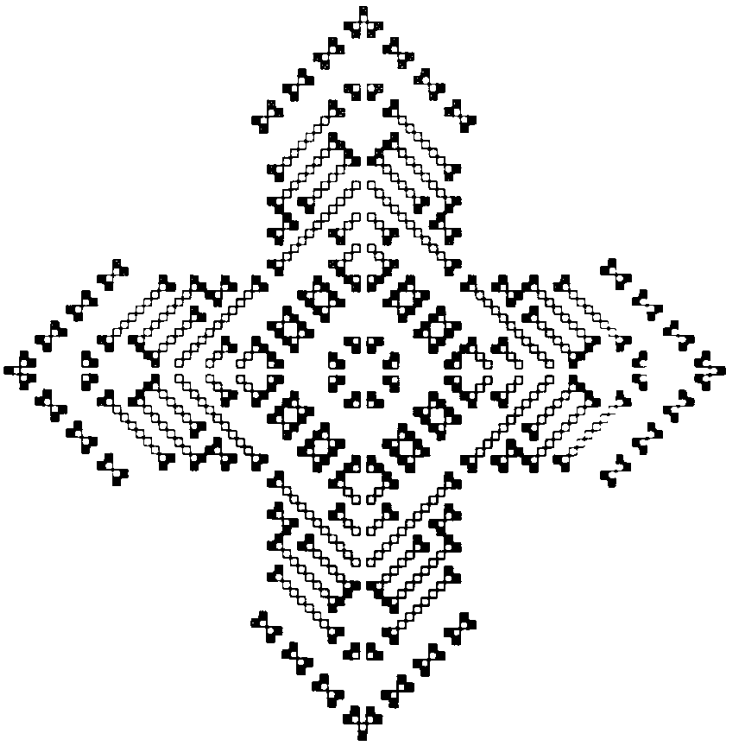

Figure 4.4: The 'dog bone' configuration as referred to by Schrandt and Ulam in which a cell becomes alive in the next generation if it it has exactly one alive (black) neighbor in the current generation and any cell of the previous generation dies.

there are still $2^{32} \approx$ ten billion possible transition functions to choose from.

Beginning in the early 1960s, Stanislaw Ulam and co-workers J. Holladay and Robert Schrandt at Los Alamos Scientific Laboratory began using computing machines to investigate various two-dimensional cellular automata. An infinite plane was considered and divided up into identical squares. The transition rules were eclectic and the results were mostly empirical. One particular automaton was as follows: A cell became alive at the next generation if it was a neighbor (in the von Neumann sense) of exactly one live cell of the current generation. This rule was later coupled with a 'death rule' that required all cells that were a fixed number ($m$) of generations old to die. Say if $m = 2$, then the $(n+1)st$ generation is derived from the $nth$ generation and the $(n-1)st$ generation is erased. Thus only the last two generations survive in any configuration (See Figure 4.4).

We can use the same transition rule together with the death rule, and two different initial configurations on the same array, say in different colors

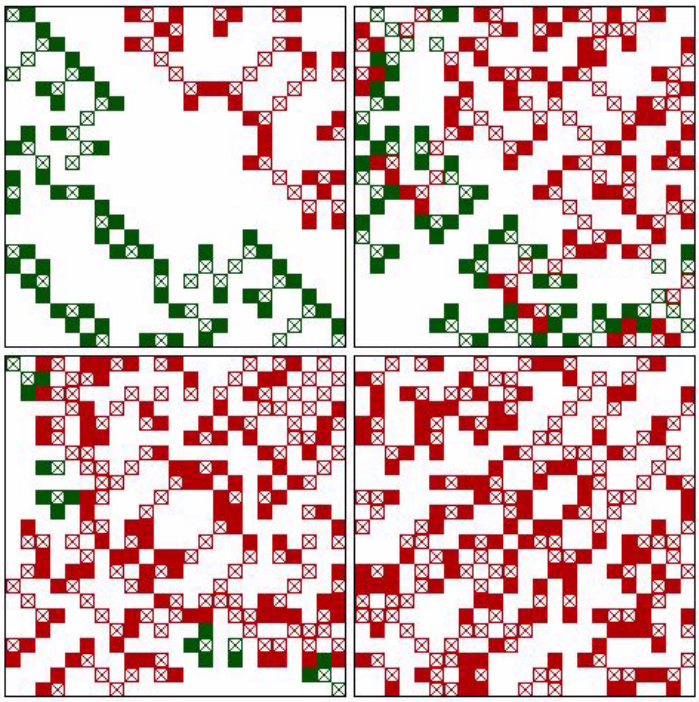

Figure 4.5: A contest on a finite array in which one system (red) on the top right of (a) gradually eliminates the system on the bottom left (green) of (a). Solid cells represent the current generation, crosses the previous generation. The figures represent the situation at generations: 11, 32, 49, and 77 respectively.

to distinguish their respective live cells. The growth rule prohibits the appearance of a live cell at the next generation that has two live neighbors at the present generation, and this condition now includes neighbors of either color. This leads to contests between the two opposing systems with the result that one or both may end up eliminated (Figure 4.5).

## 4.1 The Game of Life

*It is probable, given a large enough "Life" space, initially in a random state, that after a long time, intelligent self-reproducing animals will emerge and populate some parts of the space.*

John Conway

The Game of Life first entered the world stage from the pages of Martin Gardner's *Mathematical Games* column in the October 1970 issue of *Scientific American*. The creator of the Game of Life was the English mathematician, John Horton Conway. Originally, Life was to be played out using counters on a chess or Go board, but soon computer screens became Life's natural domain. The Game of Life caused an international sensation following its rather dramatic creation (R.K. Guy [1985]):

*... only after the rejection of many patterns, triangular and hexagonal lattices as well as square ones, and of many other laws of birth and death, including the introduction of two and even three sexes. Acres of squared paper were covered, and he and his admiring entourage of graduate students shuffled poker chips, foreign coins, cowrie shells, Go stones or whatever came to hand, until there was a viable balance between life and death.*

The rules for the Game of Life are quite simple as each cell has exactly two states (1 - alive, or 0 - dead) and the 8-cell Moore neighborhood is the one considered to determine the state of the central cell:

• A dead cell becomes alive at the next generation if exactly 3 of its 8 neighbors are alive;

• A live cell at the next generation remains alive if either 2 or 3 of its 8 neighbors is alive but otherwise it dies.

In anthropomorphic terms, the second rule says that if a cell is alive but only one if its neighbors is also alive, then the first cell will die of loneliness. On the other hand, if more than three of a cell's neighbors are also alive, then the cell will die of overcrowding. By the first rule, live cells are born from a *ménage á trois*. Of course this is not really a game that you play in the conventional sense but rather a microcosm of another universe that one can explore since we know its physics entirely. An excellent exposition of the Game of Life can be found in the monograph *The Recursive Universe* by William Poundstone [1985].

There are many variants of Conway's original set of rules, as well as using other lattices such as triangular or hexagonal ones or considering Life in other dimensions, but none seem to offer the richness and diversity of the original game. It is common to let the Game of Life evolve on a lattice with periodic boundary conditions ('wrap'), that is to say, cells on the extreme left and right are considered neighbors, and cells at the extreme top and bottom of the lattice are also to be considered neighbors. Whereas, using periodic boundary conditions in the one-dimensional case resulted in a continuous loop, in the two-dimensional case our lattice becomes a torus (donut shaped).

The Game of Life is an example of a Class IV automaton. We note that Packard and Wolfram [1985] found no two-dimensional Class IV cellular automata other than "trivial variants of Life". When the updating of cells is asynchronous, the Game of Life no longer exhibits Class IV behavior but instead converges to a stationary state (Bersini & Detour [1994]).

Life is also an example of an 'outer totalistic' rule in that the state of the central cell at the next time step depends on the prior state of the central cell as well as the sum of the values of the 8 neighboring cells. The preceding rules for the Game of Life can be represented in the following transition table:

| | | | | s | u | m | | | |
|---|---|---|---|---|---|---|---|---|---|
| $c_i(t)$ | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

where the row that one considers for the value of $c_{i+1}(t)$ is given by the value of the central cell state $c_i(t)$.

Conway had considered the rules very carefully so that Life is carefully poised between having patterns that die out quickly and those that continue to grow. In fact, when Life was first proposed, Conway conjectured that no initial collection of live cells could grow without bound and offered a $50 prize to the first person who could prove or disprove this before the end of 1970. The conjecture was short lived as we will see below.

### 4.1.1 Lifeforms

There is an entire pantheon of Lifeforms and the interested reader can refer to the website: http://pentadecathlon.com/index.shtml. We will discuss the evolution of just some of the more common ones. Clearly any fewer than three cells will die in one generation. For a triplet of live cells (that do not vanish after the first generation), they either rapidly evolve to extinction (top three rows of Figure 4.6), become a block of static cells that remains unchanged with all subsequent generations (4th row), or become an oscillating 2-cycle triplet (bottom row):

Four-cell configurations evolve to stable forms (top four rows of Figure 4.7) as well as a long sequence of various forms.

The 5-cell 'R-pentomino' (see Figure 4.12 right) is one of the most fascinating elementary Lifeforms in that its evolutionary history does not stabilize until 1103 generations. In the process it generates what are known as 'gliders', (discussed below) among other Lifeforms.

### 4.1.2 Invariant Forms

Some configurations in the Game of Life remain unchanged at every time step (Figure 4.8). We have already seen the block of four cells and what is known as the 'beehive' (the terminal figure of rows 2,3, and 4 of Figure 4.7).

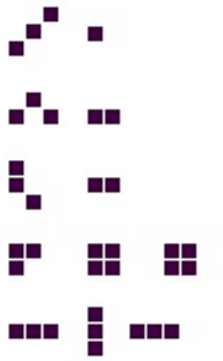

Figure 4.6: The possible evolutionary histories of three cells in the Game of Life. The orientation of the initial cells is not relevant.

Figure 4.7: The evolution of 4 live cells with time increasing to the right. The last two configurations of the last row alternate in a two-cycle.

The 'block' is the most common and it turns up frequently in Life. As each cell of the block has exactly three live neighbors, they all persist in time yet surrounding dead cells have only two live neighbors which is insufficient for them to spring into life.

### 4.1.3 Oscillators

There are some Lifeforms that exhibit periodic behavior oscillating indefinitely between a fixed number of configurations. Period-2 oscillators alternate between two distinct states; they arise spontaneously and are very common; see Figure 4.9.

Many other cyclic oscillators have been created artificially, including periods 3,4,5,... 18, plus a variety of others, including one of period 144 by

Figure 4.8: Some common invariant forms (l to r): block, tub, beehive, ship, snake, pond, fishhook or eater, loaf.

Figure 4.9: Period-2 oscillators. The two rows indicate the two different forms of each oscillator.

Achim Flammenkamp.

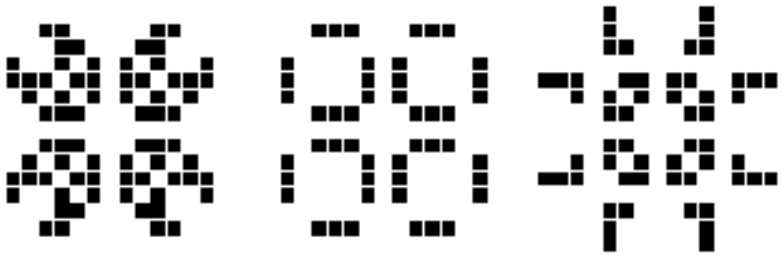

Figure 4.10: An elaborate period 3 oscillator known as the CP-pulsar displaying its three different states. The final state returns to the first at the next time step.

### 4.1.4 Methuselah Configuations

Such initial patterns have less than 10 live starting cells but they continue to evolve to considerable old age before stabilizing and necessarily exclude configurations that grow forever in the sense of an ever increasing number of alive cells. An R-pentomino (Figure 4.12 right) remains alive for 1103 generations having produced six gliders that march off to infinity. The acorn (center) was discovered by Charles Corderman and remains alive for 5,206

Figure 4.11: A period 5 oscillator displaying its five different Lifeforms.

generations. Rabbits were discovered by Andrew Trevorrow in 1986 and stabilize after 17,331 into an oscillating 2-cycle having produced 39 gliders.

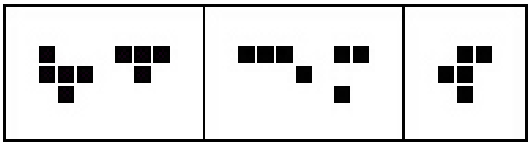

Figure 4.12: Methuselah configurations (l to r): rabbits, acorn, and R-pentomino.

### 4.1.5 Gliders

Gliders are another 5-cell configuration and they actually move one cell diagonally at the fourth time step (Figure 4.13). They are known as gliders as at time step $t+2$, they are reflected in a diagonal line, mathematically, a "glide reflection". By time step $t+4$ the glider is reflected once again back to its original orientation, but one cell (diagonally) displaced. This 4-cycle is then endlessly repeated. The glider is a marvelous creature to watch as it marches in its ungainly fashion across the computer screen. In the words of computer scientist Steve Grand, "The glider is a thing – a coherent persistent phenomenon that moves across 'space' – and yet is not separate from or superimposed on that space. It is simply a self-propagating disturbance in the space created by these little rule-following [cells]" (*Creation*, p. 40).

Figure 4.13: A glider moves one cell diagonally to the right after four generations.

The maximum speed that information can propagate from one cell to

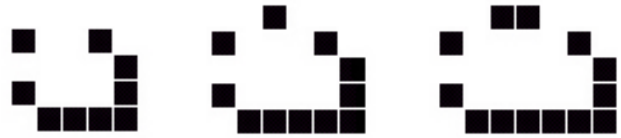

Figure 4.14: From left to right: light-weight, medium-weight, and heavy-weight spaceships. These move horizontally at the speed $c/2$.

another (either horizontally, vertically, or diagonally) is one cell per generation. In Life, this is known as the *speed of light* ($c$). Since the glider moves exactly one diagonal cell after four generations, it is said to move at one-fourth the speed of light ($c/4$).

Conway has proved that the maximum speed of any configuration moving horizontally or vertically is $c/2$. Configurations that actually do move at this speed are what Conway called 'spaceships', depicted in Figure 4.14.

One of the most remarkable configurations to arise in the early days of Life, was the 'glider gun'. This arose out of work done by Robert April, Michael Beeler, R. William Gosper Jr., Richard Howell, Richard Schroeppel and Michael Speciner who were in the Artificial Intelligence Project at M.I.T. In November, 1970 they claimed the $50 prize offered by Conway by demonstrating the glider gun (Figure 4.15) found by Gosper that would indefinitely generate gliders every 30 generations, thus disproving Conway's conjecture that the number of live cells cannot grow without bound. Somewhat remarkably, Gosper's group found that the collision of 13 specially arranged gliders can create their glider gun. Since then glider guns have been found with other periods of generation, even one having period 256.

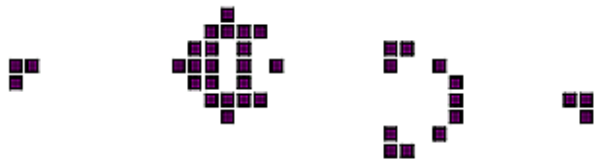

Figure 4.15: The initial configuration of the original glider gun discovered by Bill Gosper that generates a new glider every 30 generations.

Another way to produce unbounded growth in Life is via a 'puffer train'. These objects travel in a vertical direction and leave behind 'smoke' or 'debris' that becomes stable. The first puffer train was also found by Bill

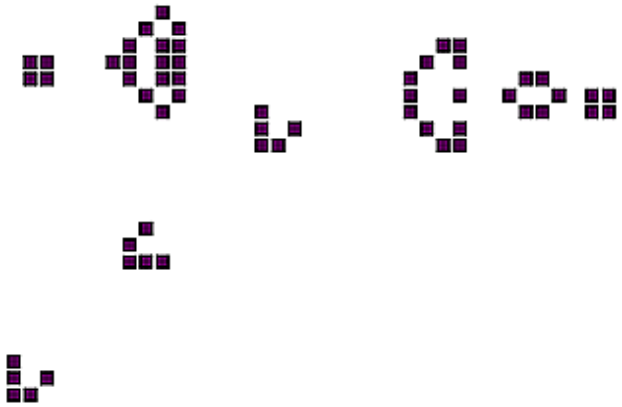

Figure 4.16: The glider gun after it has fired off three gliders toward the lower left.

Gosper and consisted of an engine escorted by two lightweight spaceships. Since then numerous other ones have been discovered (Figure 4.17).

There are also 'glider eaters' that devour gliders and are useful in the creation of logic gates (Figure 4.18).

Another type of unbounded growth was discovered by David Bell in 1993. Starting with a finite configuration of live cells, it produces a 'spacefiller' which is a feature that can appear in other cellular automata (Figure 4.19).

### 4.1.6 Garden of Eden

The evolution of a cellular automaton is governed by the local transition function which alters the states of all cells in the array synchronously at discrete time steps. Thus, patterns of cells are changed into other patterns of cells as the system evolves. It is natural to ask if there are some patterns of cells that do not arise at all in the evolution of the system?

A cellular automaton configuration that can have no prior configuration generating it (via the underlying local transition function) is called a *Garden of Eden* pattern, a term due to John W. Tukey of Princeton University. In a paper in 1962, Edward F. Moore found that in the evolution of a cellular automaton, if a particular configuration had more than one distinct predecessor, then there would have to exist some configuration that would have no predecessor (the Garden of Eden pattern). This was only an 'existence theorem' and no method was given for actually finding the Garden of Eden configuration. The converse to this result was established by John Myhill in 1963, namely, if there is a Garden of Eden configuration with no predecessor, then some configuration must have two distinct predecessors. Both results

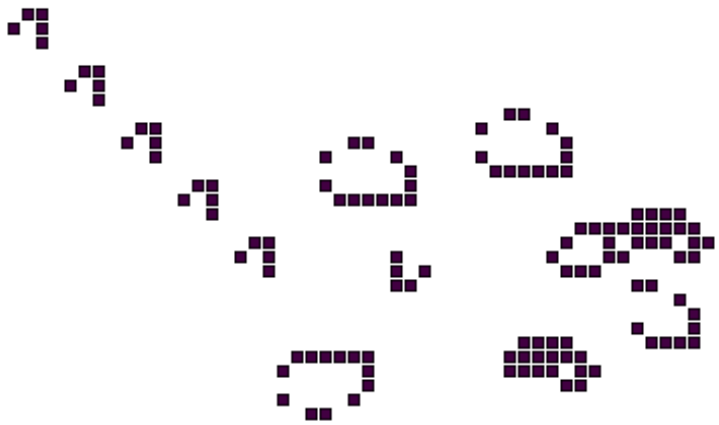

Figure 4.17: A period 16 puffer train (at right) that produces a smoke trail. Based on a design of Tim Coe.

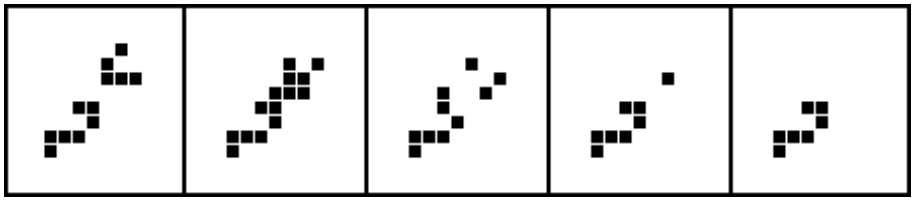

Figure 4.18: In this sequence, a glider-eater in bottom left of the first frame is confronted by a glider approaching at 45 degrees. The glider is gradually eaten until it has disappeared in the last frame. The glider-eater is an essential component of certain logic gates.

are quite general, applying to any cellular automata of finite configurations (i.e. those configurations with only a finite number of non-quiescent cells – although the configurations are infinite in number) and in any dimension. It is clear that if the array consists of a *finite number* of cells, say $N$, then the total number of possible configurations allowing for two possible states per cell is just $2^N$, with at most $2^N$ possible outcomes resulting from the transition function acting on each of these. If it so happens that two different configurations are transformed by the action of the transition function onto the same configuration, then only $2^N - 1$ configurations have been generated by the transition function, leaving one configuration without a precedessor. But the Moore and Myhill theorems are about the infinitely many finite configurations of cellular automata within infinite arrays.

Because the Game of Life does have configurations that have more than

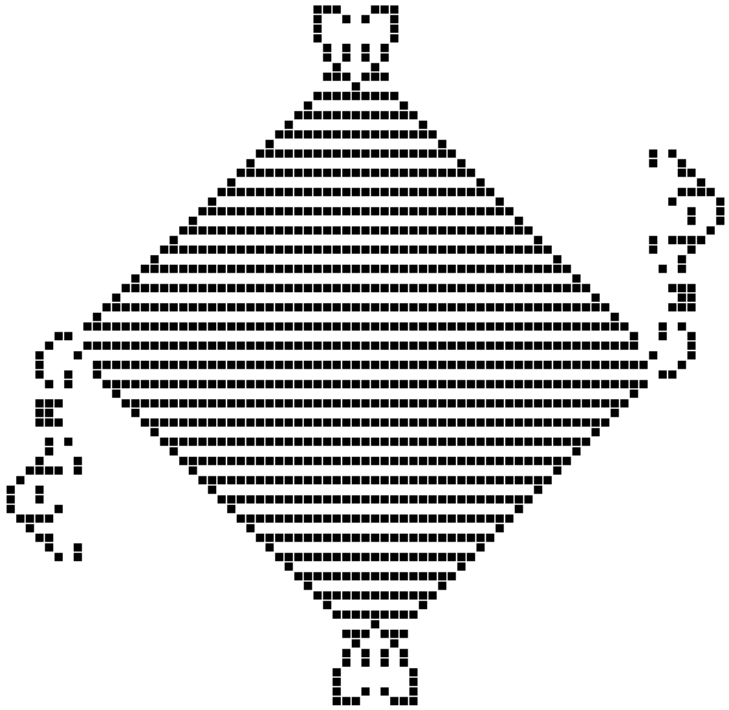

Figure 4.19: The spacefiller Lifeform of David Bell after 46 time steps.

one predecessor (for example, a block has numerous precedessors), and the fact that finite configurations are transformed to finite configurations, Garden of Eden patterns must exist and indeed have been found over the years. Roger Banks found the first one in 1971 with 226 live cells. The one with the smallest number of live cells (at the time of this writing) is 143 due to Achim Flammenkamp (Figure 4.20 right).

The preceding considerations can be discussed in a more mathematical framework. In general, a function $f$ is called *injective* (or *one-to-one*) if whenever $x \neq y$ then $f(x) \neq f(y)$. In other words, distinct points $x$ and $y$ must have distinct *images* $f(x)$ and $f(y)$. This also means that if $f(x) = z$, then no other point can be mapped by $f$ to $z$, since if some other point $y$ did satisfy $f(y) = z$, then we would have $x \neq y$ and $f(x) = f(y)$, a clear violation of the injective property of $f$.

We now consider the collection of all *finite* configurations $C$ of a cellular automaton, such as the Game of Life. By finite, we mean that all but a finite number of cells in the lattice are in the quiescent state. Then any configuration belonging to $C$ is transformed by the local transition function acting on each cell into another configuration and moreover, that configu-

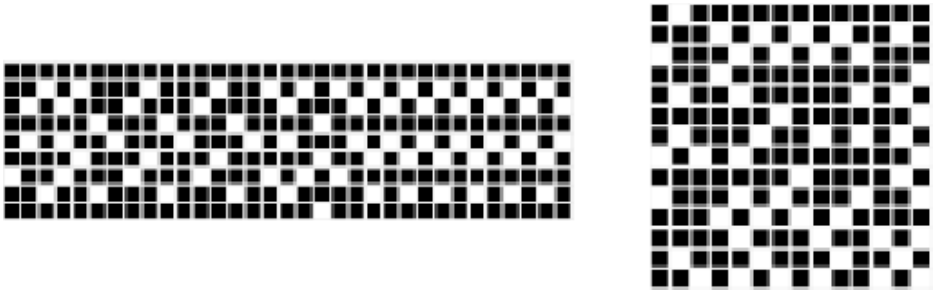

Figure 4.20: The Garden of Eden configurations for the Game of Life of Roger Banks (left) and Achim Flammenkamp, having 226 and 143 live cells respectively .

ration will also belong to $C$ if we make the supposition that if a cell and all its neighbors are in a quiescent state, then the cell remains quiescent at the next time step. Therefore, although the array is infinite in extent, only finitely many cells at any time step ever leave the non-quiescent state and no infinite configuration ever becomes the successor of any finite configuration. We have already encountered an example in Rule 1 where this is not the case since its rule string is given by:

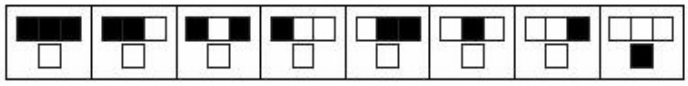

Figure 4.21: Rule 1 allows something to be created from nothing which is not allowed in the present context.

Hence a quiescent neighborhood generates a live central cell at the next time step. This results in an initial finite configuration of one black cell generating an infinite configuration at the next time step and it is exactly this sort of behavior we wish to exclude. We are in a sense getting something from nothing and that is not allowed here.

The transformation from one configuration into another induced by the local action of the transition function can be considered as another function, the *global transition function* $F$ that maps configurations $c$ belonging to $C$ to other configurations $c'$ in $C$. So for example, in the one-dimensional case taking nearest neighbors, if $\phi$ is the local transition function, we define the global transition function by

$$[F(c)]_i = \phi[c_{i-1}, c_i, c_{i+1}],$$

where we have suppressed the time step $t$ in the notation.

If the global transition function $F$ is *not* injective, then this would mean that two distinct configurations, say $c$ and $c'$ in $C$ would be mapped to the same configuration (remembering that $F$ is induced by the local transition function), so that there would be a configuration with two predecessors ($c$ and $c'$) as was discussed above. Therefore, having more than one predecessor configuration is equivalent to the global function $F$ not being injective.

Since $F$ is a mapping from $C$ to $C$, symbolically, $F : C \longrightarrow C$, we can ask if *every* configuration $c$ in $C$ arises as the image of the global transition function acting on some other $c$ belonging to $C$? If the answer is yes, then we say that the function $F$ is *surjective* (or *onto*). In other words, for every $c'$ in $C$, there is some other $c$ in $C$ such that $F(c) = c'$. Then, if it so happens that $F$ is *not* surjective, we find that there is some $c'$ belonging to $C$ with no other configuration in $C$ that is transformed into it. Such a configuration $c'$ is then a Garden of Eden configuration since it would have no precedessor arising from the global transition function $F$, and thus from the local transition function. Therefore, the existence of a Garden of Eden configuration is equivalent to the global transition function being not surjective.

We can now state the:

**Moore-Myhill Theorem**. *If $C$ is the collection of all finite configurations and $F : C \longrightarrow C$ is the global transition function, then $F$ is not injective if and only if it is not surjective.*

In other words, if some configuration has more than one precedessor ($F$ is not injective), then there is some configuration with no precedessor ($F$ is not surjective), and conversely. Moore's theorem is the first result and Myhill's the converse. In most contexts injective and surjective are two mathematical properties that normally have nothing to do with one another.

The global transition function (in the preceding context) being surjective is now also injective and this latter is equivalent to the cellular automaton being *reversible* (also called *invertible*), that is, at every time step it is possible to go back to a unique predecessor configuration, as was discussed in the section on reversibility in Chapter 3. Since we know that the Game of Life has Garden of Eden configurations, then by the Moore-Myhill theorem it cannot be reversible. But of course we already know this as the 'beehive' configuration has multiple predecessors as is seen in Figure 4.7. In general, given the local transition function for a two-dimensional cellular automaton, the question of reversibility is undecidable (Kari [1990]), although for one-dimensional cellular automata, the question of reversibility is decidable (Amoroso and Patt [1972]).

When Moore proved his theorem, he was interested in the question of when a machine was unable to replicate itself. About the Garden of Eden configuration, Moore had this to say. "Since it is a machine that cannot arise by reproduction, it must be a machine that cannot reproduce itself."

Another result of Moore's about the rate at which a cellular automaton configuration can replicate itself is found later in Section 4.3.

Interestingly, if we drop the restriction of finite configurations then in the general case we have the following result of Hedlund [1969] who was one of the earliest pioneers in the subject of cellular automata:

**Hedlund Theorem**. *If the global transition function on the set of all configurations is injective then it is also surjective.*

The situation is more straightforward with respect to additive CA. If $\varphi$ is the local transition function acting on cell states $c_1, c_2, ...c_n$ in some neighborhood given by

$$\varphi(c_1, c_2, ...c_n) = \sum_{i=1}^{n} \lambda_i c_i \mod m,$$

then the global transition $F$ is surjective if the greatest common divisor of all the numbers $\lambda_1$, $\lambda_2$, ... $\lambda_n$, $m$, is 1.

### 4.1.7 Universal Computation in Life

An outline of a proof that the Game of Life was capable of universal computation was presented by John Conway in 1982 (in the book in *Winning Ways for your Mathematical Plays,* vol.2) and independently by William Gosper. The key here is to use a glider gun to emit gliders at regular intervals. This permits the transmission of information from one region to another and simulates the electrical pulses of a regular computer. As the Game of Life is a Class IV cellular automaton (whose Langton parameter is $\lambda = 0.273$ which lies in the phase transition region), it is not surprising that it could be capable of universal computation, in view of Wolfram's conjecture that all Class IV automata should have this capability.

As was mentioned in the Preliminaries, at the heart of any computer system is the construction of the 'logic gates' NOT, AND, OR. What Conway and Gosper demonstrated was that each of these logic gates could be constructed within the Game of Life, together with a form of infinite memory storage.

In order to construct the NOT gate we consider that $P$ is either a 0 or 1 and a data source gun that emits a glider whenever $P$ is 1 and nothing whenever $P$ is 0. An emitter glider gun at $E$ is positioned as in Figure 4.22 and is synchronized to fire gliders simultaneously. Whenever two gliders collide they will annihilate one another. Hence, $P$ being 0 permits the unfettered passage of a glider from the gun $E$, thus turning a 0 into a 1, whereas if $P$ is 1 it emits a glider that collides and annihilates the glider from the gun $E$ resulting in a void that turns the 1 into a 0. The result at receptor $R$ is *not* $P$.

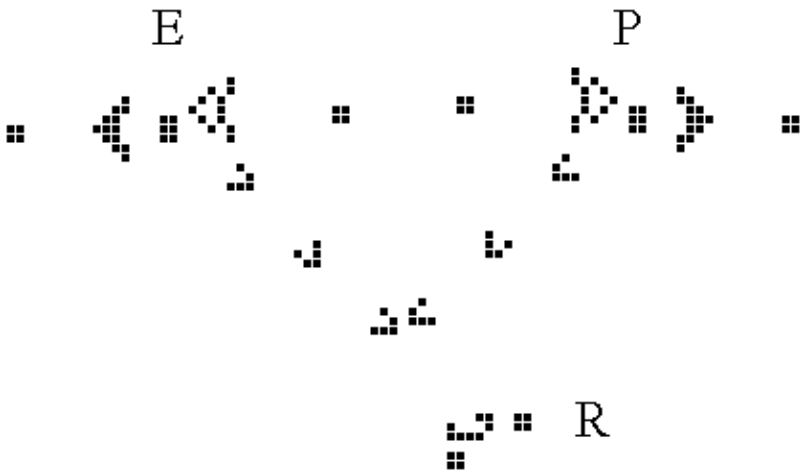


Figure 4.22: The implementation of the NOT gate in the Game of Life. If $P$ is 0 the result at the receptor $R$ is 1 and if $P$ is 1 the receptor receives no input, or 0 since the colliding gliders annihilate one another.

We can construct the AND logic gate $P \wedge Q$ by adjoining a bit more structure to the NOT gave as indicated in Figure 4.23. According to the Truth Table the conjunction $P \wedge Q$ is to be the value 1 only when $P$ is 1 and Q is 1. In all other cases, $P \wedge Q$ is to be 0. Taking $P$ and $Q$ both to have truth value 1 as in the figure, results in a glider emanating from the glider gun $Q$ being annihilated by the glider emanating from the emitter $E$, whereas the glider from $P$ passes unhindered to produce a 1 at the receptor $R$. However, if $P$ is 1 and a glider emanates from $P$, and $Q$ is 0 so that nothing emanates from $Q$, then the glider from $P$ this time is annihilated by the one from $E$ and the receptor $R$ receives no data, hence is 0. On the other hand, if $P$ is 0 and $Q$ is 1, then the glider from $Q$ is annihilated by the glider from $E$ so that again $R$ becomes 0. Finally, if both $P$ and $Q$ are 0 then the glider from the emitter $E$ is eaten by the glider eater and $R$ is once again 0.

The OR gate incorporates a portion of the AND gate as depicted in Figure 4.24 but now we are provided with two emitter glider guns, both designated $E$. Note that $P \vee Q$ has truth value 1 if $P$ or $Q$ is 1 (or both) and is 0 in all other cases. Suppose firstly that $P$ is 0 and $Q$ is 1 as in the figure. Then the gliders from $Q$ are annihilated by the emitter on the right, but the emitter on the left produces a signal at the receptor $R$. Similarly, if $P$ is 1 and $Q$ is 0, $P$'s gliders are annihilated this time, but the receptor $R$ still receives an input. If both $P$ and $Q$ are on, then again $R$ receives a signal. Only when both $P$ and $Q$ are off do the gliders from each emitter

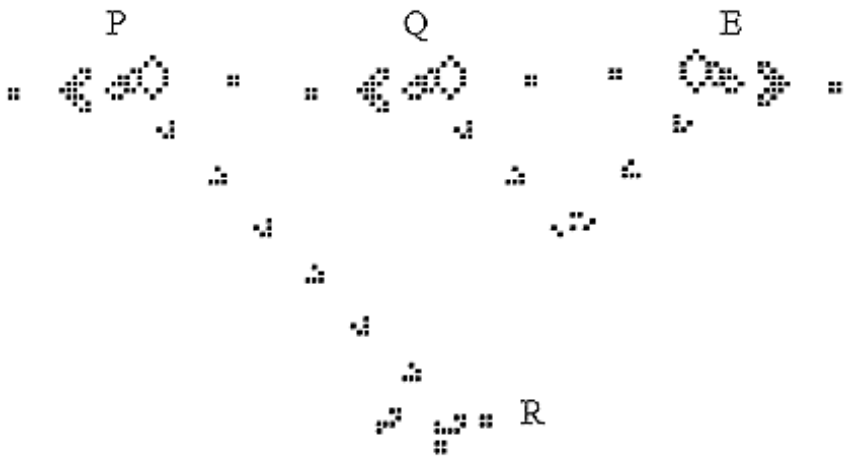


Figure 4.23: The AND gate in the Game of Life. Only when both $P$ and $Q$ are on does the receptor $R$ receive any data. If only $P$ or $Q$ is on (i.e. emitting gliders), then the gliders from $E$ annihilate them and the receptor $R$ receives nothing. Note the glider-eater at the bottom left which mops up gliders emitted by $E$ when both $P$ and $Q$ are off.

annihilate each other and there is no signal at the receptor.

Memory storage for the universal computer is accomplished by simulating a Minsky register by sliding a Life block utilizing a salvo of gliders to either push it or pull it along a diagonal. In the Conway model, the block (whose location gives its numerical value which can be arbitrarily large) could be pulled 3 units along a diagonal by a 2-glider salvo, whereas a 30-glider salvo was required to push the block 3 units along the diagonal. A test for zero was also implemented. This 'sliding block memory' was later improved by Dean Hickerson in a much more practical construction that allowed a block to be pulled just one unit by a 2-glider salvo and pushed one unit by a 3-glider salvo. This latter became the basis for the construction of a universal register machine implemented in Life by Paul Chapman (http://www.igblan.com/ca/) in November 2002. This machine is capable of universal computation with communication between the various components carried out by lightweight spaceships. Therefore the Game of Life has all the computational ability of any modern electronic computer.

In this context it should also be mentioned that Paul Rendell in 2000 constructed a Turing machine, however, with a finite tape which could in principle be extended to a universal Turing machine and thus capable of universal computation.

In the spirit of von Neumann, Conway in *Winning Ways* demonstrated that there are configurations in Life capable of self-replication.

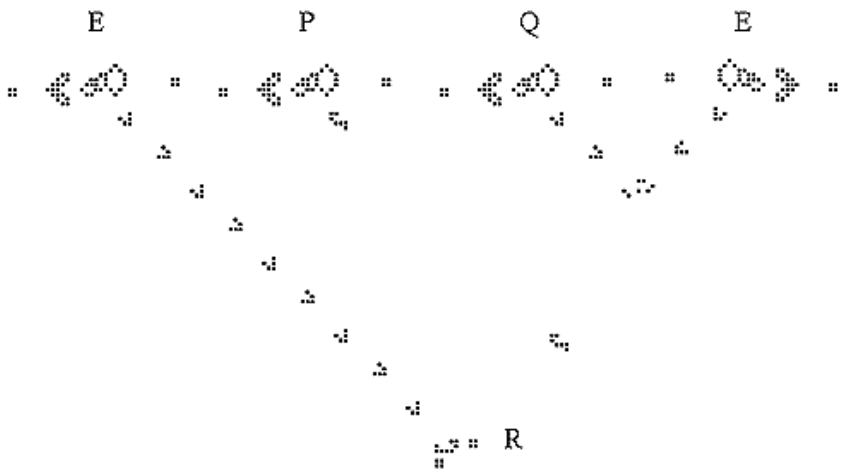


Figure 4.24: The OR gate in the Game of Life. If both $P$ and $Q$ are off then the gliders from each emitter $E$ annihilate each other and there is no signal at the receptor $R$. In the other cases when either $P$ or $Q$ are on or both are on, a signal is received by the receptor $R$ from the emitter on the left.

There are many websites devoted to the Game of Life. Actually seeing the mélange of Lifeforms develop and evolve is a wonderful experience to behold. One of the best implementations is *Life 32* by John Bontes which can be downloaded at: http://psoup.math.wisc.edu/Life32.html. An excellent resource website is Paul Callahan's:

http://www.radicaleye.com/lifepage/lifepage.html#glossback.

## 4.2 Other Automata

There is actually a myriad of two-dimensional cellular automata that have been created over the past thirty years. One of the classical models is called *Brian's Brain* and is due to Brian Silverman. The automaton has three cell states denoted as 'ready'(0), 'firing'(1), and 'refractory'(2) respectively. The rules bear some resemblance with how neurons in the brain behave (a more sophisticated model of neural activity will be presented in Chapter 5):

• A cell fires only if it is in the 'ready'(0) state and exactly two of its (eight) neighbors are 'firing'(1);

• Upon firing, a cell changes to the 'refractory' state (2) for one time step and then reverts to the 'ready' state (0).

As in the Game of Life, there are various Brainforms such as 'haulers', 'butterflies' and 'twizzlers'. The butterflies are the analogue to the gliders

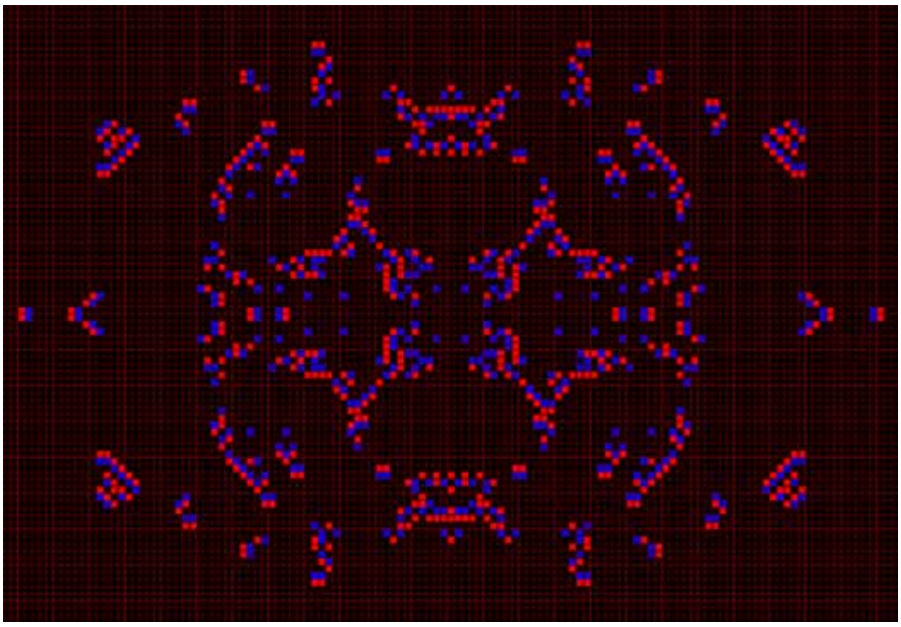

Figure 4.25: The three state cellular automaton Brian's Brain created by Brian Silverman. Waves of live cells tend to sweep across the array. Here the colors are: black = ready, red = firing, blue = refractory.

in Life and move diagonally at the rate of one cell every four time steps. A 'butterfly gun' has been implemented by Rudy Rucker in his CelLab environment.

Another ingenious creation by Silverman is called *Wireworld* and allows for the creation of sophisticated electronic circuits. This automaton has four states: 'background'(0), 'electron head'(1), 'electron tail'(2), and 'wire'(3) and the following set of rules for each time step:

- 'Background' cells never change their state;
- 'Electron head'(1) cells change their state to 'electron tail'(2);
- 'Electron tail'(2) cells change their state to 'wire'(3);
- 'Wire'(3) cells change their state to 'electron head'(1) if one or two of its eight neighbors are 'electron heads'(1).

An adjacent pair of an 'electron tail' (T) and 'electron head' (H) comprise an 'electron' which can be sent along a strand of 'wire' (W) cells in accordance with the preceding rules ($TH$WWW...$\longrightarrow$W$TH$WW...). With this set-up it is possible to build AND, OR, and NOT gates, memory storage and hence a computer. An AND gate is depicted in Figure 4.26.

A simple adder has been constructed by student Peter Lane that computes the sum in base 2 of two inputs as in Figure 4.27.

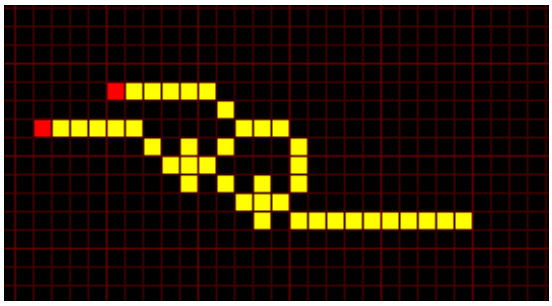

Figure 4.26: The arrangement of the AND gate in *Wireworld.* Two electron head cells are the inputs at the left along paths of wire cells. Only when there are two electrons entering this configuration will the output be an electron. Other cases result in no output.

Here the two inputs are given on the left and fed into the circuit so that the following binary output is computed:

| input A | input B | top output ($2^1$) | bottom ouput ($2^0$) |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

A binary adder has also been demonstrated in the Game of Life by D.J. Buckingham [1978].

Another interesting two-state, two-dimensional cellular automaton is called Vote (also called Majority) created by Gérard Vichniac and is a totalistic rule given by the rule table:

| sum | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| $c(t+1)$ | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

where $c(t+1)$ is the value taken by the central cell of the 9-cell Moore neighborhood at the next time step. Here one can easily see where the automaton gets its name. If 5 or more of the cells (i.e. a majority) are '1', then the central cell's state also takes the value '1' at the next time step. But if less than 5 cells (i.e. a minority) take the value '1', then the central cell becomes '0'. The time evolution from a random initial state depends critically on the initial concentration of 1's and 0's and yields large shifting regions made up respectively of the two states and dominance at the initial stage leads to even greater dominance at the equilibrium stage (Figure 4.28). The totalistic code number is 992 which is 1111100000 in

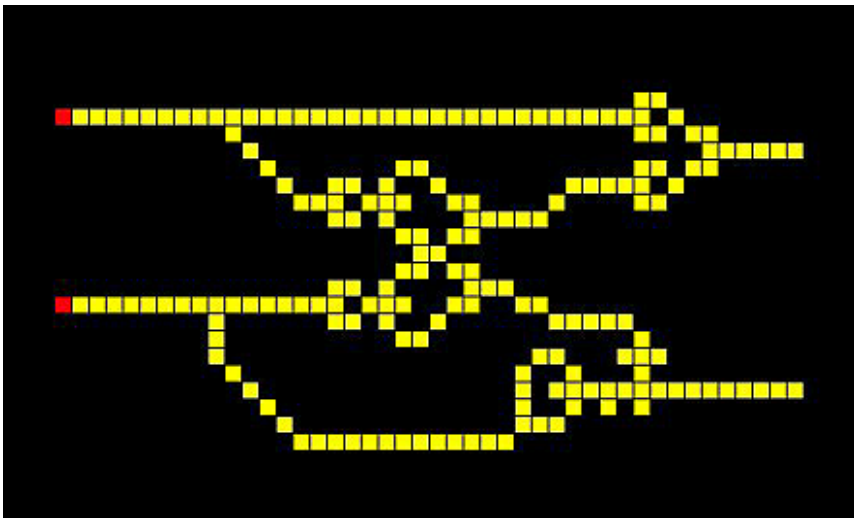

Figure 4.27: A simple adder in *Wireworld* that computes the sum of two inputs in base 2 with the result outputed at the bottom.

base 2. Von Neumann neighborhoods can also be considered so that $c(t+1)$ becomes 1 whenever the neighborhood sum is 3 or more. These automata are related to models of percolation and pattern formation on animal coats.

There is another more scientific way to depict the Vote rule by using the so-called *Heaviside step-function* $H$, also referred to as a *threshold* function.. This is defined as follows:

$$H(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ 0 \text{ if } x < 0 \end{cases} .$$

Thus you obtain an output of 1 if the quantity $x$, whatever it happens to be, is nonnegative, and otherwise the output is 0. If we denote the nine cells of the Moore neighborhood (with a slight abuse of our usual notation) by $c_1(t)$, $c_2(t)$, ... $c_9(t)$, then according to the Vote rule, the value of the central cell at the next time step is given by:

$$H\left(\sum_{i=1}^{9} c_i(t) - 5\right),$$

whereby if five or more of the cells have state value '1', then the function $H$ returns the value '1', otherwise it is '0'. The value of 5 is just a threshold value that turns 'on' the central cell at the next time step once the threshold is reached. This notion is an underlying feature of various cellular automata models in biology (see Section 5.4).

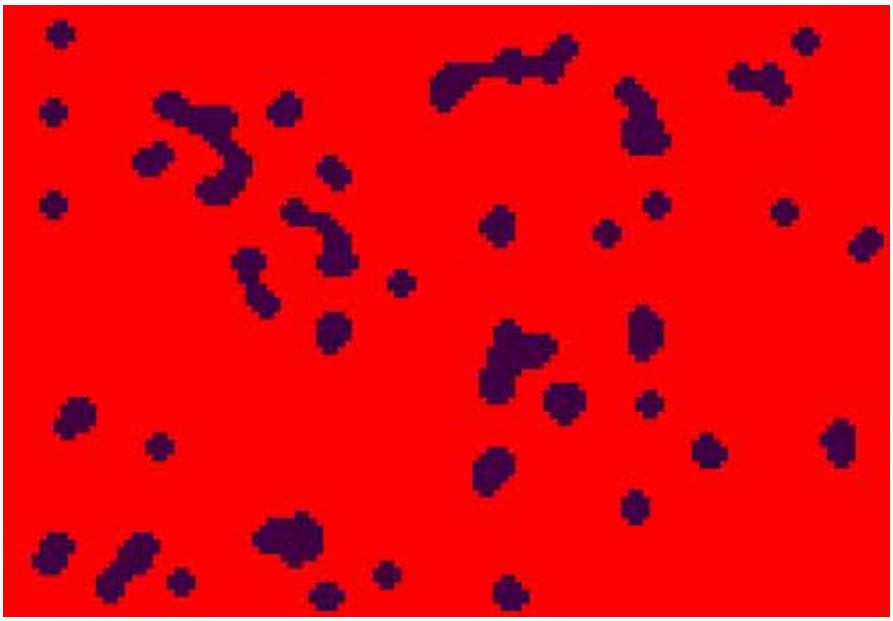

Figure 4.28: The cellular automaton Vote which decides the value of the central cell of a Moore neighborhood by what a majority of the values of its neighbors are. The initial configuration had 60% red (zero state) cells and the above is the equilibrium state.

Another variant of Vote due to Vishniac reverses the states of the sum 4 and 5 values.

| sum | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| $c(t+1)$ | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |

This has the effect of permitting interchange at the boundary between regions of opposing colors where the majority is not very strong for either. This CA has served as a model of annealing.

### 4.2.1 Partitioning Cellular Automata

A new type of neighborhood was devised by Margolus that is fundamentally different from either the von Neumann or Moore neighborhoods. It consists of a partitioning of the usual lattice into blocks of cells, which is 2x2 in size in the simplest case and which we only consider here. The transition rule, or rather *block rule*, updates the entire block as a distinct entity rather than any individual cell as in the usual cellular automaton. Another unique feature is that two overlapping partitionings into 2x2 are employed, one say

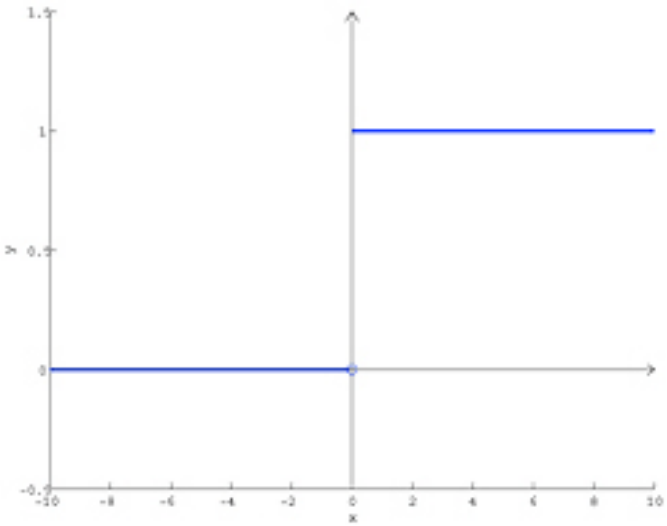

Figure 4.29: The Heaviside step function which only has an output when a nonnegative threshold has been reached.

given by dark lines, and one by light lines as in Figure 4.30 below. At each time step, one switches from one type of neighborhood to the other.

The block rule is applied to all 2x2 blocks alternating between utilizing the dark neighborhoods and light neighborhoods. At first glance this system appears to be very unlike an ordinary cellular automaton, but with more states and neighbors, it can indeed be expressed as an ordinary cellular automaton. The Margolus neighborhood comes into its own in the section on lattice gas automata in Chapter 5..

## 4.3 Replication

It is possible to demonstrate a trivial form of self-replication in a two-dimensional cellular automaton model of Edward Fredkin. In our first example, known as the 'parity rule', given a 4-cell von Neumann neighborhood:

• a central cell state becomes 1 (alive/black) if it had an odd number of black (0) neighbors at the previous time step;

• a central cell becomes 0 (dead/white) if it had an even number of black (0) neighbors at the previous time step.

This is also the transition function of the one-dimensional Rule 90 and is outer totalistic in that we are only considering the sum of the four cell states other than the central one. The effect is a quadrupling of any initial cell pattern (Figure 4.31, top left) each $2^n$ generations, with $n$ depending on the original configuration.

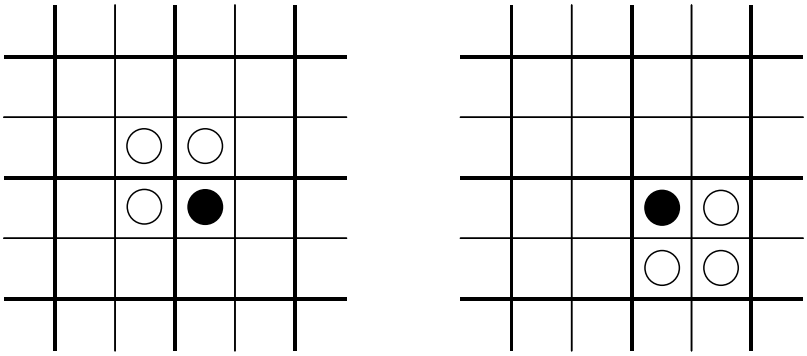

Figure 4.30: A Margolus neighborhood consists of two distinct overlapping 2x2 blocks of cells and a particular cell has successively two overlapping neighborhoods. The neighbors of a black cell (indicated by the black dot) in a 'light' neighborhood are indicated at the left, and the neighbors of a 'dark' neighborhood are as on the right.

Here the outer totalistic rule we have used is given by the transition table:

| sum | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|
| $c(t+1)$ | 0 | 1 | 0 | 1 | 0 |

.

where $c(t+1)$ is the value of the central cell and the initial pattern has replicated after $32 = 2^5$ time steps (as do all the others below). However, if we take into account the state of the inner cell as well as the 4-cell von Neumann neighborhood, we have the totalistic rule:

| sum | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|
| $c(t+1)$ | 1 | 0 | 1 | 0 | 1 | 0 |

with a 5-fold replication of the initial pattern (Figure 4.31, top right).

We can also consider the analogous replication with respect to an 8-cell Moore neighborhood and here we implement the Fredkin (outer totalistic) rule given by the transition table:

| sum | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| $c(t+1)$ | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

that results in an 8-fold replication (Figure 4.31, bottom left).

And lastly, we have a 9-cell Moore neighborhood counterpart with the totalistic rule:

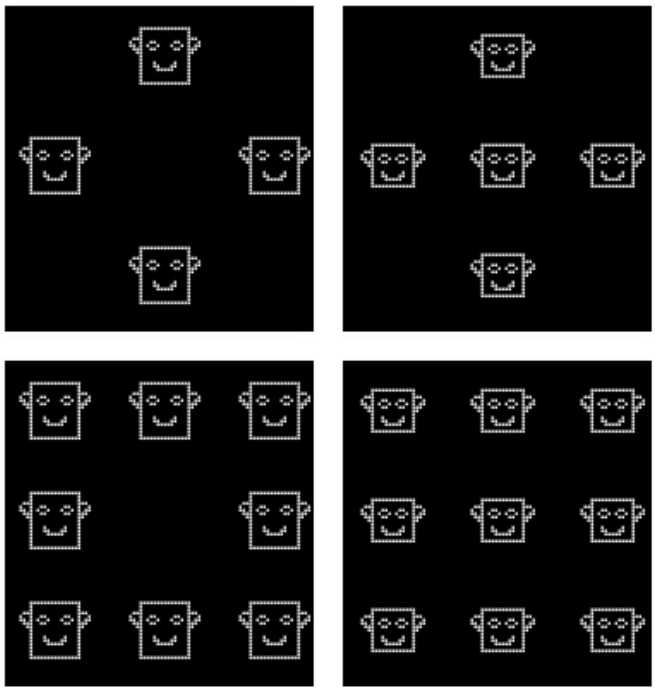

Figure 4.31:

| sum | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| $c(t+1)$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

.

which replicates a two-dimensional configuration 9-fold (Figure 4.31, bottom right). This so-called 'parity rule' can be designated Code 692 which is the value in base 2 of the binary digits.

It must be noted however, that none of the foregoing examples represents the same sort of replication in the von Neumann sense we saw in the Introduction as it is purely a consequence of the transition function and not accomplished by the automaton itself.

Edward F. Moore, whom we encountered previously regarding Garden of Eden configurations also made an interesting observation concerning the rate at which a configuration can replicate. Here again, as in the discussion of Moore's Garden of Eden theorem, we are only considering finite configurations. Again, we also require the further condition that if a cell is in a quiescent state and all its neighbors upon which the transition function acts are in the same state, then the cell remains in the quiescent state at the next time step. If we denote a finite configuration by $c$ and the number of copies of $c$ at any time step $t$ by $\#c$, Moore showed that

$$\#c \leq kt^2,$$

where $k$ is some positive constant. One consequence of this is that a self-reproducing configuration cannot keep doubling with each time step, since the values of $t^2$ are $1, 4, 9, 16, 25, 36, 49...$ and doubling yields $\#c$ values of $2, 4, 8, 16, 32, 64, 128, ...$

To see how Moore established this result, suppose that the smallest *square* array that contains the initial configuration $c$ has $s$ cells on a side, and hence an area of $s^2$. At the next time step, $t = 1$, the square array can only grow (via the action of the transition function) by one cell on each of its four sides, so that the square array now has area $(s+2)^2$. At time step $t = 2$, the square array will have an area equal to $(s+4)^2$, and in general, the array will have area $(s + 2t)^2$ at each time step $t$. This value also represents the maximum number of alive (non-quiescent) cells at each time step $t$. Letting $a$ denote the number of alive cells of the configuration $c$, then the number of copies of $c$ at each time step is at most

$$\frac{(s + 2t)^2}{a}.$$

By expanding and simplifying this expression, we arrive at Moore's inequality above.

## 4.4 Asynchronous Updating

In general, it has been found that the asynchronous cellular automata evolve much differently from their synchronous counterparts. For example, cyclic dynamics can only occur with synchronous updating although the set of stable attractors of a CA are the same for both update regimes (Schönfisch & de Roos [1999]). Indeed, any configuration that is stable under one updating regime is stable under any other since it is of no consequence in what order the cells are updated. In addition, various patterns formed in synchronous updating are absent with asynchronous updating. For example, in the two-dimensional version of the Iterated Prisoner's Dilemma (see Section 5.3), Huberman & Glance [1993] found that the complex mosaics of defecting and cooperating cells attained with synchronous updating completely disappeared with asynchronous updating and the CA reached a fixed state of entirely defecting cells. Similarly, we have already mentioned that the Game of Life also converges to a stable state with asynchronous updating.

So, the question arises as to which updating regime is the most appropriate for modelling physical phenomena. Some argue that asynchronous

updating is more natural since there is no universal clock in Nature. The highly geometric patterns produced in synchronous updating dissolve away with asynchronous regimes and hence are really artifacts of the updating regime. However, weighing in on the side of synchrony we have M. Sipper [1995] who states that, "It may be argued that from a physical point of view synchrony is justified: since we model a continuous spatial and temporal world we must examine each spatial location at every time step, no matter how small we choose these (discrete) steps to be." However, as pointed out by Schönfisch & de Roos [1994], "... the difference between synchronous and asynchronous update is a question of how we look at the (real) process. If we observe only in large time intervals we will see that all cells have been updated (at least) once in one time step, implying synchrony. If we refine the time scale such that in every time interval at the most one event will happen, then we find asynchrony... In summary, it will depend on the actual cellular automata how strong the influence of different updating methods will be."

There are various methods to update cells asynchronously and these fall into two distinct categories, *step-driven* and *time-driven.* In step-driven updating, each cell of the array is updated by some algorithm one cell at a time. For example, in the simplest case, a fixed directional line-by-line sweep of each row can be made to update each of the cells sequentially. Or, the cells can be randomly ordered and each pass is made through this ordering or one can even take a different ordering for each pass. In time-driven updating, each cell has an internal clock that 'wakes up' the cell at some specific point in time so that its state can be updated. This waking up can also be done in a stochastic manner so that each cell will be updated with a certain probability, $p$.

In the sequel, we consider an example found in Roli & Zambonelli [2002] of the local rule:

- A dead (black) cell becomes alive (white) if it has 2 alive neighbors;
- A living cell remains alive if it has 1 or 2 neighbors alive, otherwise it dies..

These criteria are a watered-down version of the Game of Life but it has some star-studded dynamics of its own in the synchronous case (Figure 4.32).

However, starting with the same block of 4 black cells in the asynchronous case is much less dramatic and leads to a stationary state as in Figure 4.33.

Roli & Zambonelli considered a third type of updating regime in addition to synchronous and asynchronous that they termed *dissipative.* These CA (called dissipative cellular automata – DCA) have asynchronous time-driven updating, but also allow an external perturbation to change the state of any of the cells of the array concurrently with the transition function. This can be thought of in terms of the environment interacting with the automaton by providing 'energy'. The notion of a DCA was inspired by the dissipative

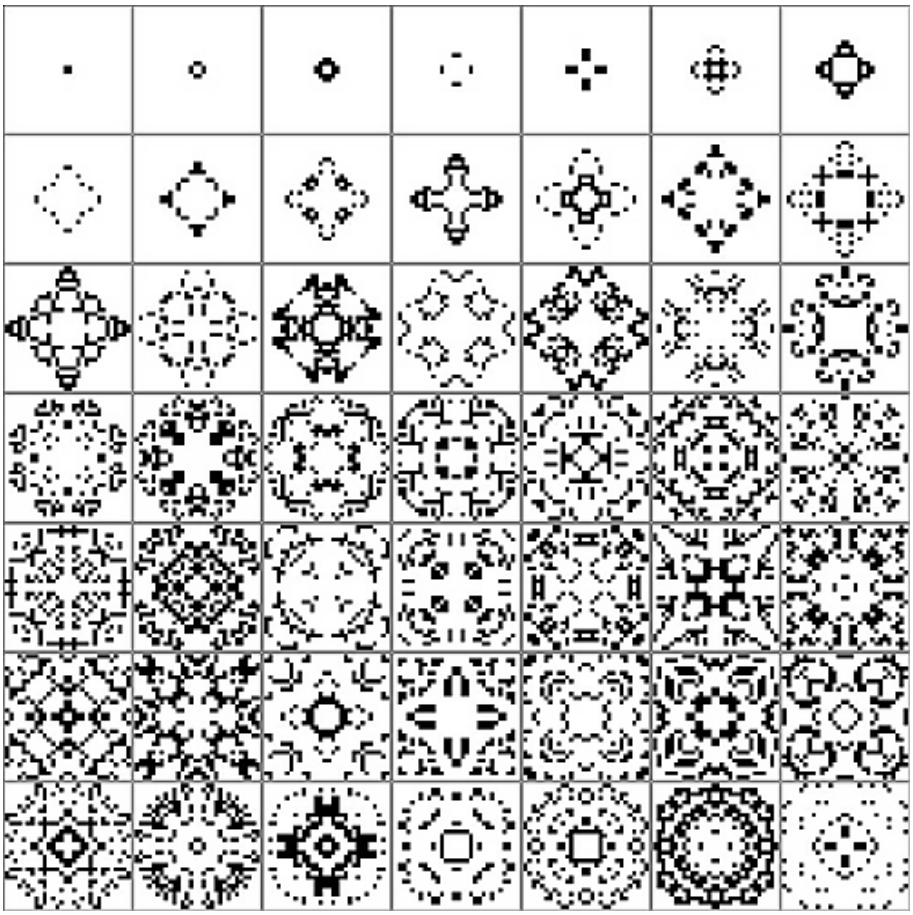

Figure 4.32: Starting with four black cells and the CA in the text under synchronous updating. These are the first 49 configurations with many similar that follow.

systems much studied by Ilya Prigogine and his school (cf. eg. Nicolis & Prigogine [1989]).

One way to achieve interaction with the environment is for the perturbation to take the form of forcing cells at random to become alive with a certain probability, $p_d$. The degree of perturbation must be sufficiently high to affect the dynamics of the CA, but not too high so as to make the behavior essentially random. The ratio $p_d/p_c$ is the crucial factor here and although stationary configurations are not reached, nevertheless large-scale patterns do persist.

Because of the open nature of DCA to their environment, the authors assert that, "the dynamic behavior of DCA is likely to provide useful insight into the behavior of real-world open agent systems."

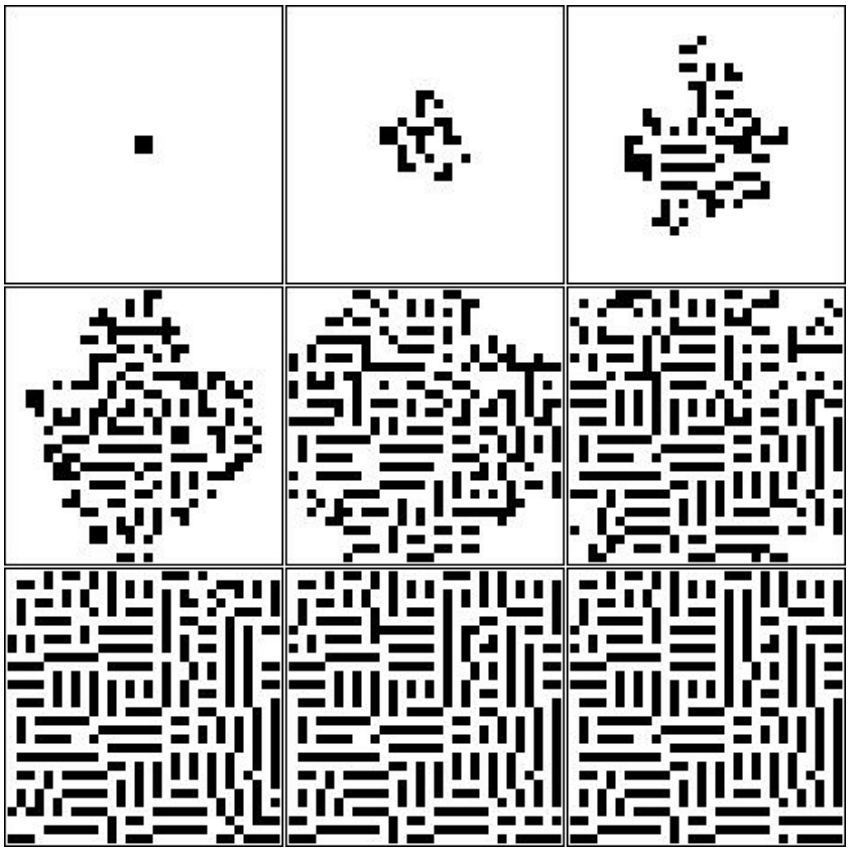

Figure 4.33: What a difference a delay makes! The asynchronous updating of the same CA with each cell being updated or not with a probability of $p_c = 0.5$. The configurations are a sampling every 10 steps and the final configuration is stable.

A certain class of CA was shown by Goles and Martinez [1990] to have both stationary and cyclic states with synchronous updating but only stationary states were attainable with asynchronous updating.

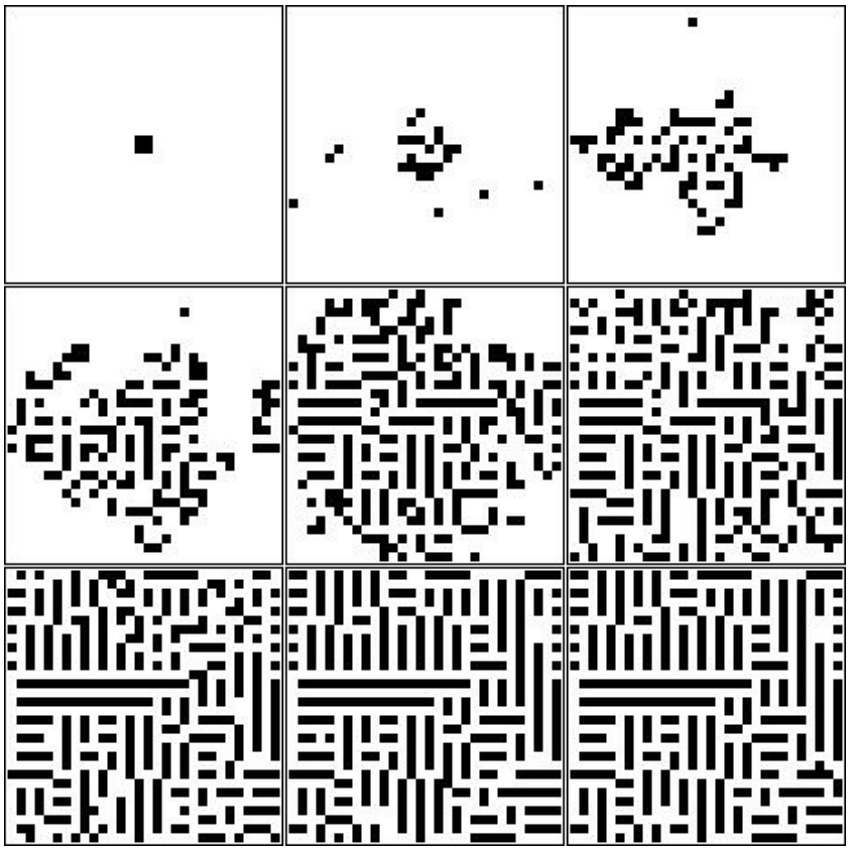

Figure 4.34: The evolution of the dissipative CA whereby each cell has a probability of $p_d = 0.001$ to be perturbed to be alive. Large scale patterns evolve that are seen to persist but are not completely static. Interestingly, these patterns arise even without any black cells in the initial conditions due to the perturbations.