Physica 22D (1986) 120–149
North-Holland, Amsterdam

STUDYING ARTIFICIAL LIFE WITH CELLULAR AUTOMATA*

Christopher G. LANGTON
Department of Computer and Communication Sciences, The University of Michigan, Ann Arbor, MI 48109, USA

Biochemistry studies the way in which life emerges from the interaction of inanimate molecules. In this paper we look into the possibility that life could emerge from the interaction of inanimate artificial molecules. Cellular automata provide us with the logical universes within which we can embed artificial molecules in the form of propagating, virtual automata. We suggest that since virtual automata have the computational capacity to fill many of the functional roles played by the primary biomolecules, there is a strong possibility that the 'molecular logic' of life can be embedded within cellular automata and that, therefore, artificial life is a distinct possibility within these highly parallel computer structures.

1. Introduction

Biochemistry is the study of the molecular basis of life. Lehninger, in the introduction to his classic text on biochemistry [22], asks:

> If living organisms are composed of molecules that are intrinsically inanimate, why is it that living matter differs so radically from nonliving matter, which also consists of inanimate molecules? Why does the living organism appear to be more than the sum of its inanimate parts? Philosophers once answered that living organisms are endowed with a mysterious and divine life-force. But this doctrine, called *vitalism*, has been rejected by modern science, which seeks rational and, above all, testable explanations of natural phenomena. The basic goal of the science of biochemistry is to determine how the collections of inanimate molecules that constitute living organisms interact with each other to maintain and perpetuate the living state.
>
> The molecules of which living organisms are composed conform to all the familiar laws of chemistry, but they also interact with each other in accordance with another set of principles, which we shall refer to collectively as *the molecular logic of the living state*. These principles do not necessarily involve new or as yet undiscovered physical laws or forces. Instead, they are a unique set of relationships characterizing the nature, function, and interactions of *biomolecules*....

In this paper we will explore the possibility of implementing the 'molecular logic of the living state' in an *artificial biochemistry*, based on interactions between *artificial molecules*. These artificial molecules are modeled as *virtual automata*, which are free to roam around in an abstract computer space and interact with one another. We use *cellular automata* to implement the abstract computer space within which the virtual automata reside. We show that cellular automata are capable of supporting virtual automata that are equivalent to Turing machines and can thus perform any computable task. On this basis, we propose that the notion of the 'molecular logic of the living state' can be captured by the interactions of virtual automata and thus that the existence of *artificial life* within cellular automata is a distinct possibility.

We will approach this study in the following manner. First, we will discuss some of the major functional roles carried out by biomolecules. Then we will look at cellular automata and a study of

*This work was supported by NSF grant MCS 83-05830.

the way in which systems of interacting artificial molecules can arise spontaneously in these highly parallel computing structures. Next, we will look at these artificial molecules as 'virtual' automata and examine their potential for carrying out the kinds of functional roles that are carried out by the various biomolecules. We then show examples of some artificial biochemistries and two examples of systems of virtual automata that support other 'life-like' behaviors: a simulated insect colony and a self-reproducing structure. We conclude with a brief discussion of how such systems might by applied to the study of emergent behavior in general.

2. The functional roles of biomolecules

In order to understand how to build artificial molecules, we need to understand the functional roles they must perform if they are to participate in anything like an artificial biochemistry.

There are four major classes of biomolecules: proteins, nucleic acids, polysaccharides, and lipids. From the point of view of 'molecular logic', the first two classes (the proteins and nucleic acids) cover most of the important functional roles. The primary functional roles provided by these two classes are:

Catalysis. Proteins constitute *enzymes*, which are primarily responsible for mediating the chemical interactions between the biomolecules of the cell. Enzymes mediate the chemical interactions between biomolecules by acting as *catalysts*: entities that enter into a chemical reaction and speed up the rate at which that reaction reaches equilibrium. The rate of speed up is so large (a factor of 10^8 or more) that for all practical purposes enzymes determine which reactions happen and which do not. Thus, enzymes constitute "molecular machines", which are active agents in the logic of life. Their function includes the capacity to *recognize* specific structures and to effect changes in them.

Transport. Proteins are primary vehicles for molecular and ionic transport. Many proteins are finely tuned to bind to substrate material in areas where it is in plentiful supply and to release the substrate material in areas where it is in short supply.

Structure. Proteins constitute many of the building blocks out of which cellular components and tissues are constructed. Certain protein conformations, such as the α-helix and β-sheet, provide excellent material for rigid, flexible, or high-tensile strength cellular or extra-cellular structures.

Regulation. Proteins constitute the *repressors* and *hormones* that cells and organisms use to regulate the production and interactions of biomolecules. In this role, they function primarily as messengers that trigger changes in catalytic activity or protein synthesis.

Defense. Proteins constitute the *immunoglobulins* and *antibodies* that fight against invasion by foreign or disruptive agents. These functions involve the recognition of non-native molecules or organisms and the subsequent production of molecular machines to bind together or break down the foreign material.

Information. The nucleic acids DNA and RNA provide for the storage of genetic information and its translation in the processes of protein synthesis. The DNA of a cell is typically covered by a swarm of polymerase enzymes, which initiate the transcription of DNA sequences to some form of RNA: messenger RNA (mRNA), transfer RNA (tRNA), or ribosomal RNA (rRNA). The mRNA – which may be further edited – is guided to ribosomes in the cytoplasm. Ribosomes are made up primarily of rRNA, and function as machines for making proteins. They are assisted in this task by molecules of tRNA, which transport amino-acid building blocks to the ribosomes for incorporation into a polypeptide string in the order dictated by the sequence of bases on the mRNA.

This polypeptide string will 'fold up' into the characteristic shape that will determine its function as a protein. Other enzymes cause the DNA molecule itself to be replicated. These processes involve the storage, transmission, transcription, translation, and replication of information.

The additonal functional roles provided by polysaccharides and lipids primarily involve the storage of energy, although they also provide important structural elements.

Thus, biomolecules fill a wide range of functional roles in the molecular logic of living systems. If we want to implement an artificial version of this logic, we must provide entities that can fill a similar set of functional roles. Note that, while we ask for *functional* similarity, we do not require *structural* similarity as well. We are *not* trying to simulate biomolecules themselves, we are merely trying to find other entities that can fill similar functional roles.

3. Properties of the molecular logic of life distributed

The molecular logic of life is a *dynamic distributed* logic. An initial set of operators and operands goes to work producing more operators and operands, which immediately enter into the ongoing logical 'fray'. Some of these new operators and operands are distributed as new initial sets in the process of self-reproduction. This dynamical character of the molecular logic of life is unlike a typical formal logic which, although it provides an initial set of operators and primitive operands, has no internal dynamics of its own. Instead, formal logics of the standard variety provide people with convenient tools, but they are passive tools not active ones. They must be applied by something or somebody outside of the logical system.

Although the proteins and the nucleic acids are the principal *operators* in the molecular logic of life, they also constitute many of the *operands* on which the operators 'operate'. Thus, a fundamental property of this logic is that its operators can operate on each other. This property of having operators being able to operate on each other provides a logic with special properties, as was demonstrated by Gödel in his famous incompleteness proof [24]. This same property underlies the power of the standard 'von Neumann' stored-program computer, in which a program stored in memory is available as data to another program, or even to itself. The principle requirement for such self-operation is that operators and operands be implemented (or represented) in the same language, i.e., out of the same 'building blocks.'

A program running on a computer captures some of the dynamical spirit of the molecular logic of life. A program in a computer can be started from an initial state and left to run through its instructions and perform its functions on its own. However, most computer programs are designed specifically to avoid modifying themselves or other programs; their attention being focused strictly on data that are not intended to be executed. Even in the field of artificial intelligence, where the LISP programming language makes it very easy to write programs that modify (or create) other programs, this capacity is little used.

In short, neither the static formal logics nor the more active logics implicit in current computer programming languages come close to capturing the dynamic spirit of the molecular logic of life. At any one time in a single cell there may be hundreds of thousands of molecular operators actively engaged in the processes of creating, modifying, and destroying other such operators. Furthermore, this massive, ongoing dynamical logic is superbly regulated by the very processes out of which it is constituted. It is this kind of dynamic interaction of operators as operands that we seek to implement in an appropriate computational medium.

4. Cellular automata

In order to simulate the molecular logic of life efficiently, we need very special computers. Many of the architectural requirements for such computers can be derived from the nature of the dynamical interactions between molecular operators.

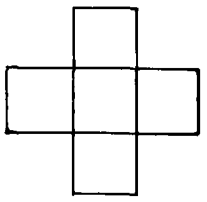
First, the computer structures must support *massive* parallelism, which is to say that they must have many, many computing elements in order to provide for the simultaneous interactions of many, many operators.

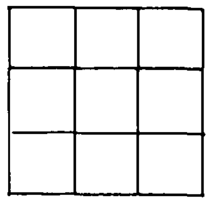
Second, the many computing elements need only be *locally* connected, since almost all of the actions of molecular operators are taken solely in response to local conditions. This local connection property is fortunate indeed when dealing with potentially massive parallelism, because it means that the number of connections per processing element can be independent of the *number* of processing elements.

Third, since molecular operators depend on being able to move around relatively freely within the various compartments of the cell in order to encounter their 'operands', the computer must support the motion of *operators* through the field of processing elements. Although there are architectures that support motion of *operands* through a field of processors containing the operators (e.g., 'systolic' arrays), few if any existing or proposed machines allow the operators to meander about at will, modifying *each other* as well as the data. Thus, an architecture for simulating molecular logic must have the capacity to support the free motion of operators as well as operands.

An architecture that satisfies these criteria was proposed by John von Neumann in the early 1950's, based on a suggestion from Ulam [2, 29]. It is not merely coincidental that the architecture he proposed should be suitable to the simulation of life, for he was attempting to model the process of natural self-reproducton when he suggested it. The architecture he proposed is known as a *cellular automaton* and it can best be understood by looking at an example.

A potentially infinite two-dimensional euclidean space is divided up into an array of unit squares each of which is called a *cell*. Each cell contains an identical copy of the same *finite automaton*, which can be in any one of k discrete states and changes state as a function of its own state and the states of the automata in its immediate *neighborhood*. Time progresses uniformly in the array in discrete steps with all cells changing state simultaneously. The automaton used by each cell is defined by a *transition function*, Θ, which is the same for every cell. Θ maps the states of a local neighborhood to a new state for the cell at the 'center' of that neighborhood at the next time step. Two common neighborhoods are: the *five* cell neighborhood, a cell together with its neighbors to the N, E, W, and S, and the *nine* cell neighborhood, which is just the five cell neighborhood plus the 4 cells to the NE, NW, SE, and SW.


five cell neighborhood


nine cell neighborhood

If there are n cells in the neighborhood of a cell, including itself, then there are k^n possible neighborhood-states. For each of these, Θ must specify one of the k cell-states as the next state for the cell at the center of the neighborhood. There are k states to choose from, so there will be $k^{(k^n)}$ different possible Θ mappings. For example, if $k = 8$ and $n = 5$ then there will be 8^5 or 32,768 possible neighborhood states. For each of these, there are eight choices for assigning the next state of the center cell. Thus, there are $8^{(8^5)} = 8^{32,768}$ possible Θ mappings, an exceedingly large number.

By convention, there is usually one special state (out of the k states possible for each cell) called the *quiescent* state. A neighborhood that consists entirely of cells in this special quiescent state results in the center cell remaining in the quiescent state at the next time step. Thus, an entirely quiescent space will remain so indefinitely. Later, when we are discussing activities in the array taking place against a *quiescent background* we will mean not just that the background is unchanging, but that the background consists of cells in the special quiescent state.

Cellular automata provide computing structures that satisfy the criteria we derived above for simulating the molecular logic of life. They are highly parallel devices with the property that their basic computing elements, the cells, are only locally connected. Furthermore, it turns out to be rather easy to define 'operators' that can migrate freely throughout the array interacting with one another. Indeed, it turns out that such operators can emerge 'spontaneously' in cellular automata.

5. Cellular automata as dynamical systems

There are two general approaches to the study of behavior in cellular automata:

1) Start with specific behaviors in mind and derive a Θ function that will support those behaviors;

2) Start by specifying a Θ function and observe the resulting behavior.

The early investigations with cellular automata were of the former variety [2, 5, 29]. Von Neumann was attempting to discover the logical principles underlying the natural phenomenon of self-reproduction when he first introduced cellular automata. More recently, physicists have been investigating cellular automata using the latter approach [8, 10, 28]. Physicists have become interested in cellular automata because they constitute discrete space/time dynamical systems. A dynamical system is one in which the system's variables change as a function of their current values. Thus the behaviors of many dynamical systems are governed by systems of nonlinear differential equations, making them difficult to analyze.

The study of a dynamical system involves the analysis of its 'phase space', which is the space defined by all of its variables. The phase space covers all possible states of the system: each point in phase space represents a unique value for each of the system's variables. The system's behavior in time is represented as a path through its phase space, and the study of dynamics involves the characterization of the geometry of these paths [1, 3, 6, 23]. In general, when a physical system is started from some initial state, the point representing its state will travel around in some restricted region of the phase space.

There are three possibilities for the long term behavior of the path of a system's behavior in phase space: it will stop moving altogether, it will fall into a closed cycle, or it will not close on itself at all. In the first case, the system is said to have evolved to a *fixed point* or *limit point*, in the second case to a *limit cycle*, and in the third case to something called a *strange attractor*. Actually, all three are kinds of *attractors*, the name deriving from the fact that if a system is in a state that is 'near' an attractor in phase space, it generally evolves 'toward' the state or cycle represented by that attractor. The set of all points in phase space from which a system can be started and still end up at the same attractor is termed the *basin of attraction* for that attractor. The path between the point in phase space at which a system is started and the attractor it ends up at is termed a *transient*. There may be many attractors in the phase space of a system, and hence many basins of attraction. Systems governed by the third class of attractor, the strange attractors, are associated with behaviors that are termed *chaotic*, and give the appearance of being random and unpredictable. Thus, unless we simulate every step of their time evolution, we must resort to probabilities when describing the behavior of such systems, even when the rules governing their behavior are completely deterministic.

The analogue of the phase space for a cellular automaton is its 'state space'. At any one time there is a unique distribution of states over the cells of the automaton and this distribution is represented as a point in state space. The time evolution of a cellular automaton can be studied by observing the 'path' that it follows in its state space. It turns out that attractors often abound in the state spaces of cellular automata as well. It is primarily this feature of cellular automata that has 'attracted' the attention of physicists. Cellular automata can exhibit behaviors characteristic of all three of the classes of attractor mentioned above.

Stephen Wolfram has undertaken a detailed study of cellular automata and their relationship to dynamical systems [32, 33]. He has identified the following four qualitative classes of cellular automaton behavior:

- Class 1 evolves to a homogeneous state.
- Class 2 evolves to simple separated periodic structures.
- Class 3 yields chaotic aperiodic patterns.
- Class 4 yields complex patterns of localized structures, including *propagating* structures.

Wolfram finds the following analogues for his classes of cellular automaton behavior in the field of dynamical systems [33].

- Class 1 cellular automata evolve to *limit points*.
- Class 2 cellular automata evolve to *limit cycles*.
- Class 3 cellular automata evolve to *chaotic* behavior of the kind associated with *strange attractors*.
- Class 4 cellular automata 'effectively have very long *transients*, and no direct analogue for them has been found among continuous dynamical systems'*.

Thus, cellular automata seem to provide a nice 'bridge' between theoretical physics and the formal theory of automata. Physical behaviors that have been captured in cellular automata can be subjected to logical analysis from the perspective of automata theory, which may provide new insights into the nature of the low level physical phenomena generating the behaviors. Our purpose in this paper is to argue that cellular automata constitute an equally useful bridge between automata theory and theoretical biology by providing a convenient and simple formal system within which life-like behaviors can be 'captured' and analyzed, in isolation from their physical basis. It is a matter of no small import that biologists and physicists could use the *same* formal system to capture and analyze their respective behaviors of interest. By 'transitivity', cellular automata provide a useful bridge between theoretical biology and theoretical physics.

*We will have more to say about this in section 8.

6. Notes on the simulations

In the sections that follow, we will consider the results of some empirical studies of cellular automaton behavior. These results were obtained using a general purpose cellular automaton simulator, CELLSIM, developed by the author. The simulator is written in the C programming language, and runs on Apollo Corporation DN600 or DN660 color workstations. Plate 1 shows the 'control panel' of the virtual CELLSIM machine.

CELLSIM allows simulations of one- or two-dimensional, *finite* cellular automata, with array sizes of up to 256×256. The most convenient working size has turned out to be a 64×64 array. The CELLSIM simulator will update a 64×64 array almost 16 times per second on a DN660, which gives the experimenter a good visual understanding of the *dynamics* of the processes going on in the array. CELLSIM will allow total transition functions to be defined for up to sixteen states per cell over the five cell neighborhood, and up to four states per cell over the nine cell neighborhood. The edges of the arrays are wrapped around, yielding the topologies of a torus for two dimensional arrays and of a circle for automata of one dimension. All of the simulations discussed below take place on a finite, unbounded torus using the five cell neighborhood. Most of the simulations use 8 states per cell†.

†It is very difficult to convey dynamical activity via static pictures and figures. Thus, a video-tape containing the sequences from which the figures were taken will be available from Aerial Press, Box 1360, Santa Cruz, CA 95061, USA.

7. Emergent behavior in cellular automata

It is convenient to think of a cellular automaton as a logical universe all of its own, with its own *local* physics: the transition function Θ. This universe can be populated with *objects* by specifying an initial assignment of non-quiescent states to some of the cells in the array. Once Θ and the initial state of this logical universe have been specified, the universe can be started and the resulting global behavior observed. As might be surmised from the number of Θ mappings, there is quite a wide range of possible behaviors.

One of the things that is so interesting about cellular automata is that any behavior that appears on scales larger than that of a single cell will be *emergent* behavior. No global behavior is specified in the Θ function explicitly: global behavior will emerge hierarchically – on the 'shoulders' so to speak – of the local behavior.

In this section, we will take the approach of generating Θ functions and then observing the resulting behavior. We want to do two things. First, we want to get a general feeling for the kinds of global behavior that can emerge in cellular automata. We will do this by observing the behavior yielded by a succession of randomly generated Θ functions, each supporting more 'activity' in the array than the previous one. Second, we want to investigate further some subset of these global behaviors that shows promise for supporting a kind of 'artificial biochemistry'. This requires that we find a way to produce Θ functions that will yield behavior that falls in the relevant subset.

In order to generate random Θ's we start with an effectively undefined transition function and then fill it in by assigning the next-state of each different neighborhood rule at random. In order do this in some sort of orderly fashion, we define a parameter λ that measures how many neighborhood states are mapped to a non-quiescent state as opposed to the special quiescent state for a particular Θ function. Specifically, λ is defined as

λ = number of neighborhood states that map to a non-quiescent state/total number of neighborhood states ($= k^n$).

λ can be used to generate Θ functions as well, and should provide us with a rough control of the overall level of activity supported by a Θ function. For more than two states per cell ($k > 2$) we expect that with low values of λ most of the neighborhood-states will map to the quiescent state and so there will probably be little activity in the array, while with high values for λ, most of the neighborhood-states will map to non-quiescent states, so there will probably be a great deal of activity in the array. In the case where $k = 2$, high values of λ will tend to reverse the roles of the quiescent and the non-quiescent state, so there will probably be little activity in the array for high values of λ in cellular automata with only two states per cell.

For our simulations, which use eight states per cell and the five cell neighborhood, there are 8^5 or 32,768 possible neighborhood-states. We will designate state zero (0) as the quiescent state and assign it as the image of the neighborhood-state consisting of all five cells in state zero. The other seven states will all be considered to be *active* states. Now, starting with the function undefined except for the quiescent neighborhood transition, we want to go through the remaining 32,767 neighborhood rules, one by one, and assign a 'next-state' image to each one. We pick a λ between 0 and 1, and let it represent the probability that, for any particular neighborhood rule, we assign a non-zero state as the next-state of the center cell of that neighborhood. For example, if we pick $\lambda = 0.2$, then each possible neighborhood rule will be mapped to the quiescent state with probability $1 - \lambda = 0.8$. The remaining probability of 0.2 is distributed evenly among the other 7 states. Thus there is a $0.2/7 = 0.02857$ chance for each neighborhood rule that it will map to any specific one of the 7 non-quiescent states, and a 0.8 chance that it will map to the quiescent state.

Some caveats are in order concerning generating random Θ functions with the λ parameter. First, this procedure will generally yield Θ's that do not support rotational symmetry. That is, neighborhood states that differ only by rotation will generally not map to the same next-state. This means that the 'universes' controlled by these Θ functions will be non-isotropic and so there will be preferred directions. The simulations that follow do not enforce rotational symmetry, however the procedure is easily modified so that all rotations of a neighborhood-state will map to the same next-state. The structures in which we will be interested are more readily observable in non-symmetric spaces, but occur in symmetric spaces as well.

Second, the λ parameter can only be an approximate measure of activity because there will be special cases when even a very low value for λ will result in a very small set of neighborhood rules that, by chance, all trigger each other and cause a great deal of activity in the array. Similarly, ongoing activity in the array may depend on the presence of very specific configurations which we have little chance of discovering accidentally. Thus, the λ value of a Θ function is an aggregate statistic that is *correlated* with, but not a certain predictor of, a certain level of behavioral complexity. In this sense it is like a Reynold's number, which is correlated with the tendency of a geometric object to produce turbulence in a fluid flowing over its surface.

Third, since we are interested in the emergence of ongoing *dynamical* activity in the array, we should be interested in contrasting dynamic behavior against *all* possible forms of inactivity, whereas the λ parameter only takes into account the kind of inactivity associated with the special *quiescent* state.

Fourth, the association between specific values for λ and specific 'levels' of activity in an array seems sensitive to the size of the neighborhood, the number of states per cell, and the degree of symmetry. These dependencies have not yet been thoroughly explored and so specific correlations should be taken with a grain of salt. For the present, overall trends are sufficient for our purposes.

The above notwithstanding, the λ parameter still serves as a useful first approximation of the potential for activity provided by a particular Θ function. It gives us a 'knob' we can use for the coarse tuning of activity level, a knob that we can turn to different settings in order to generate different Θ maps that will support widely different categories of behavior.

We will now look at the kinds of behavior associated with different settings for λ between 0 and 1. When we start a *random* initial configuration in our simulated array under the Θ's that we generate by the procedure above, we uncover the following behavioral spectrum (plate 2). For $\lambda = 0.0$, of course, all cells become quiescent at the next time step. For λ close to 0, there may be one or two neighborhood states that map to a non-quiescent state, but they will most likely be isolated, and will become quiescent by the next time step. As we raise λ further, we begin to notice that some areas of the array may hold activity for several time steps before becoming quiescent.

When λ is around 0.2, something new happens. We find that a single state will persist, either fixed in place or propagating steadily in one direction. What has happened is that a rule for a neighborhood state that is quiescent except for one cell, maps to the same non-quiescent state. If the one non-quiescent cell was the center cell of the neighborhood, then it will remain at a fixed position in the array. If the one non-quiescent cell was one of the bordering cells of the neighborhood, then that state will propagate steadily through the array. One could call such a rule a *self-triggering* rule in that, once its conditions are satisfied, it helps to set up the conditions necessary for itself to fire again. It is also possible that several neighborhood rules that map to different states get 'linked' together, so that a repeating *cycle* of states stays fixed in place, or propagates across the array.

As λ is raised still higher (e.g., to around 0.3) the activity in the array increases dramatically. We have crossed a threshold into a region of the behavioral spectrum where ongoing activity in the array is almost certain*. Many neighborhood rules will now participate in setting up the conditions for other neighborhood rules to fire, and the activity in the array becomes, in a sense, *self-sustaining*. Whereas the simple, isolated propagating states observed for λ around 0.2 did not interact with each other, one now begins to see several species of *propagating structures* travelling in different directions in the array and engaging in complex interactions when they encounter one another†. These propagating structures, consisting of packets of 2 to 6 co-travelling states, may cycle through several different configurations periodically as they propagate‡. They may also leave behind a trail of 'debris', which can consist of fixed or periodic structures, some of which may propagate themselves.

Much of the activity in the range of λ between about 0.2 and 0.4 takes place against a largely quiescent background. By the time λ is in the range of 0.5 or more, the activity largely dominates the quiescent behavior. In this range, the activity in the array has become quite chaotic and defies simple description. In these chaotic reactions, the individual propagating structures have become like virtual particles, emerging briefly from one complex process only to be absorbed instantly by the next one. Furthermore, these complex processes are no longer just temporary consequences of the collision of propagating structures, they are ongoing processes in their own right, which occasionally emit and absorb propagating structures.

As λ approaches 1, of course, the transition function becomes saturated with transitions to non-quiescent states and the activity in the array becomes one large complex process; all localized, isolated processes in the array having been engulfed.

We might think of the λ scale as a *temperature* scale. For low λ 'temperatures' we observe precipitate-like behavior, where everything is stable and nothing changes, while for high temperatures we observe the behavior of a hot gas where everything changes and nothing is stable. For temperatures in between, where we have the chance of both stability and changeability, we observe more interesting dynamics. It is interesting to note that life on our planet has evolved in an intermediate range of the Kelvin temperature scale. Thus, at least one example of life has emerged in physical conditions that allow for both stasis and change, which is one of the properties that we have observed for cellular automaton behavior in roughly the middle region of the λ scale.

To conclude this section, let us take a look at what the analysis using the λ parameter has accomplished. By providing a 'knob' that controls the amount of activity in a cellular automaton, we have the opportunity to catch emergent behavior *in the act of emerging*. As we raise λ a little way from 0, we begin to get isolated periodic structures. As λ is raised further, we begin to get propagating periodic structures that can interact with one another in interesting ways. With λ still higher, we begin to see interactions between the local interactions of these propagating structures, which begin to dominate the activity in the array. As λ approaches 1, the behavior we see has become so chaotic that it approaches the randomness of pure white noise. Thus, we see that, up to a point, behavior emerges *hierarchically* with increasing λ; the behavior of structures at one level providing the basis for more complex structures at

*This amounts to having crossed a critical *percolation* threshold for λ. See [27].

†The *glider* in Conway's cellular automaton game of 'life' is a familiar example of one of these propagating structures. See [11]. The 'life' transition rule has $\lambda = 0.27$.

‡In order to visualize these propagating structures, imagine that there are many people standing in a circle who will sing the two part round 'row-row-row your boat' in the following manner. Each person sings the round through only once, starting 'row-row-row...' when the person on his right starts singing 'merrily-merrily-merrily...'. If one person starts singing the round, it will propagate around the circle of singers, with at most two people engaged in singing the round at any one time. This is analagous to a two-cell propagating structure progressing around a circular, one-dimensional cellular automaton.

higher levels. After some point however, the hierarchies of complex structures break down into chaotic, seemingly disordered behavior. The emergence of these hierarchies of self-sustaining processes is of fundamental importance to the study of cellular automata.

8. Placing emergent behavior in context

If we compare Wolfram's classes of cellular automaton behavior with the spectrum of behaviors associated with λ we find the following correspondances*:

• Class 1 (limit point) behavior is found in the range $0.0 < \lambda < 0.2$.

• Class 2 (limit cycle) and class 4 (complex periodic and propagating periodic) behaviors are found in the range $0.2 < \lambda < 0.4$.

• Class 3 (chaotic) behaviors are found in the range $0.4 < \lambda < 1.0$.

Thus, we identify three regions of interest within the spectrum of behavior associated with λ:

• Region 1) Activity tends to die out. We could call this the *quiescent* region.

• Region 2) Activity tends to periodic structures, both fixed and propagating, and their localized interactions. We could call this the *balanced* region.

• Region 3) Activity tends to fully developed chaos. We could call this the *chaotic* region.

The interesting thing about these correlations is the location of class 4 behaviors. The primary difference between Wolfram's class 2 and class 4 behaviors is that, although both contain localized periodic structures, in class 4 there can be periodic structures that propagate themselves in space, whereas there are only localized periodic structures in class 2. Without the possibility of interaction, the localized periodic structures of class 2 give rise to global limit-cycle behavior. In class 4, however, the existence of propagating structures means that there can be arbitrarily complex interactions between localized periodic structures and propagating periodic structures. This means that the global behavior of a class 4 cellular automaton is potentially in the chaotic regime.

*Keeping in mind the caveats of the previous section.

It seems the Wolfram's class 4 behaviors are associated with the *onset* of chaotic behavior in cellular automata. Thus, this behavior properly lies between the limit cycle dominated behavior of class 2 and the fully developed chaotic behavior of class 3. Rather than calling them 'class 4' behaviors, perhaps there should be a sub-division of class 2 behaviors into class 2a – *limit cycle* behavior – and class 2b – *partially* developed chaotic behavior – reserving class 3 for *fully* developed chaotic behavior.

Von Neumann proved that cellular automata are capable of universal computation by showing that a universal Turing machine could be embedded in a cellular array [29]. Of his four classes of behavior, Wolfram identifies class 4 as the only class within which universal computation could take place [32]. From the point of view of computation then, systems in the 'balanced' region of the λ scale are the most likely to hold useful computational structures. For systems capable of universal computation, there will be questions that can be asked about computations that will be *undecidable* in the general case [17], questions such as 'will this computation halt or not?' Thus, the 'balanced' region represents a dynamics that, although balanced delicately between collapse to fixed point or limit cycle behavior and explosion to fully developed chaos, is a *meta-stable* dynamics. It is impossible in principle to predict in the general case whether the forces tending towards chaos or the forces tending towards quiesence will ultimately dominate the dynamics of the system or whether, for that matter, neither one will ever dominate. Indeed, for many such systems, the conflicting pulls toward order and chaos seem to provide an essential tension which keeps the ongoing dynamics on an indefinitely extended transient, far from equilibrium.

From the point of view of artificial life, the most interesting region of the spectrum of behavior associated with the λ parameter is also the 'balanced' region. It is a region characterized by the existence of relatively stable, fixed and propagating periodic structures together with their interactions, which can support an ongoing dynamics that is far from equilibrium. The behavior of these propagating periodic structures is just the kind of behavior we would expect from our artificial molecular operators, and it is because of the existence of such structures that this region shows the most promise for supporting an artificial biochemistry.

We would like to incorporate these fixed and propagating periodic structures into a 'molecular logic', involving propagating periodic structures as molecular operators with the capacity to operate on themselves as well as on fixed periodic structures. If we can do so, there is every reason to believe that such systems could support some kind of artificial life.

9. Virtual automata

We now take a closer look at the periodic, propagating structures in cellular automata whose interactions dominated the balanced region we observed in our λ analysis.

Some dynamical systems governed by nonlinear differential equations involve *solitary waves*: particle-like waves that are capable of complex interactions. A *soliton* is an example of a special kind of solitary wave that preserves its shape during interactions with other solitons. The analogues of solitary waves in cellular automata seem to be the propagating periodic structures that we have observed in our λ analysis and that distinguish Wolfram's class 4 from class 2 behaviors. It is even possible to have soliton-like propagating periodic structures that preserve their identity during interactions with one another. Thus, from the point of view of dynamical systems theory, these propagating structures are essentially periodic, solitary waves of state-change propagating through the array. Due to the discrete nature of cellular automata, these structures constitute *digital* solitary waves. From the point of view of automata theory, a propagating structure in a two-dimensional cellular automaton is essentially a *finite automaton* operating on a two-dimensional tape.

A finite automaton is an entity that consists of a finite set of states and a set of transition rules that dictate how the automaton will change its state in response to an input symbol. The transition rules map the current state and input symbol onto the next state of the automaton. The input symbols are drawn from a finite alphabet and are considered to be supplied on an input tape (although any source of input such that an input symbol arrives at each time step will suffice). A finite automaton may produce output but it cannot read its own output nor may it go back and review its past input. Thus, its memory is limited to the size of its state set.

The finite bound on the size of its memory restricts the computational power of a finite automaton. If we consider a finite automaton that is augmented so that: a) it can write onto its tape as well as read from it; b) it can move in either direction on the tape; and c) the tape can be indefinitely extended, then we have removed the restrictions on its computational power. This kind of machine is known as a *Turing machine* and it is believed that such a machine can compute anything that can be computed in principle: there are no further modifications that can be made to a Turing machine that will allow it to compute a larger class of functions [17]. There is even such a thing as a *universal* Turing machine that can simulate the computation of any simple Turing machine. It does so by being augmented by a special tape that contains the transition rule for some simple Turing machine, and then emulates that machine by looking up the requisite transitions on this special tape. This is, in essence, what confers *generality* on general purpose, stored program computers. When they are given a specific program, they are emulating the special purpose machine that is specified by the program.

If we define one of the states of a finite automaton as the *initial* or *start* state and another as the *halt* state, we can use the automaton to classify input strings into one of two classes depending on what state it is in just after it consumes the last input symbol. The automaton is said to *accept* the set of all strings that leave it in the halt state and it *rejects* all others. The (potentially infinite) set of strings accepted constitutes the *language* recognized by the automaton. Due to their greater computational power, Turing machines can recognize a wider class of languages than finite automata.

We can view the periodic propagating structures we have seen above as automata that are operating on a (potentially infinite) two-dimensional tape. They cycle through a set of states as they move, and they can mark or read the states of the cells in the array that they encounter in the course of their propagation. Whether they are most correctly viewed as finite automata or as Turing machines depends on how they operate. If they can never encounter their own previous input or output, then their memory is limited to their set of states and they will function as finite automata. If, however, they *can* encounter previous input or output, then they are potentially Turing machines*.

It is important to note that both fixed and propagating periodic structures that span more than one cell are *emergent phenomena*: they are not explicitly coded in the transition function. If we call the finite automaton that occupies every cell of a cellular array a *first-order* automaton, then periodic structures that span more than one cell can be called *second-order* automata. If we combine second order automata in interesting ways we can generate automata of even higher order. We will refer to first order automata as *physical automata* (because they are explicitly coded in the transition function) and all higher order automata as *virtual automata*.

*In the case of a finite cellular array, these Turing machines will be limited in principle to the class of *space bounded* computations. See [17].

A fixed virtual automaton always occupies the same set of physical cells, wheras a propagating virtual automaton occupies a constantly changing set of physical cells. If a propagating virtual automaton can ever encounter physical cells that it has traversed before, then it is potentially a *virtual Turing machine*; if not, then it is a *virtual finite automaton*. This distinction is complicated by the fact that it is possible that the output of a virtual automaton may *itself* propagate through the array, in which case a virtual automaton may encounter some of its own previous output even if it never retraces its path.

There are a number of observations that follow when we view periodic structures as virtual automata. In order to have a convenient name, we will refer to virtual automata as *virtual state machines* (VSM's), whether they are functioning as finite automata or as Turing machines.

- VSM's *are embedded in the very tape upon which they are operating*. Both machine and data are represented as states of the same medium: an array of cells. Thus, VSM's are both *processes* and *data* at the same time.
- Since VSM's are both processes and data at the same time, *writing on the 'tape' of the enviornment is equivalent to construction*.
- Since VSM's can 'write' the special quiescent state, they can *erase* as well as construct.
- Because VSM's could cycle into quiescence altogether, they can be *self-erasing*, which is the ultimate form of *halting*.
- Since constructed configurations can also be viewed as either data or processes, VSM's *can construct other* VSM's (plate 3). Likewise, VSM's can erase other VSM's.
- Because they are both processes and data, VSM's *can treat other* VSM's *as 'data' and read or modify their structure*.
- Because configurations can occur on all scales, VSM's *can be embedded in other* VSM's. Thus VSM's can be hierarchically composed of smaller VSM's.

Identifying these propagating structures as virtual automata provides us with a good, formal foundation upon which we might base a *logic* of interacting virtual automata. We want to view such a logic as a dynamic logic for the composition of finite automata (or Turing machines) that have the capacity to operate on each other directly.

10. Virtual automata as artificial molecular operators

From the point of view of the molecular logic of life, the VSM's of section 9 are ideal candidates for filling the roles of molecular operators. Let us look at a comparison between the roles played by the various biomolecules that we identified in section 2, and the potential behaviors for VSM's that we have pointed out above.

Catalysis. VSM's can perform arbitrary acts of construction. They can operators that recognize and interact with other operators, possibly changing or modifying their structure or function. Thus they have the capacity to function as *artificial enzymes*.

Transport. VSM's can 'transport' structures by erasing them where they encounter them and re-constructing them elsewhere in the array.

Structure. VSM's can be fixed in place and can thus constitute static 'structures'.

Regulation. VSM's can be seen as data as well as processes. Thus they can be interpreted as messages. Propagating VSM's can travel between process in the array and fixed VSM's can function as markers, being sensed by propagating VSM's.

Defense. VSM's, acting as language recognizers, could detect 'foreign' VSM's because they would not *recognize* their structure as being part of the *language* of self. This could trigger the construction of VSM's that *can* recognize the structure of the invader and whose action upon recognition would be to incapacitate it in some manner.

Information. VSM's can be 'read' by other VSM's and thus can function to store information. They can therefore be used as DNA is used: as a repository for the *descriptions* of other VSM's (or molecular operators). Some VSM's can take on the roles of the various RNA's in mediating the transcription and translation of VSM *descriptions* into operating VSM's. Others can initiate the replication of the descriptions themselves.

We must emphasize again that we are interested primarily in simulating the *functions* of biomolecules, not the biomolecules themselves. We are after a simulation of the logic of life, not of the biological 'wet-ware' in which we have found this logic to be implemented. Thus the term 'artificial life' is appropriate.

In The Sciences of the Artificial [26], Simon says:

> Artificiality connotes perceptual similarity but essential difference, resemblance from without rather than within. The artificial object imitates the real by turning the same face to the outer system... imitation is possible because distinct physical systems can be organized to exhibit nearly identical behavior... Resemblence in behavior of systems without identity of the inner systems is particularly feasible if the aspects in which we are interested arise out of the *organization* of the parts, independently of all but a few properties of the individual components.

Thus, it would appear that virtual automata have the computational capacity to fill many of the functional roles played by biomolecules in the molecular logic of life. The specification of a transition function for a cellular automaton is also the specification for an *implicit* logic of virtual automata. Thus, many of the Θ rules with λ in the balanced region *will* yield molecular logics. Some may even provide molecular logics that are sufficiently rich to support a kind of artificial life.

11. Examples of systems of virtual automata

In this section, we will look at several cellular automaton systems that illustrate various aspects of the dynamical behaviors that are possible in systems of interacting virtual automata. Each of these systems incorporates its own molecular logic, involving operators that can operate on each other in interesting ways.

We first look at two *artificial biochemistries* governed by transition functions that were produced by the methods of section 7. This method involves first generating transition functions and then observing the behaviors that they support.

Next, we look at two systems controlled by transition functions that were carefully crafted to support other kinds of 'life-like' behavior. The first of these shows an attempt to capture the dynamics of an insect colony. It demonstrates the way in which simple VSM's can interact with one another in complex ways, and suggests that one might identify systems of interacting VSM's at the level of social systems as well as at the molecular level. The second of these examples demonstrates a composite structure of VSM's that reproduces itself. We discuss the nature of the problem of self-reproduction and suggest how the process of natural selection among variants might be achieved in cellular automata.

11.1. *Artificial biochemistries*

In sections 7 and 8, we found a 'balanced' region in the spectrum of behavior associated with the λ parameter, which showed promise for supporting artificial biochemistries. This region included λ values in the approximate range of 0.2 to 0.4.

Plate 4 shows an example of one of these systems that seem so nicely balanced between quiescence and chaos ($\lambda = 0.218$). In order to maintain this balance dynamically, there must be some mechanism for self-regulation embodied in the activity of the array. Although it is not easy to tell from the static picture, there are two types of propagating structures interacting in this system. Type $\mathscr{A}$ travels to the left in the figure and produces a trail of type $\mathscr{B}$ propagating structures, which travel upwards in the array. When $\mathscr{B}$'s run into each other, they occasionally produce a type $\mathscr{A}$. When $\mathscr{B}$'s run into $\mathscr{A}$'s, both are generally annihilated. Thus, the population of type $\mathscr{A}$'s is maintained by a kind of negative feedback. A large population of $\mathscr{A}$'s produces many more $\mathscr{B}$'s, which collide with and annihilate $\mathscr{A}$'s faster than they produce them, thus reducing the population of $\mathscr{A}$'s. A small population of $\mathscr{A}$'s spread thinly provides enough space for $\mathscr{B}$'s to collide with each other and produce $\mathscr{A}$'s faster than they destroy them, thus raising the population of $\mathscr{A}$'s. This process results in the maintenance of the population of $\mathscr{A}$'s at a dynamic equilibrium. The equilibrium population density is such that there is a good chance that the population will become extinct in an array of 64×64 cells. So far, the population has not been observed to become extinct in arrays of 128×128 or larger.

Plate 5 shows another example of one of these 'balanced' systems ($\lambda = 0.218$). Structures in this system are propagating downward and also to the right in the figure. The population of downward propagating structures is maintained dynamically by mechanisms similar to those discussed above. Downward propagating structures occasionally emit a series of rightward propagating structures, which cause a burst of dynamical activity. Downward propagating structures also occasionally leave trails of regularly spaced, fixed states, resulting in the vertical dotted lines visible in the figure. As above, the population of propagating structures, though ever changing, is maintained in a dynamic equilibrium.

This dynamic maintainance of a population density of propagating structures is a common feature of systems in the 'balanced' region of the λ spectrum of behaviors. This demonstrates how easy it is to get systems of interacting virtual automata to behave in a self-regulating manner. Self-regulating systems of VSM's occur 'naturally', in a sense, in cellular automata governed by a certain class of transition functions.

11.2. *An artificial insect colony*

A common aggregate organization in nature is that of a *society*. The global behavior of a society is an emergent phenomenon, arising out of all of the local interactions of its members. From our previous discussion of cellular automata as dynamical systems, we know that complex behavior can emerge from the interaction of very simple parts. Colonies of social insects provide good subject material for the study of artificial life because they so readily exhibit complex behavior emerging from the interaction of very simple living parts.

Simon has made an interesting point about the seeming complexity of the behavior of an ant:

An ant, viewed as a behaving system, is quite simple. The apparent complexity of its behavior over time is largely a reflection of the complexity of the environment in which it finds itself *.

This is true enough for a solitary ant, as in Simon's discussion. It is quite an understatement when the environment in which the ant finds itself is largely dominated by other ants. If an ant's behavior is indeed a 'reflection of the complexity of its environment', then the behavior of an ant in an ant colony is based on reflections of reflections of reflections. E.O. Wilson has written extensively on the social insects [30, 31], and has identified many phenomena occurring at the level of the aggregate colony: phenomena such as *mass communication*, which he defines as 'the transfer, among groups, of information that a single individual could not pass to another.'

One could use an ant colony as the model for a variant form of a cellular automaton, one in which each individual cell is mobile and can move about semiautonomously. Each cell would still change state by virtue of the states of the other cells in its immediate neighborhood. Now, however, instead of consisting of a fixed set of cells, this neighborhood would consist of a constantly changing set of cells. The dynamics of the resulting *colony* automaton (a large space filled with these mobile cells) would depend on the many individual cells being driven around the territory that the colony occupies, encountering one another and engaging in complex interactions. We have seen that this is just the kind of interactive dynamics that VSM's are capable of supporting. Thus, we can simulate *artificial* ants with virtual automata in cellular arrays.

Plate 6 shows a solitary 'virtual ant' (which we will call a *vant*) residing in a quiescent background. It is roughly 'V' shaped and will travel in the direction of the apex of the 'V'.

The rules governing the motion of a *vant* are simple. *Vants* reside in an environment that consists of uniformly spaced, fixed cells that are in one of two states (either blue or yellow in the following figures). A *vant* travels in a straight line in empty space. If it encounters a blue cell, it turns right and leaves the cell colored yellow. If it encounters a yellow cell, it turns left and leaves the cell colored blue. Thus, the *vant* leaves a 'trail' wherever it goes. More accurately, a *vant* operating in such an environment can identify whether a cell has been passed over an even (blue) or an odd (yellow) number of times, zero times being even.

Even for the case of a solitary *vant*, the resulting behavior is quite complex. Plate 7 shows the path followed by a single *vant* when started in a uniform environment of even (blue) fixed cells. The complexity is due to the fact that the *vant* keeps running into its own path. Since its own past behavior serves to complicate the environment it finds itself in, its current behavior is always partially a reflection of its past behavior. Thus it is, in fact, operating as a virtual Turing machine.

Several other interesting behaviors have been discovered for solitary *vants*. Plate 8 shows a *vant* that has settled down into building a periodic, self-limited pathway. Once a *vant* enters into such a pattern, it will continue constructing it indefinitely, unless it runs into some other pattern in the array. After I demonstrated these *vants* at the Evolution, Games, and Learning conference at Los Alamos National Laboratory, Jim Propp of Berkeley [25] discovered a solitary *vant* behavior that is reminiscent of web-building (plate 9). In

*From Simon's The Sciences of the Artificial [26], p. 24.

building this 'web' the *vant* lays out several 'orbs' of the web and then very 'fussily' goes about re-positioning them before it goes on to lay out more orbs.

For interactions between many *vants*, the behavior gets much more complex. Now, each *vant* is responding, not just to its *own* behavior, but to the past behavior of all of the *vants collectively*. Thus, the *vants* can cooperate in performing tasks *en masse* that are beyond the capabilities of any one of them individually.

There are so many ways that these virtual ants can encounter one another that the transition rules have not yet been worked out for all of the possible encounters. The only encounters worked out to date involve pairs of *vants* that collide at one of the uniformly spaced environment cells. In this case, they essentially pass through each other, each responding to the environment cell as it would in the absence of the other. Thus, the *vants* interact *indirectly* through their effects on the environment rather than by directly affecting each other. Nonetheless, what has been observed has shown interesting ways in which the behaviors common to solitary *vants* can, when aggregated together, become building blocks for higher order behaviors.

Plate 10 shows how two *vants* can cooperate in building an ever expanding 'circular' trail. This behavior involves two *vants* engaging jointly in 'trail displacement', an activity that can be observed in solitary *vants*. Plate 11a shows the long term result of eight *vants* interacting with one another after being started from a symmetric initial state. After a great deal of meandering around and many interactions with one another, four of the *vants* have headed off in opposite directions, constructing self-limited pathways, while the remaining four are still involved in pairwise interactions at opposite ends of the central construction. Plate 11b shows a similar evolution from a *slightly* different starting configuration, which shows how sensitive the global behavior is to slight differences in initial conditions.

These systems of interacting *vants* exhibit some of the aggregate phenomena known from the study of insect societies. As we have seen previously for systems of interacting VSM's, macro-level behavior is highly dependent on micro-level behavior. E.O. Wilson describes an important phenomenon in insect societies that involves this sensitive dependence between levels: the *multiplier effect*:

> A small evolutionary change in the behavior pattern of individuals can be amplified into a major social effect by the expanding upward distribution of the effect into multiple facets of social life*.

Wilson provides a nice example that illustrates both the multiplier effect and another fundamental property of emergent phenomena in social organizations. The following is taken from a discussion of the way in which *M. bellicosus* termites go about building a part of their nest.

> The *Macrotermes* workers give every appearance of accomplishing their astonishing feat by means of what computer scientists call dynamic programming. As each step of the operation is completed, its result is assessed, and the precise program for the next step (out of several of many available) is chosen and activated. Thus *no termite need serve as overseer with blueprint in hand*. The opportunities for the multiplier effect to operate in the evolution of such a system are obviously very great. A *slight alteration of the termite's response to a particular structure will tend to be amplified to a much greater alteration in the final product*†.

The *vants* exhibit both of these phenomena nicely. Any alteration in the very simple rules that the *vants* obey will result in enormous changes in the 'structures' that they produce. Plate 12 shows the result of starting the same initial configuration of eight *vants* that gave rise to the structure in plate 11b, but with the small change

*Wilson [31], p. 11.
†Wilson [31], p. 12. Emphasis added.

that when a *vant* encounters a yellow cell it goes *straight*, rather than turning left, after changing the state of the cell back to the blue state. The large scale structure that results is quite different, an illustration of the multiplier effect. Furthermore, it is clear that the *vants* are not obeying any 'overseer with blueprint in hand'. The structures 'emerge' as a result of each individual *vant* choosing its next action solely on the basis of its local conditions.

When looked at as resulting from the interactions of VSM's, these two phenomena were to have been expected. Any change in the rules that govern the behavior of an automaton will change the 'function' that the automaton is computing. Thus the global function that an aggregate of automata is computing on a shared two-dimensional tape will show sensitive dependence to the nature of the individual computations of which it is composed. Furthermore, individual automata take their actions solely on the basis of their own state and the current symbol on their tape, without the benefit of any overseer program that makes reference to a 'blueprint' of the final structure.

These simple, simulated ants, whose behavior is dictated by an extremely simple set of rules can, when they act collectively, exhibit some of the phenomena exhibited by large societies of much more complex organisms. From this example, we can begin to get an idea of the *arbitrariness* of the relationship that exists between some of the complex phenomena exhibited by living systems and their underlying hardware of implementation.

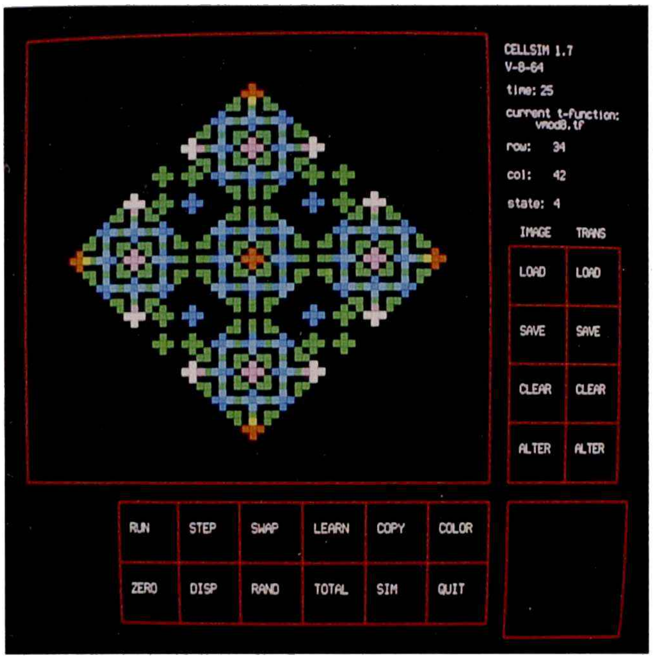
In 'reality' these behaviors are just the result of the interactions of a specific kind of VSM. The interpretation of their behavior as that of simulated ants resides entirely in our heads. Thus, this example can equally well be viewed as another example of an artificial biochemistry, based on the interactions of artificial molecular operators. By keeping the arbitrariness of our interpretations of these behaviors clearly in mind, we can see that complex, VSM-like dynamics might be possible in any system that involves the interactions of many simple parts, be they systems at the molecular level or at the level of societies.

11.3. *Artificial self-reproduction*

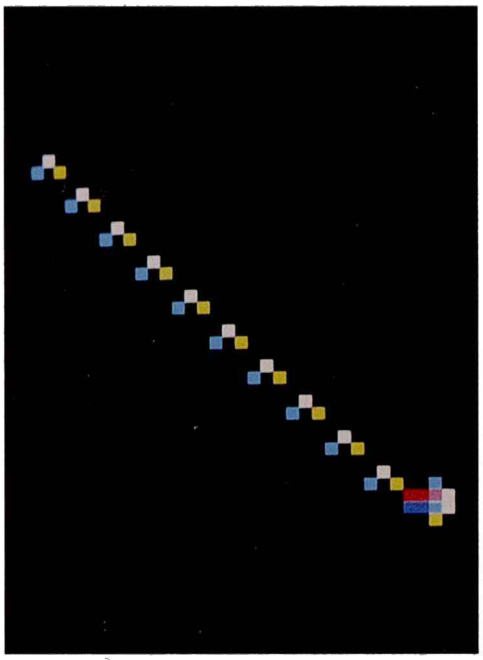
The capacity for self-reproduction is probably the most characteristic property of living organisms. John von Neumann made the first rigorous investigations into the logic of self-reproduction in the study that introduced cellular automata [29].

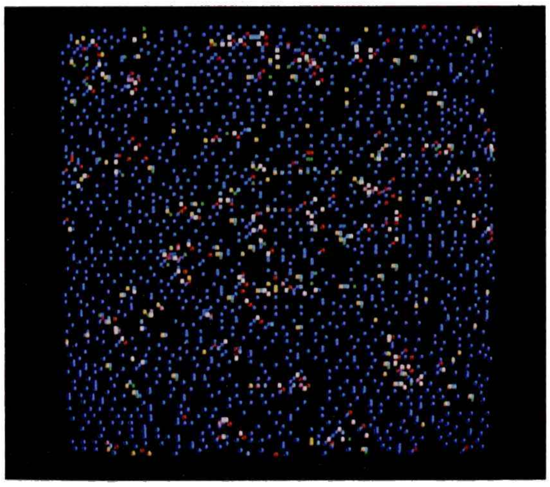
Von Neumann's approach to the problem of self-reproduction was a classically logico-mathematical one: If self-reproduction is being carried out by some kind of complex biochemical machinery, then that machinery's behavior is describable as a logical sequence of steps, i.e., as an algorithm. If an algorithm can be carried out by any machine at all, then there is a Turing machine that can implement the same algorithm. Thus, von Neumann set out to demonstrate the existence of a Turing machine that could effect its own reproduction. Since he was able to demonstrate that such a machine can exist, it becomes plausible that many, perhaps all, of the processes upon which life is based are algorithmically describable and that, therefore, life itself is achievable by machines.

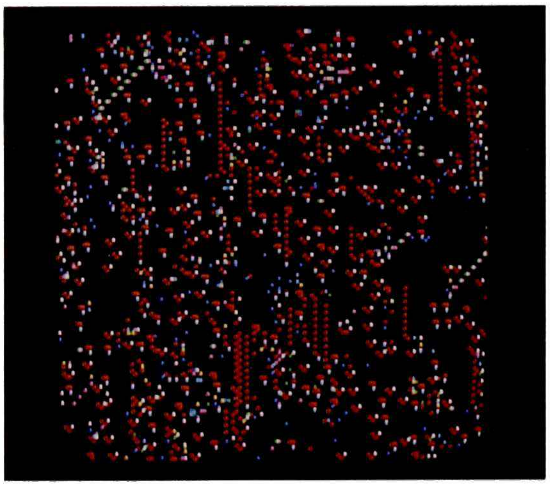
Von Neumann demonstrated that self-reproduction was a logical consequence of the existence of a certain kind of 'machine' – a *universal constructor* – and showed how such a machine could be embedded in a cellular automaton. The concept of a universal constructor is an extension of the concept of a universal Turing machine. Recall that a universal Turing machine can compute any function that can be computed by a simple Turing machine by using a description of the simple Turing machine to guide its computation. Similarly, a universal constructor can read the description of any machine from a tape and then proceed to build that machine. If such a machine is given its *own* description, it will build a copy of itself. This is not quite self-reproduction, however, because the first machine had a description of itself, whereas the constructed copy does not, and hence cannot build another copy. It does not help, by the way, to provide the initial machine with a *description* of the description of itself, in addition

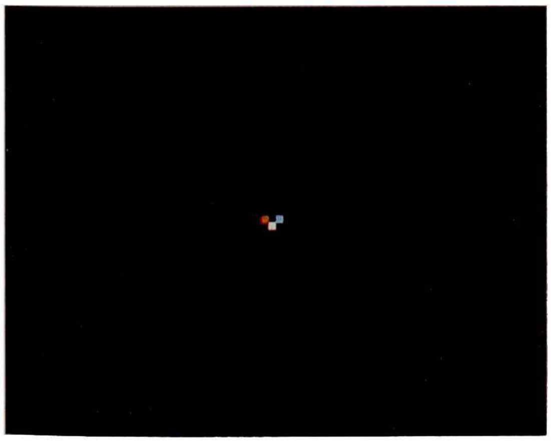

1

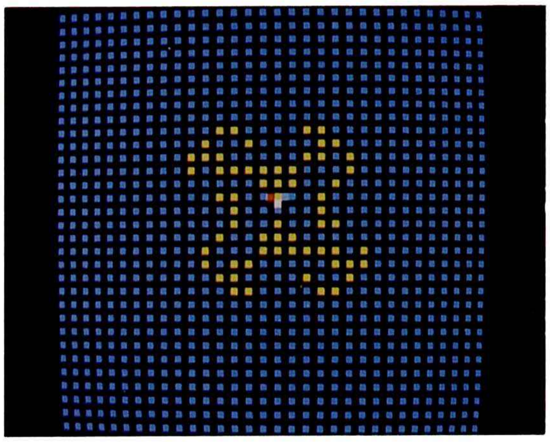
```
CELLSIM 1.7
V-8-64
time: 25
current t-function:
      vmod8.tf
row:   34
col:   42
state: 4
```


3


4


5


6


7

Plates 1–7 (for captions see text following this set of color plates).

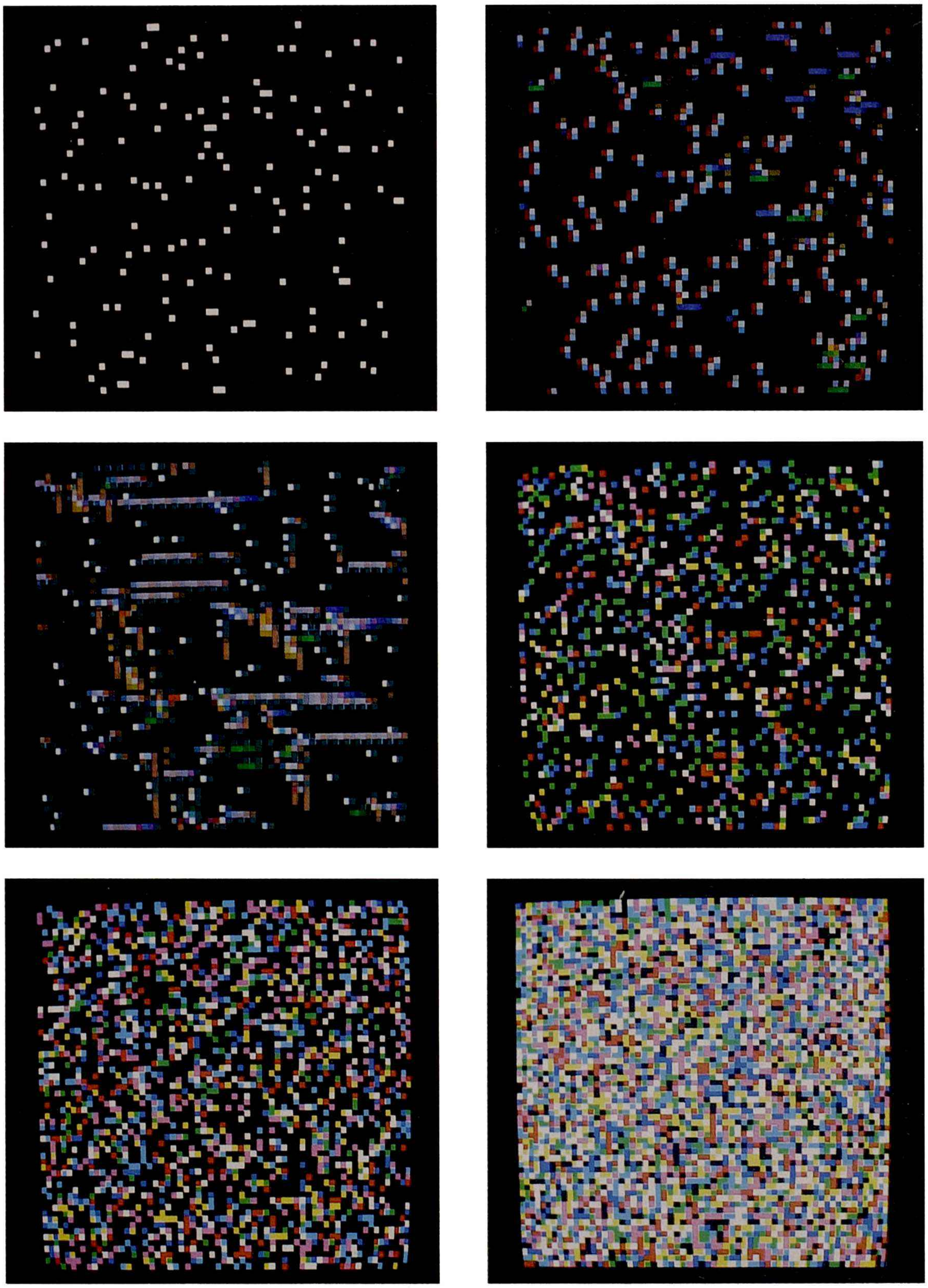
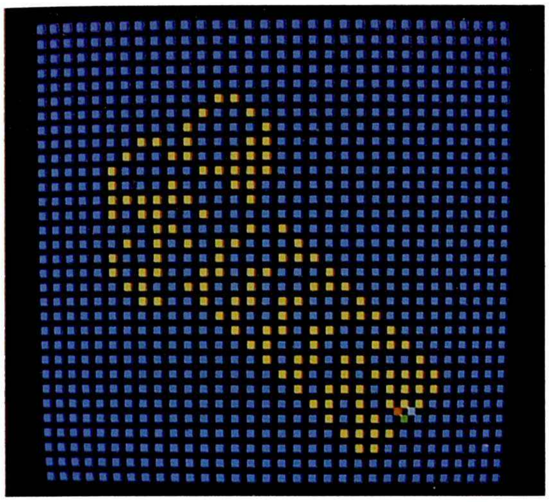
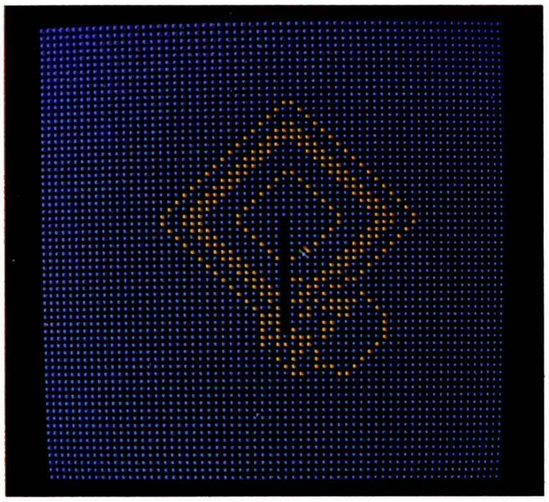
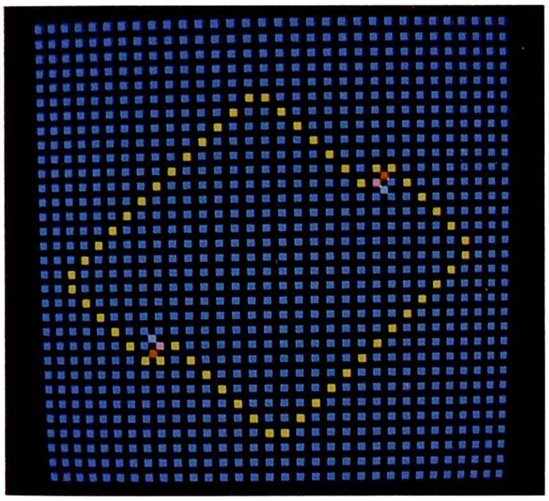
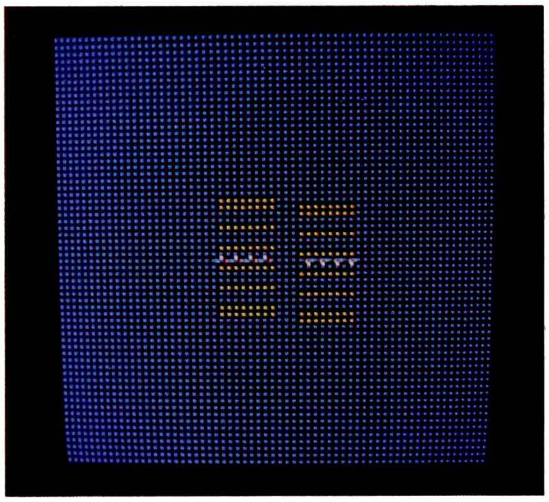


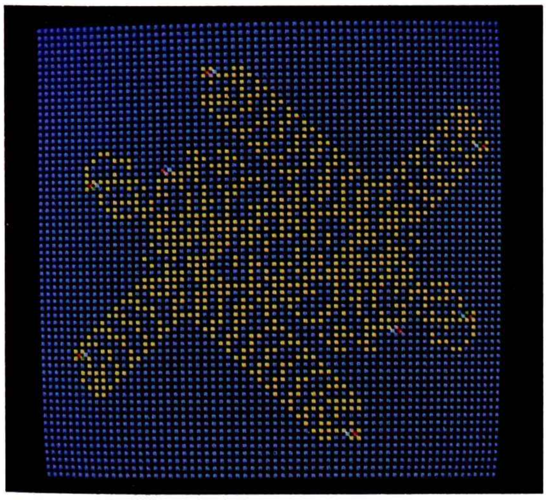
Plate 2 (for caption see text following this set of color plates).

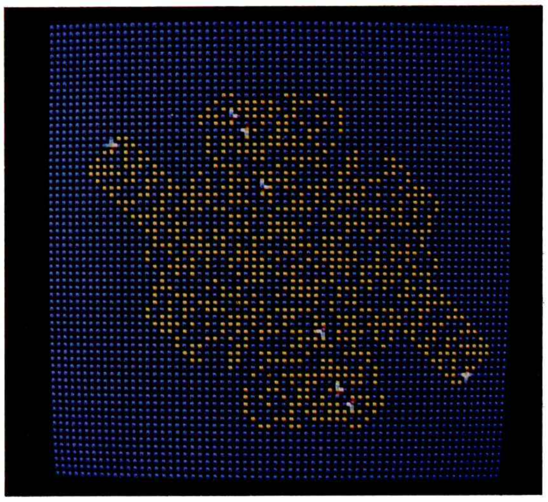

8


9


10


12


11a


11b

Plates 8–12 (for captions see text following this set of color plates).

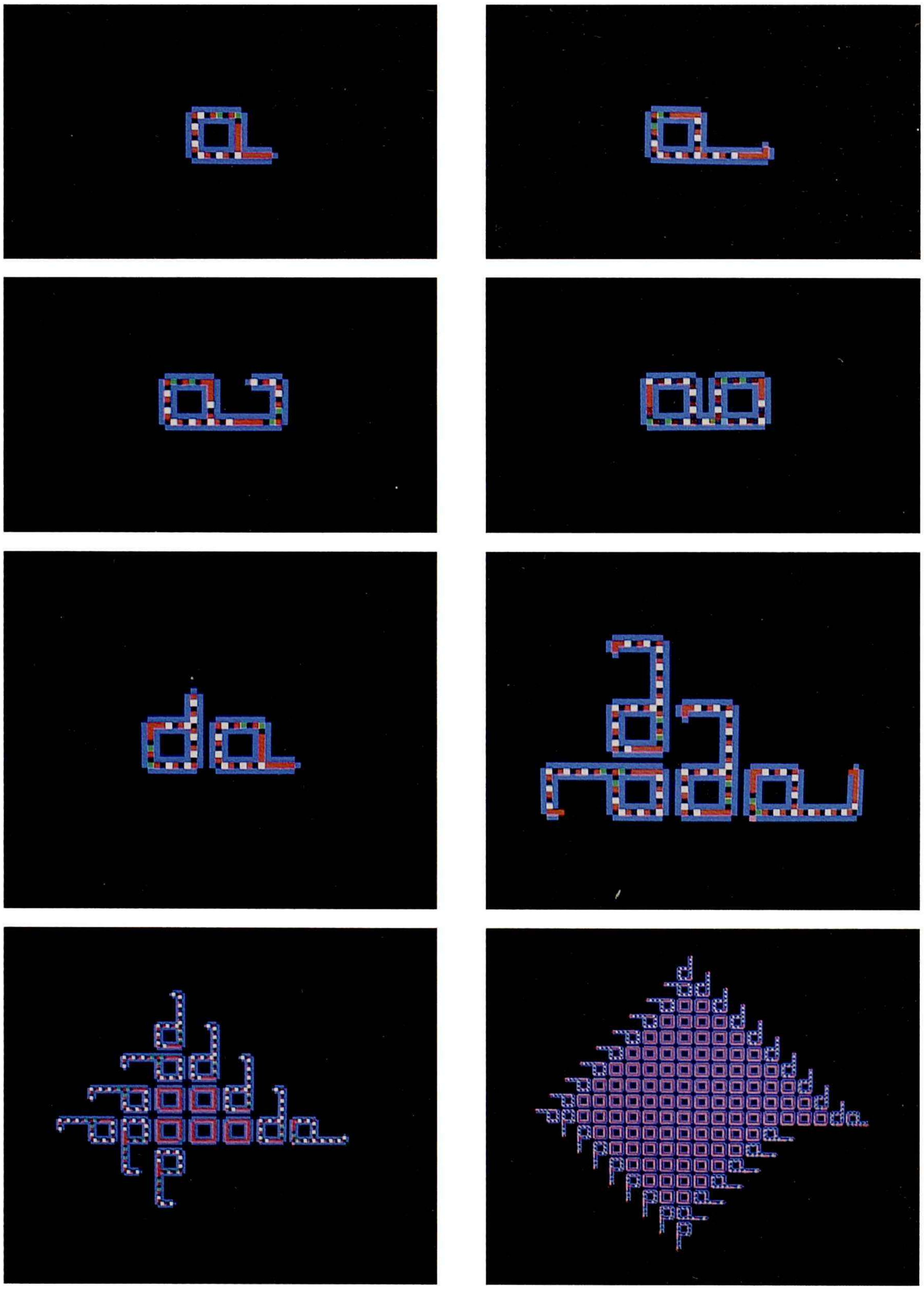


Plate 13 (for caption see text following this set of color plates).

Captions to color plates

Plate 1.	The control panel of the CELLSIM cellular simulator. The pattern displayed is a stage in the evolution of a two-dimensional cellular automaton governed by a modulo-8 addition rule over the five-cell neighborhood.
Plate 2.	Representative global configurations for various settings of λ. a) $\lambda = 0.17$; b) $\lambda = 0.19$; c) $\lambda = 0.22$; d) $\lambda = 0.33$; e) $\lambda = 0.45$; f) $\lambda = 0.86$. Figures b) and c) were 'smeared' over several time steps to convey a feeling for the dynamics of the propagating structures.
Plate 3.	A VSM propagating to the right producing other VSM's which are propagating upward.
Plate 4.	An example of an 'artificial biochemistry' ($\lambda = 0.218$).
Plate 5.	Another example of an 'artificial biochemistry' ($\lambda = 0.218$).
Plate 6.	A solitary *vant* in a quiescent background.
Plate 7.	An early stage in the travels of a single *vant* operating in an originally uniform environment.
Plate 8.	A self-limiting periodic structure, which the *vant* will continue to build indefinitely.
Plate 9.	A web-like structure built by a *vant* that initially travels up an empty tube in an otherwise regular field.
Plate 10.	An ever expanding 'circular' structure being built cooperatively by two *vants*.
Plate 11.	Two structures resulting from the interaction of eight *vants*. The *vants* in b) were started in a slightly different initial configuration from the *vants* in a).
Plate 12.	The structure resulting from the interaction of eight *vants* started from the same initial configuration as were the *vants* in fig. 11b, but with a slight change in the *vant's* behavioral responses to their environment.
Plate 13.	Several stages in the development of a colony of self-reproducing loops from a single initial loop.

to just the description of itself. This will result in the offspring machine having just the description of itself so that, although now it can construct a copy of itself, *its* offspring will not have a description of itself. Thus this strategy leads to an infinite regress.

To avoid the infinite regress, von Neumann designed a machine M that consists of three parts: a universal constructor A; a tape copier B; and a control C. This machine is then given a tape that contains a description of itself: $\Phi(M) = \Phi(A + B + C)$. Now, when M is started, it reads the description $\Phi(M)$, builds a copy of itself M', makes a copy of the description $\Phi'(M)$, and attaches the copy of the description $\Phi'(M)$ to the copy of itself M'. Thus, the machine M has reproduced itself completely, and its copy M' can go on to reproduce *itself* completely, and so forth.

The crucial property that is evident from the above discussion is that the information contained in the description on the tape must be used *twice*, in two fundamentally different ways. First, the information must be interpreted, or *translated*, as instructions for building a machine. Second, the information must be copied, or *replicated*, without interpretation, in order to provide the offspring with a copy of the description so that it too may reproduce itself.

This dual use of information is, of course, found in the process of natural self-reproduction as well. The information on the DNA is *transcribed* into messenger RNA and then *translated* into polypeptide chains at the ribosomes*. This involves *interpreting* the information as instructions for constructing a polypeptide chain. The information of the DNA is also *replicated* to form two copies of the original information. This involves merely copying the information without interpretation. Notice, however, that both uses involve construction of some form.

Laing [19] designed a clever variant of the von Neumann plan that doesn't require that a description be provided from the outside. His machine generates its own description by *self-inspection*. This makes use of the fact that the very structure of a machine can serve as its own description, given that the structure can be fully determined non-destructively.

*In a sense, ribosomes function as universal constructors.

Von Neumann worked out the details for his self-reproducing machine in a cellular space using 29 states per cell and the 5 cell neighborhood [2, 29]. Codd [5] found a simpler cellular base for von Neumann's construction by showing a self-reproducing machine that required only 8 states per cell. Both of these constructions occupy many tens of thousands of cells. As far as I know, neither construction has been implemented in an 'actual' simulation. Both of these constructions depend on the demonstration of the capacity of their machines for universal construction to make the argument go through. Hence, they need not actually be run to demonstrate that they can reproduce themselves. Self-reproduction is a simple and direct consequence, once universal construction and tape copying capacity have been demonstrated.

In a previous publication [21], I demonstrated an extremely compact structure that makes dual use of the information contained in a description to reproduce itself (plate 13). The structure consists of a looped pathway with a construction arm projecting out from it. The description consists of a sequence of VSM's that cycle around the loop. When a VSM encounters the junction between the loop body and the construction arm, it is *replicated*, with one copy propagating back around the loop again, and the other copy propagating down the construction arm, where it is *translated* as an instruction when it reaches the end of the arm. The first six of the eight VSM's that constitute the description extend the arm by one cell each. The final two VSM's cause a left-hand corner to be constructed at the end of the arm. Thus, with each cycle of the VSM sequence around the loop, another side of the offspring loop is constructed. When the construction arm eventually runs into itself, the VSM's encounter new local conditions and get interpreted in different ways. Now, they cause the connection between the parent loop and the offspring loop to be broken, and cause a new construction arm to be built in both the parent and the offspring loops. When the connection between the two loops is broken, the fragments of two separate copies of the VSM sequence that were trapped in the offspring loop get merged together to become one complete sequence. Hence the offspring loop can go on to reproduce itself.

Since this structure occupies an area of only ten by fifteen cells, its capacity for self-reproduction can be demonstrated by running it and watching it reproduce itself. Plate 13 shows several stages in the development of the *colony* that results from starting up a single loop in a cellular array. Each loop will produce at least one offspring and will then produce further copies of itself as long as it continues to find an empty site 90-degrees counter-clockwise from where it built its previous offspring. If it tries to build an offspring at a site already occupied by another loop, it will retract its construction arm and erase the cycling VSM sequence, leaving an empty loop. Thus, the colony grows indefinitely outward, consisting of a reproductive outer fringe surrounding a growing "dead" core.

Although each loop contains the same sequence of cycling VSM's, they will behave differently in different environments. The initial loop, which is surrounded by empty space, will reproduce itself four times. Its offspring will each reproduce themselves twice before they run into their parent and stop reproducing. The first offspring of each of the original loop's 'children' will reproduce itself twice, while the second offspring will reproduce only once. Thus a loop's behavior will be partially determined by features of the environment it finds itself in, the relevant environmental factor being the local concentration of other loops.

This self-reproducing structure demonstrates how VSM's can be composed to form more complex, higher order virtual automata. If we modify the behavior slightly so that instead of being erased, the VSM sequence is left cycling around the loop and is subjected to yet another interpretation, we might get the loops to interact with one another. Since the loops can respond differently to different environments, the final interpretation of the cycling instruction sequence might be different for different loops, which would mean that the loops had *differentiated*.

Because of the regular spacing of the loops, we could also have something very much like an embedded cellular automaton, where each of the higher-order 'cells' is composed of a loop covering a square of 10×10 first-order cells. If this could be achieved, then there could be much higher order VSM's embedded in this higher order cellular automaton. The point here is that given a large enough array and the right conditions, the hierarchical level at which VSM's might emerge is unbounded. Some of these levels might, in principle, be behaving very much like other levels, albeit at different time scales.

In his discussion of self-reproducing machines, von Neumann points out that one could augment the description of a self-reproducing machine with the descriptions of other 'machinery'. As long as these extra bits of machinery don't interfere with the process of self-reproduction, they will be constructed right along with the self-reproducing machine.

This suggests a couple of interesting extensions to a simple self-reproducing machine. If we complicate the task of the machines by requiring them to perform other tasks in the environment before they can reproduce, then insofar as the extra bits of machinery aid in the performance of these tasks they will be aiding in their own reproduction. Thus, they might set up a symbiotic relationship with the reproductive machine.

Furthermore, suppose that the extra bits of machinery can detach themselves from one machine, move over to another self-reproducing machine, and write their own description, possibly many times over, onto the tape that that machine will use to reproduce itself. Then one has the behavioral equivalent of a *virus* that can infect a population of machines, using the reproductive machinery of the true self-reproducers to get copies of itself made.

If we generalize from these two extensions, we have the notion of a small colony of different machines, all of which share centralized reproductive machinery, and which cooperate in performing the tasks necessary to furthering their own descriptions through the reproduction of the colony. Although the self-reproducing machine by itself might not be considered alive, this small cooperative colony seems much more like a living cell. If we could populate a large area with multiple copies of such reproducing colonies, and introduce variation into at least the portion of the description that codes for the extra machinery, we would have all of the raw material necessary for natural selection to operate among variants and hence we would have a sufficient basis for the process of evolution.

There seems to be no reason in principle why such a colony of machines cannot be implemented by a set of interacting VSM's. Thus there is reason to believe that one could implement systems of VSM's in cellular automata that would evolve over time.

12. Discussion

In this section, I wish to discuss several issues that have arisen in my thinking about systems of virtual automata and their potential for supporting artificial life.

12.1. *Virtual automata in general aggregate systems*

The emergence of dynamical systems of interacting virtual automata is a very important property of cellular automata. However, there is no reason why the emergence of such systems should be limited to cellular automata. Indeed, VSM-*like dynamics should be possible in principle in many aggregate systems*, provided that the rules controlling the individuals out of which an aggregate is built cause them to respond *selectively* to conditions in their environment. VSM-like dynamics should arise in aggregate systems in which the individuals are responsive to their environmental conditions *but not overly so*, yielding the kind of balance between action and inaction that we have observed for cellular automaton behaviors in the 'balanced' region of the λ scale.

General aggregate systems may vary in many respects from the rather specialized cellular automata. In general, the individuals of aggregate systems may not be identical to one-another, time may progress in a *continuous* flow rather than by discrete steps, individuals may update their state *asynchronously* instead of synchronously, individuals might be *mobile* as well as *sessile* (fixed in place), and so on. Since cellular automata are just one instance of a more general class of aggregate systems, it is reasonable to expect that the VSM behavior that occurs in cellular automata would belong to a more general class of such behaviors to be found in the more inclusive class of aggregate systems in general.

In an aggregate system whose individuals are mobile, as in an insect colony or a flock of birds, a VSM-like process may continue to occupy the same fixed set of individuals as it propagates*. In an aggregate whose individuals are sessile (as in a cellular automaton) the set of individuals that constitutes a VSM must be constantly changing if the VSM is to propagate. Even in an aggregate of mobile individuals, however, the set of individuals constituting a VSM could be a transient one, with some mobile individuals getting temporarily caught up in the 'spirit' of a VSM and shouldering its propagation for a while, before passing the role on to other mobile individuals†.

Thus, it would seem worthwhile to investigate the possibility that systems of interacting VSM-like processes are implicated in the dynamical behavior of many naturally occurring aggregate systems, including fluids, gases, and especially societies and brains.

*Do individual birds fly south in a flock, or does the *flock* fly south and bring its birds with it?

†Douglas Hofstadter has hypothesized that such processes are 'afoot' in ant colonies [14].

12.2. *Parallel computation*

Parallel computers are examples of aggregate systems to which the notion of interacting virtual automata might usefully be applied. As a cellular automaton *is* one kind of parallel computer, such processes are clearly possible for parallel computers.

In a standard 'von Neumann' serial computer, the functions of processing and data storage are implemented by two physically distinct structures: the CPU and the memory, respectively. Processes in such a computer reside in the memory as static data structures until one of them is summoned up from memory by the CPU, one instruction at a time, at which point it becomes semi-active, as the point of execution cycles around through the body of its constituent instructions.

In the standard approach to parallel computation, or multi-processing, many processors and memories can be operating together simultaneously, but the treatment of individual processes by each individual processor is pretty much the same as in the standard von Neumann machine: instructions are fetched one at a time from either local or shared memory. Although more finely *distributed*, the functions of processing and data storage are still handled by separate modules.

The kinds of processes we have been investigating suggest a different model for parallel computation. In this model, the functions of processing and data storage have been thoroughly intermixed, and now reside in the same simple module (e.g. the cell in a cellular automaton) which is iterated many times over. Thus, instead of lying dormant in *static* memory, processes *execute where they lie* in a *dynamic memory*. They need not await the summons of some distant and central CPU, because no such centralized, totalitarian authority exists outside of this uniform field of processing/memory modules. Furthermore, *all* of the structures residing in this dynamic memory are executing simutaneously.

Several current efforts into parallel computation involve dynamic memories (e.g. the *Connection Machine* [4, 12], and the *Boltzmann Machine* [13].) Some of these even involve a *computational temperature* which controls the ongoing level of dynamical activity in the machine in much the same way as the λ parameter [13, 18]. However, such efforts do not seem to make use of the potential inherent in an *ongoing* dynamics of interactions among *free-ranging* operators. In fact, much of the research in parallel computation goes into *eliminating* the possibility of the emergence of a complex dynamics among the various processing elements rather than encouraging it*. One notable exception to this general rule is evident in the work of John Holland at the University of Michigan. He has been actively engaged in the attempt to evoke useful dynamics from parallel systems for many years [9, 15, 16].

Note that the processes of interest in the VSM model of parallel computation are 'larger' than the individual processors (e.g. a VSM may occupy many cells). In standard models of parallel computation the *processors* are 'larger' than the *processes* of interest, many of which may reside within the local memory of each processor. In standard models of parallel computation, the processes residing in each processor work at a very *high* level of the 'dynamics' of the overall parallel computation. In the model we have been discussing, the processes residing in the individual processors work at a very *low* level in the overall dynamics. This is the difference between 'coarse-grained' and 'fine-grained' models of parallel processing. Thus, the dynamics of interacting virtual automata shows promise for forming the basis of a model for fine-grained parallel computing.

12.3. *Artificial intelligence*

Since the brain is obviously a highly parallel aggregate system, it is entirely possible that the

*This has probably been due to the seemingly sensible assumption that one either has order or one has chaos. As we have seen, however, there is a third alternative inbetween the two where order and chaos can co-exist amicably.

dynamics of this massive neural-net involves the interactions of virtual automata in some form. Based on our interpretation of VSM's as artificial molecules, we could argue that our studies of emergent behavior in cellular automata provide evidence for the proposition that intelligence is a kind of artificial life which has evolved in the neural aggregate of the brain, supported by an artificial molecular logic of interacting virtual automata. On this interpretation, the study of artificial intelligence should involve the study of the ways in which interacting systems of virtual automata embedded in highly parallel computer architectures might be applied to tasks requiring intelligence. Furthermore, on the view that learning is a form of adaptation, the study of machine learning might benefit from an understanding of how the processes of natural selection and evolution might be embedded in the dynamics of interacting virtual automata.

12.4. *Laing's artificial molecular machines*

The view that living systems have a great deal to teach us about computation was prevalent in computer science from its inception, largely due to the influences of von Neumann and Weiner, until the middle to late 1970's, when it seems to have passed from vogue, at least in the United States. In 1975, Laing [20] proposed *artificial molecular machines*, which would be dynamic 'tapes' that would interact with one another by reading and writing, and which would be both data and process simultaneously. Laing also showed how such machines might be embedded in cellular automata. It turns out that VSM's are quite similar in principle to Laing's artificial molecular machines. Although we have arrived at these structures in the course of an empirical investigation rather then by specific design, this seems to be more than a coincidence. 'Machines' similar to Laing's arise 'naturally' in certain classes of cellular automata.

Thus, these kinds of VSM's may be fundamental to unlocking the potential inherent in cellular automata for highly organized parallel processes, the kinds of processes that draw their promise from the unqualified success of similar processes implemented in a biochemical medium: the processes of life.

12.5. *Hypercycles*

The possibility that VSM's might become engaged in catalytic activity raises the question of whether these interactions themselves could become involved in the kinds of higher order cycles that Eigen and Schuster have termed *hypercycles* [7]. Hypercycles are multi-level hierarchies of cyclic catalytic reactions. Eigen and Schuster introduced their theory of hypercycles in order to provide a basis for understanding the driving forces behind pre-biotic molecular evolution.

If we analyze the conditions of hypercyclic organization we immediately see their equivalence to the prerequisites of Darwinian selection. The latter is based on self-reproduction which is a kind of linear autocatalysts. *The hypercycle is the next higher level in a hierarchy of autocatalytic systems.* It is made up of autocatalysts or reproduction cycles which are linked by cyclic catalysis, i.e. *by another superimposed autocatalysis*. Hence a hypercycle is based on non-linear (e.g. second or higher order) autocatalysis*.

It is indeed an intriguing property of cellular automata that VSM's and their interactions might provide the basis for precisely the same kinds of hierarchies of cyclic interdependencies that have been shown for catalytic reactions involving real proteins. It is further evidence pointing to the possibility that some kind of 'life' could be created within a cellular automaton.

Insofar as the theory of hypercycles correctly identifies the forces that drove pre-biotic molecular evolution towards the emergence of life, we may argue that perhaps the best way to 'create' life in cellular automata will be to rely on the

*From [7] p. vii, emphasis added.

'proven' method of providing a sufficiently rich Θ physics and initial configuration 'soup', and letting the hypercycles develop on their own, driving an analogue of pre-biotic evolution towards higher and higher levels of organization until structures emerge that must indisputably be called alive.

13. Synthesizing life

In trying to simulate the way that life emerges from the interactions of inanimate molecules, we are engaging in the process of *synthesis*, meaning 'the combining of separate elements or substances to form a coherent whole'. This is exactly the opposite of *analysis*, which means 'the separation of an intellectual or substantial whole into constituents for individual study*. Thus, synthesis is simply analysis turned bottom-up. One starts with a set of behavioral primitives and uses them as building blocks in order to discover more complex behaviors. Synthesis in abstract computer models should be seen as the general study of *emergent behavior*, and is an important field of study on its own merits, regardless of whether the individual parts or the behaviors that emerge from their aggregate interactions have direct analogues in the natural world.

By synthesizing 'life-like' behaviors in the study of artificial life, we want to try to distinguish between the relevant and irrelevant details of life's biochemical implementation in order to uncover the 'molecular logic' of life. The ultimate goal of the study of artificial life would be to create 'life' in some other medium, ideally a *virtual* medium where the essence of life has been abstracted from the details of its implementation in *any* particular hardware. We would like to build models that are so life-like that they cease to be *models* of life and become *examples* of life themselves.

In many ways, the study of artificial life is to real life what the study of artificial intelligence is to real intelligence. Each involves the study of artificial systems that exhibit behaviors normally associated with natural systems. Actually, the study of artificial life is really part of the study of natural life and the study of artificial intelligence is really part of the study of natural intelligence. However, life and intelligence as they are manifested in natural systems are too complex for direct analysis, hence the need for studying artificial models.

There are two reasons why studying life and intelligence are such difficult tasks. First, they are both *nonlinear* phenomena, in the sense that they are properties of whole systems: if we try to break them up into smaller pieces they disappear altogether. Second, all of the examples of living or intelligent organisms that are available to us for study are the result of the long process of evolution under the specific history of physical conditions on the planet Earth. This makes it very hard to distinguish the features of living or intelligent systems that are *fundamental* to life or intelligence from those that are merely present due to a combination of local historical accident and common genetic descent.

The study of artificial life might alleviate both of these complicating factors to a certain extent. First, as part of a general study into the way in which complex behavior can be *synthesized* from the interaction of many simple parts, we can hope that the study of artificial life will contribute to a better understanding of certain nonlinear systems for which *analysis* has proven impotent. Second, the synthesis of artificial living systems could extend the corpus of empirical evidence upon which biology is based, leading to a firmer foundation for a theoretical biology that will be capable of making the kind of *universal* statements that can be made in theoretical physics.

Thus, although the study of artificial life can be seen as the study of artificial systems that exhibit behaviors characteristic of natural living systems, it should not be seen solely as an attempt to simulate living systems as they occur in 'nature' as we know it. Rather, it should be seen as an

*Definitions from the American Heritage Dictionary of the English Language.

attempt to '*abstract from natural living systems their logical form.*' In this sense, it should be seen as the study of not just organic life, but of life in principle.

14. Summary

Biochemistry studies the way in which life emerges from the interaction of inanimate molecules. In this paper we have looked into the possibility that life could emerge from the interaction of inanimate *artificial* molecules. Cellular automata provide us with good artificial universes within which we can embed artificial molecules in the form of *virtual automata*. Since virtual automata have the computational capacity to fill many of the functional roles played by the primary biomolecules, there is a strong possibility that the 'molecular logic of the living state' can be embedded within cellular automata and that, therefore, artificial life is a distinct possibility within these highly parallel computer structures.

Acknowledgements

I would like to thank the following people for their valuable contributions to this effort (witting or unwitting): Rik Belew, Bob Blakely, Stephanie Forrest, John Holland, Greg Huber, Elvira Langton, Tom Libert, Bob Lindsay, Jim Propp, Paul Scott, Irene Stadnyk, and John Vidolich. Special thanks are due to Arthur W. Burks. Special thanks are also due to Doyne Farmer and Norm Packard, both for providing this forum and for many helpful comments and discussions.

References

[1] R.H. Abraham and C.D. Shaw, Dynamics – The Geometry of Behavior, Vols. I and II (Aerial Press, Santa Cruz, CA, 1984).
[2] A.W. Burks, ed., Essays on Cellular Automata (Univ. of Illinois Press, Urbana, IL, 1970).
[3] D. Campbell, J. Crutchfield, J.D. Farmer and E. Jen, "Experimental Mathematics: The Role of Computation in Nonlinear Science," Comm. ACM 28 (1985) 374–384.
[4] D.P. Christman, Programming the Connection Machine, Masters Thesis, EECS Dept., MIT, Jan. 1984.
[5] E.F. Codd, Cellular Automata (Academic Press, New York, 1968).
[6] R.L. Devaney, An Introduction to Chaotic Dynamical Systems (Benjamin–Cummings, Menlo Park, 1986).
[7] M. Eigen and P. Schuster, The Hypercycle: A Principle of Natural Self-Organization (Springer, Berlin, 1979).
[8] J.D. Farmer, T. Toffoli and S. Wolfram, eds., "Proceedings of an Inter-disciplinary Workshop on Cellular Automata", Physica 10D (1984) Nos. 1–2.
[9] S. Forrest, A Study of Parallelism in the Classifier System and its Application to Classification in KL-ONE Semantic Networks, PhD Thesis, Logic of Computers Group, University of Michigan, 1985.
[10] E. Fredkin, R. Landauer and T. Toffoli, eds., "Proceedings of a Conference on the Physics of Computation", Int. J. Theor. Physics, vol. 21, (1982) Nos. 3–4, 6–7, and 12.
[11] M. Gardner, "The Fantastic Combinations of John Conway's New Solitaire Game 'Life'," Scientific American 223 (1970) 120–123.
[12] W.D. Hillis, The Connection Machine (MIT Press, Cambridge, MA, 1986).
[13] G.F. Hinton, T.J. Sejnowski and D.H. Ackley, Boltzmann Machines: Constraint Satisfaction Networks that Learn, CMU Technical Report: CMU-CS-84-119, 1984.
[14] D. Hofstadter, Gödel, Escher, Bach: An Eternal Golden Braid (Basic Books, New York, 1979).
[15] J.H. Holland, "Escaping Brittleness," Proc. Int. Machine Learning Workshop, Urbana, IL (June 1983), pp 92–95.
[16] J.H. Holland, K.J. Holyoak, R.E. Nisbett and P. Thagard, Induction: Processes of Inference, Learning, and Discovery (Bradford Books, Cambridge, MA) (in press).
[17] J.E. Hopcroft and J.D. Ullman, Introduction to Automata Theory, Languages, and Computation (Addison–Wesley, Menlo Park, CA, (1979).
[18] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, "Optimization by Simulated Annealing", Science 220 (1983).
[19] R. Laing, "Automaton Models of Reproduction by Self-Inspection", J. theor. Biol. 66 (1977) 437–456.
[20] R. Laing, "Artificial Molecular Machines. A Rapproachment Between Kinematic and Tessellation Automata", Proceedings of the International Symposium on Uniformly Structured Automata and Logic, Tokyo, August, 1975.
[21] C.G. Langton, "Self-Reproduction in Cellular Automata," Physica 10D (1984) 135–144.
[22] A.L. Lehninger, Principles of Biochemistry (Worth, New York, 1982).
[23] J.B. Marion, Classical Dynamics of Particles and Systems (Academic Press, New York, 1970).
[24] E. Nagel and J.R. Newman, Gödel's Proof (New York Univ. Press, New York, 1958).
[25] J. Propp, personal communication.

[26] Herbert A. Simon, The Sciences of the Artificial (M.I.T. Press, Boston, 1969).
[27] H.E. Stanley, Introduction to Phase Transitions and Critical Phenomena, 2nd edition (Oxford Univ. Press, London, 1982).
[28] T. Toffoli, "Cellular Automata Mechanics", PhD Thesis, Logic of Computers Group, University of Michigan (1977).
[29] J. Von Neumann, Theory of Self-Reproducing Automata, Arthur W. Burks, ed. (Univ. of Illinois Press, Urbana, IL, 1966).
[30] E.O. Wilson, The Social Insects (Belknap/Harvard Univ. Press, Cambridge, 1971).
[31] E.O. Wilson, Sociobiology (Belknap/Harvard Univ. Press, Cambridge, 1975).
[32] S. Wolfram, "Universality and Complexity in Cellular Automata," Physica 10D (1984) 1-35.
[33] S. Wolfram, "Twenty Problems in the Theory of Cellular Automata", Proceedings of the Nobel Symposium 59: "The Physics of Chaos and Related Problems", Graftvallen, Sweden (June 11-16, 1984).