



Institut für Informatik
der Universität Zürich
Winterthurerstr. 190
CH-8057 Zürich

Artificial Intelligence Lab

Prof. Dr. Rolf Pfeifer, pfeifer@ifi.unizh.ch
Hanspeter Kunz, hkunz@ifi.unizh.ch

Fr. 8'30 – 10'00 Uhr, Institut für Informatik, Uni Irchel, Hörsaal 27-H-25

Artificial Life, SS 2000

Skript zur Vorlesung
(Rolf Pfeifer, Hanspeter Kunz, Marion Weber)

"Artificial Life is the study of man-made systems that exhibit behaviors characteristic of natural living systems. It complements the traditional biological sciences concerned with the analysis of living organisms by attempting to *synthesize* life-like behaviors within computers and other artificial media. By extending the empirical foundation upon which biology is based *beyond* the carbon-chain life that has evolved on Earth, Artificial Life can contribute to theoretical biology by locating *life-as-we-know-it* within the larger picture of *life-as-it-could-be*. (Langton)

Das Ziel dieser Vorlesung ist es, einen Überblick über das Gebiet 'Artificial Life' zu vermitteln. Sie möchte die Studenten und Studentinnen befähigen, weiterführende Literatur kritisch zu lesen und selbständige Experimente und/oder Simulationen durchzuführen.

Contents

Chapter 1:

Introduction

- 1.1 Historical origins
- 1.2 Natural and artificial life
- 1.3 Methodological issues and definitions

Chapter 2:

Pattern formation

- 2.1 Cellular automata
- 2.2 Game of life
- 2.3 Lindenmeyer systems
- 2.4 Fractals
- 2.5 Sea shells
- 2.6 Sandpiles
- 2.7 Conclusion

Chapter 3:

Distributed intelligence

- 3.1 A robot experiment: the Swiss robots
- 3.2 Collective intelligence: ants and termites
- 3.3 Simulation of distributed systems: StarLogo
- 3.4 Flocking — the BOIDS
- 3.5 Guiding heuristics for decentralized thinking

3.7 Conclusion

Chapter 4:

Applications of distributed intelligence – ant-based control

Chapter 5:

Agent-based simulation

- 5.1 The Sugarscape model
- 5.2 Emergence of structure in societies of artificial animals
- 5.3 Schelling's segregation model

Chapter 6:

Artificial Evolution

- 6.1 Introduction: Basic principles
- 6.2 Different approaches (GA, ES, GP)
- 6.3 Morphogenesis
- 6.4 Conclusion

Chapter 7:

Conclusion

Chapter 1: Introduction

The stuff of life is not stuff.

Christopher G. Langton

In this first chapter we give a brief overview over the historical origin of the relatively young field in science called Artificial Life. Besides we try to give the reader an idea of the controversial understandings of Natural Life and the comparatively straight-forward definition of Artificial Life. In the third part of this first chapter we introduce the main methodology used in Artificial Life “the synthetic approach” which can briefly be explained by the phrase “understanding by building”.

1.1 Historical origins

The branch of science named “Artificial Life” (AL) has come into being at a workshop in September 1987 at the Los Alamos National Laboratory. Named the first workshop on Artificial Life, organized by Christopher G. Langton from the Center of the Santa Fe Institute (SFI). The Santa Fe Institute (SFI) is a private, independent organization dedicated to multidisciplinary scientific research in the natural, computational, and social sciences. The driving force behind its creation in 1984 was the need to understand those complex systems that shape human life and much of our immediate world - evolution, the learning process, the immune system, the world economy. The intent is to make new tools now being developed at the frontiers of the computational sciences and in the mathematics of nonlinear dynamics more readily available for research in the applied physical, biological, and social sciences. The purpose of this workshop was to bring together the scientists working in a new and unknown niche. Langton writes:

“The workshop itself grew out of my frustration with the fragmented nature of the literature on biological modeling and simulation. For years I had prowled around libraries, shifted through computer-search results, and haunted bookstores, trying to get an overview of a field, which I sensed, existed but which did not seem to have any coherence or unity. Instead, I literally kept stumbling over interesting work almost by accident, often published in obscure journals if published at all.” (Langton, 1989, p. xv)

At this workshop 160 computer scientists, biologists, physicists, anthropologists, and other “-ists” presented mathematical models for the origin of life, self-reproducing automata, computer programs using the mechanisms of Darwinian evolution, simulations of flocking bird and schooling fish, models for the growth and development of artificial plant, and much more. During these five days it became apparent that all the participants with their previously isolated research efforts shared a remarkably similar set of problems and visions.

It became increasingly clear, that many phenomena in nature can not be described by linear models. In a linear model, the whole is the sum of its parts, and little changes in model parameters have only little effects on the behavior of the model. But many phenomena as weather, growth of plants, traffic jams, flocking of birds, stock market crashes, development of multicellular organisms, pattern formation in nature (for example on sea shells and butterflies), evolution, intelligence, and so forth resisted any linearisation; that is, no satisfying linear model was ever found.

One vision that emerged at the workshop was to look at these problems from a different angle, trying to model them as *nonlinear* phenomena. Nonlinear models can exhibit a number of features not known from linear ones: for example chaos (little changes in parameters or initial conditions can lead to qualitatively different outcomes) and the occurrence of higher level features (emergent phenomena, attractors). 'Higher level' means, that these features were not explicitly modeled. Nonlinear models have the disadvantage that they typically cannot be solved analytically, in contrast to linear models. They are investigated using computer simulations and that is the reason why nonlinear modeling is a relatively new approach. Nonlinear modeling became manageable only when fast computers were available. The fact those nonlinear models, and in AL are used almost only nonlinear models, cannot be threatened analytically has one rather unsuspected positive side effect: One does not have to be a mathematician to work with AL models.

Langton concludes:

"I think that many of us went away from that tumultuous interchange of ideas with a very similar vision, strongly based on themes such as *bottom-up* rather than *top-down* modeling, *local* rather than *global* control, *simple* rather than *complex* specifications, *emergent* rather than *pre-specified* behavior, *population* rather than *individual* simulation, and so forth.

Perhaps, however, the most fundamental idea to emerge at the workshop was the following: Artificial systems which exhibit lifelike behaviors are worthy of investigation on their own rights, whether or not we think that the processes that they mimic have played a role in the development or mechanics of life as we know it to be. Such systems can help us expand our understanding of life as it *could* be. By allowing us to view the life that has evolved here on Earth in the larger context of *possible* life, we may begin to derive a truly general theoretical biology capable of making universal statements about life wherever it may be found and whatever it may be made of". (Langton, 1989, p. xvi)

1.2 Natural and artificial life

Natural life

Preliminary remark: This topic is highly controversial and there is a lot of literature on the topic. Thus, the discussion in this section is very limited and only intended to provide an idea of some of the issues involved. Since the topic of the class is on artificial life, we should have some idea of what natural life is. We will see that there are no firm conclusions.

There is no generally accepted definition of life, although everyone has a concept of whether he or she would call a particular thing living or not. Life is a mystery to us, and simply a definition of what life is or should be would not help much to unravel this mystery anyway. Stevan Harnad, a well-known psychologist and philosopher is reluctant to give an answer:

"What is it to be 'really alive'? I'm certainly not going to be able to answer this question here, but I can suggest one thing that's *not*. It's not a matter of satisfying a definition, at least not at this time, for such a definition would have to be preceded by a true theory of life, which we do not yet have." (Harnad, 1995, p. 293)

Aristotle made the observation that a living thing can nourish itself and almost everybody would agree that the ability to reproduce is a necessary condition for life. Packard and Bedau believe that life is a property that an organism has if it is a member of a system of interacting organisms (Bedau and Packard, 1991).

In *Random House Webster's Dictionary* the following definitions for life are found. Life is

- the general condition that distinguishes organism from inorganic objects and dead organisms, being manifested by growth through metabolism, a means of reproduction, and internal regulation in response to the environment.
- the animate existence or period of animate existence of an individual.
- a corresponding state, existence, or principle of existence conceived of as belonging to the soul.
- the general or universal condition of human existence.
- any specified period of animate existence.
- the period of existence, activity, or effectiveness of something inanimate, as a machine, lease, or play.
- animation; liveliness; spirit: (example: The party was full of life).
- the force that makes or keeps something alive; the vivifying or quickening principle.

For the most part of human history, the question “What is life?” was never an issue. Before the science of physics became important, everything was alive: the stars, the rivers, the mountains, the stones, etc. So the question was of no importance. Only when the deterministic mechanics of moving bodies became dominant the question raised: If all matter follows simple physical laws, and we need no vitalistic explanation of the world's behavior, of movement in the world, then what is the difference between living and non-living things? That there is a difference is obvious, but to pin down what this difference exactly is, seems less obvious. According to Erwin Schrödinger, a famous physicist and one of the key figures in the development of quantum mechanics, it is something that cannot be explained based on the laws of physics alone. Something “extra” is required (Schrödinger, 1944). Again, what this “extra” is remains a conundrum. Still, according to Schrödinger, it can be related to the arrangements of the atoms and the interplay of these arrangements that differ in a fundamental way from those arrangements of atoms studied by physicists and chemists. Thus, it seems that Schrödinger sees the main differences in the organization of the particles rather than their intrinsic properties. This position is also endorsed by the better part of the researchers in artificial life.

Artificial Life

While natural life is very hard to precisely define, Artificial Life (AL) can be characterized in better ways. Here is the definition by Christopher Langton, the founder of the research discipline of Artificial Life:

“Artificial Life is the study of man-made systems that exhibit behaviors characteristic of natural living systems. It complements the traditional biological sciences concerned with the analysis of living organisms by attempting to *synthesize* life-like behaviors within computers and other artificial media. By extending the empirical foundation upon which biology is based *beyond* the carbon-chain life that has evolved on Earth, Artificial Life can contribute to theoretical biology by locating *life-as-we-know-it* within the larger picture of *life-as-it-could-be*. (Langton, 1989, p. 1)

In other words, the goal AL is not only to provide biological models but also to investigate general principles of life. These principles can be investigated in their own right, without necessarily having to have a direct natural equivalent. This is analogous to the field of artificial intelligence where in addition to

building models of naturally intelligent systems, general principles of intelligence are explored. Figure 1 shows the three essential goals of the field of AL. In addition to studying biological issues and abstracting principles of intelligent behavior, based on these principles, practical applications are to be developed.

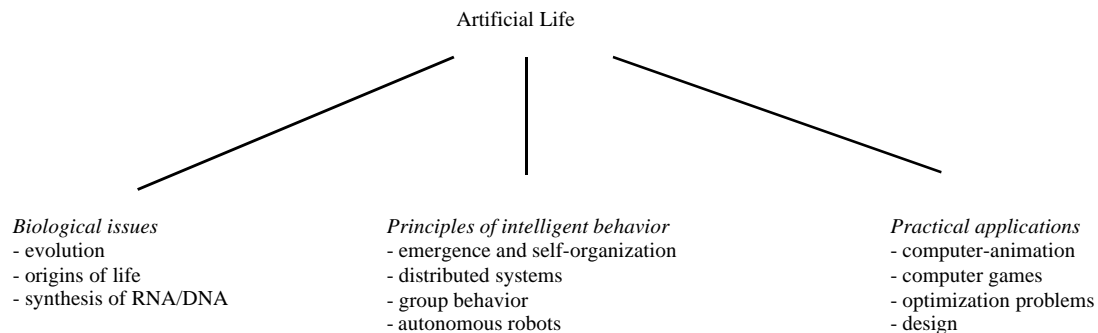


Figure 1: The goals of Artificial Life.

1.3 Methodological Issues and Basic Definitions

The Synthetic Approach

The field of AL is by definition synthetic. It works on the basis of “understanding by building”: In order to understand a phenomenon, say the food distribution in an ant society, we build aspects of the ant society’s behavior. Typically, computer simulations are employed, but sometimes researchers use robots.

Biology is the scientific study of life based on carbon-chain chemistry. AL tries to transcend this limitation to Earth bound life based on the assumption, that life is a property of the *organization* of matter, rather than a property of the matter itself. Furthermore, biology traditionally starts at the top, for example at the organism level, seeking explanations in terms of lower level entities in an analytic way, whereas AL starts at the bottom, for example at the molecular level, working its way up the hierarchy by synthesizing complex systems from many interacting simple entities. Biology works in an analytic way: Scientists are aiming to understand living beings by teasing them apart, looking for constituents, the constituents of the constituents, and so on down to cells, molecules, atoms, and elementary particles. Only recently scientists started to put these parts together again, to look how simple components can be combined to build larger systems.

Imagine, for example, that we wanted to build a model an ant colony. We would start specifying simple behavioral repertoires for the ants, and then, typically in a computer simulation, put many of these simple ants or “vants” (virtual ants) in a simulated environment. Then the vants would behave according to their (simple) rules and according to their environment. If we captured the essential spirit of ant behavior in the rules for our vants, the vants in the simulation in the simulated ant colony should behave as real ants in a real ant colony.

The analytic approach to science has been extremely successful in many disciplines like physics or chemistry. Most scientists believe that the universe is governed by laws of nature that apply for stars and galaxies as well as for elementary particles and atoms and living organisms. The question is whether —

once we know the fundamental laws — we can explain everything in these terms: Can everything, including biological systems, be reduced to these principles? There is general agreement that this is not the case and that additional — organizational — principles are required (see also Schrödinger's comments above). The synthetic methodology is particularly suited to investigate such principles.

Levels of Organization

Life, as we know it on Earth, is organized into at least four levels of structure: the *molecular* level, the *cellular* level, the *organism* level, and the *population-ecosystem* level. Of course, and fortunately, AL studies do not have to start at the lowest level. At each level behavior of the entities and their interaction can be specified and the behavior of interest then is allowed to emerge.

AL researchers have developed a variety of models at each of these levels of organization, from the molecular to the population level, sometimes even covering two or three levels in a single model. The interesting point is that at each level, entirely new properties appear. Also, at each stage new laws, concepts and generalizations are necessary, requiring inspiration and creativity to just as great a degree as in the previous one. Psychology is not applied biology and biology is not applied chemistry (Anderson, cited in Waldrop, 1992).

Time perspectives on explanation

Explanations of behavior can be given at different temporal perspectives, (1) short-term, (2) ontogenetic and learning, and (3) phylogenetic. The short-term perspective explains why a particular behavior is displayed by an agent based on its current internal and sensory-motor state (in this context the term agent, which can be understood as human, animal or artificial creature, means robot). It is concerned with the immediate cause of behavior. The second perspective, ontogenetic and learning, not only resorts to current internal state, but to some events in the more distant past in order to explain current behavior. The third, the phylogenetic one, asks how particular behaviors evolved during the history of the species. Often, an additional, non-temporal, perspective is added. One can ask what a particular behavior is for, i.e. how it contributes to the agent's overall fitness. These perspectives are closely related to what is called "the four whys" in biology (Huxley, 1942; Tinbergen, 1963). For a full explanation of a particular behavior all of these levels have to be considered.

The Frame-of-Reference Problem

Whenever we want to explain behavior we have to be aware of the frame-of-reference problem (Clancey, 1991). The frame-of-reference problem conceptualizes the relationship designer, observer, agent to be modeled, or to-be-built artifact, and environment. There are three distinct issues (Pfeifer and Scheier, 1999), perspective, behavior vs. mechanism, and complexity.

Perspective issue

We have to distinguish between the perspective of an observer looking at an agent and the perspective of the agent itself. In particular, descriptions of behavior from an observer's perspective must not be taken as the internal mechanisms underlying the described behavior of the agent.

Behavior-versus-mechanism issue

The observed behavior of an agent is always the result of a system-environment interaction. It cannot be explained on the basis of internal mechanisms only. Doing so would constitute a category error.

Complexity issue

Seemingly complex behavior does not necessarily require complex internal mechanisms. Seemingly simple behavior is not necessarily the results of simple internal mechanisms.

It is an open debate on where the description of behavior ends and where the description of the mechanism begins. Using an analytic approach we always end up with a description. If we employ the synthetic approach we not only have a description but a mechanism that actually underlies the observed behavior.

Synthetic tools

The tools of the synthetic methodology are computer simulations and robots. The field of behavior-based artificial intelligence or embodied cognitive science uses robots as modeling tools. However, in the field of artificial life, simulation is the tool of choice. Thus, for the present class we investigate mostly computer simulation.

Self-Organization

In AL the process of self-organization means the spontaneous formation of complex patterns or complex behavior emerging from the interaction of simple lower-level elements/organisms. It is an important concept and needs to be observed closely. The process of self-organization can either lead to the formation of reversible patterns (self-organization without structural changes) or to structural and therefore irreversible changes in the self-organizing system.

Emergence

The term emergence as used in AL means a property of a system as a whole not contained in any of its parts, i.e. the whole of a system being greater than the sum of its parts. Such emergent behavior results from the interaction of the elements of such system, which act following local, low-level rules. The emergent behavior of the system is often unexpected and cannot be deduced directly from the behavior of the lower-level elements.

Artificial Life and Artificial Intelligence

AL is concerned with the generation of *lifelike* behavior. The related field of Artificial Intelligence (AI) is concerned with generating *intelligent* behavior. In fact, AL and AI, at least new approaches in Artificial Intelligence have many topics in common. Mainly because AL and the new approaches in AI both work bottom-up, combining many simple elements into more complicated ones, looking for emergence and principles of self-organization, using the synthetic methodology.

In summary, AL is based on the ideas of emergence and self-organization in distributed systems with many elements that interact with each other by means of local rules.

Bibliography

- Bedau, M. A. and Packard, N. H. (1991). *Measurement of Evolutionary Activity, Teleology, and Life*. In C. G. Langton, C. Taylor, J. D. (eds.) *Artificial Life II*, Addison-Wesley.
- Clancey, W. J. (1991). *The frame of reference problem in the design of intelligent machines*. In K. van Lehn (ed.). *Architectures for intelligence*. Hillsdale, N.J.: Erlbaum.
- Harnad, S. (1995). *Levels of Functional Equivalence in Reverse Bioengineering*. In C. G. Langton (ed.): *Artificial Life, An Overview*, 293-301. MIT Press.
- Huxley, J. S. (1942). *Evolution the modern synthesis*. Allen and Unwin, London.
- Langton, C. G. (1989). *Artificial Life*. The Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems. Addison-Wesley.
- Pfeifer, R. and Scheier, C. (1999) *Understanding Intelligence*. The MIT Press, Cambridge, Massachusetts, London, England.
- Schrödinger, E. (1944). *What is Life?* Cambridge University Press.
- Tinbergen, N. (1963). *On aims and methods of ethology*. *Z. Tierpsychologie*, 20, 410-433.
- Waldrop, M. M. (1992). *Complexity, The Emerging Science at the Edge of Order and Chaos*. Simon & Schuster.

Chapter 2: Pattern formation

God used beautiful mathematics in creating the world

Paul Dirac

In chapter two we will look at some examples illustrating basic principles of pattern formation in natural and artificial systems such as cellular automata, Lindenmeyer systems (L-systems), and fractals. We will see that complex patterns can emerge from simple rules applicable to individual cells and local interactions of these cells. We will also see that the availability of many cells is a prerequisite as well as that all the rules valid for these cells are processed in parallel. The consequence of which will be that there is no need for central control.

2.1 Cellular automata

Cellular automata are examples of the large class of so-called complex systems. Complex Systems are dynamical systems that exhibit overall behavior that cannot directly be traced back to the underlying rules, that is, emergent or self-organized behavior. Complex systems typically consist of many similar, interacting, simple parts. ‘Simple’ means that the behavior of parts is easily understood, while the overall behavior of the system as a whole has no simple explanation. But often this emergent behavior has much simpler features than the detailed behavior of individual parts.

Introduction to Cellular Automata

Cellular automata (CA) are mathematical models in which space and time are discrete. Time proceeds in steps and space is represented as a lattice or array of cells (see figures 2.1 and 2.2). The dimension of this lattice is referred as the dimension of the CA. The cells have a set of properties (variables) that may change over time. The values of the variables of a specific cell at a given time are called the state of the cell and the state of all cells together form (as a vector or matrix for example) the global state or global configuration of the CA..

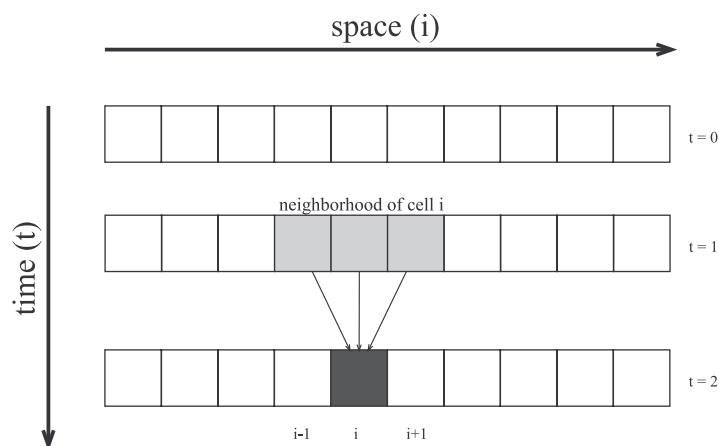


Figure 2.1: Space and time in a 1-dimensional CA.

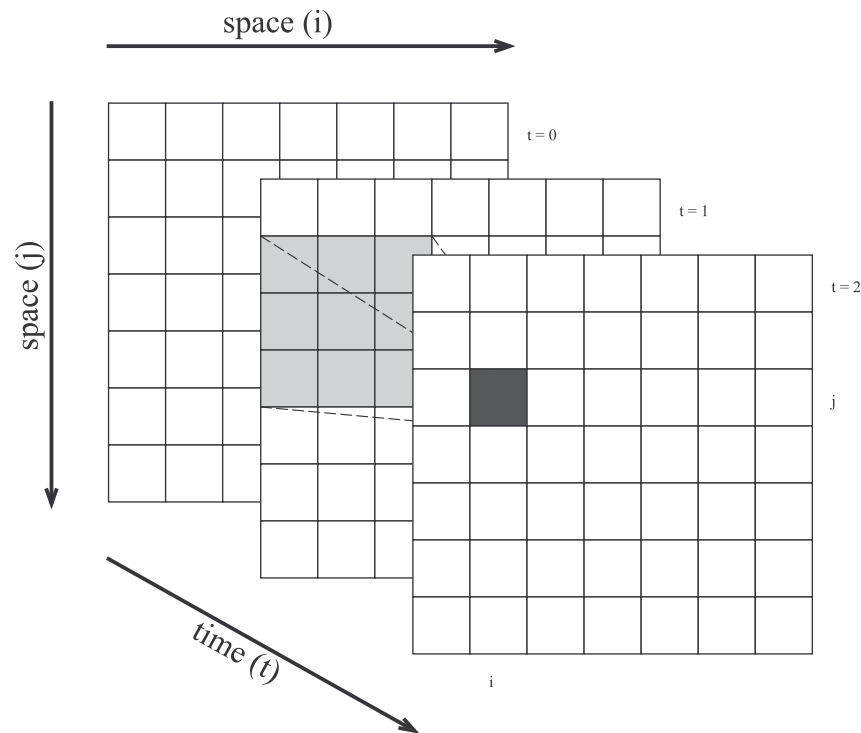


Figure 2.1: Space and time in a 2-dimensional CA.

We will consider only 1 and 2-dimensional CA. But the concept can be extended easily to any higher dimensional spaces. In the table 2.1 the mathematical notation for 1 and 2-dimensional CA is summarized. Typically the state variables have discrete values. That is, a CA is discrete in time, discrete in space and therefore perfectly suited for simulation on a computer.

Table 2.1: Mathematical notation for 1- and 2-dimensional CA.

symbol	Meaning
t	Time
Δt	time step, typically 1
$a_i(t)$	state of cell at position i at time t (1 dim.)
$a_{ij}(t)$	state of cell at position (i,j) at time t (2 dim.)
$A(t)$	global state of the CA at time t

Local Rules

Each cell has a set of local rules. Given the state of the cell and the states of the cells in its neighborhood these rules determine the state of the cell in the next time step. These rules are local in two senses: First each cell has its own set of local rules and second the future state of the cell only depends on the neighbors of this cell. It is important to note that the states of all cells are updated simultaneously (synchronously) based on the (momentary) values of the variables in their neighborhood according to the local rules. If all cells have the same set of rules the CA is called homogeneous. We will consider only homogeneous CA.

The lattice of CA can either be finite or infinite. Typically (especially if the CA is simulated on a computer) it is finite. Infinite lattices are mainly of mathematical interest. Infinite lattices have no borders, whereas on finite lattices one has to define what happens at the borders of the lattice, that is one has to define boundary conditions. The problem is that cells at the borders have only incomplete neighborhoods. There are two straightforward possibilities to solve this problem, either to assume that there are “invisible” cells next to the border-cells which are in a given predefined state or to assume that the cells on the edge are neighbors of the cells on the opposite edge as depicted in figure 2.3.

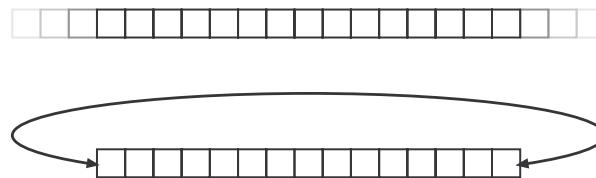


Figure 2.3: Two possibilities for boundary conditions in a 1-dimensional CA. Top: infinite (unbounded) array of cells. Bottom: finite array of cells closed to a circle. The leftmost cell becomes a neighbor of the rightmost cell.

The initial values of all the state variables are referred to as the initial conditions. Starting from these initial conditions the CA evolves in time, changing the states of the cells according to the local rules. The evolution of the CA from its initial conditions is uniquely defined by the local rules, as long as they are deterministic (we will only consider deterministic rules). Thus, CA are deterministic systems whose deterministic behavior results from local rules. Cells that are not neighbors do not directly affect each other. CA have no memory in the sense that the actual state alone (and no other previous state) determines the next state. Because the rules and the states of the cells are local any global pattern that might evolve is thus emergent.

Applications

CA have been used for a wide variety of purposes. For example for modeling nonlinear chemical systems (Greenberg et al., 1978) and the evolution of spiral galaxies (Gerola and Seiden, 1978; Schewe, 1981). In these two cases the lattice of cells in the CA corresponds directly to the physical space of the modeled system.

Any physical system satisfying (partial) differential equations may be approximated by a CA by discretisation of space, time, and state variables. Physical systems consisting of many discrete elements with local interaction are especially well suited to be modeled as CA. But also biological and social systems can often conveniently be modeled as CA.

CA and AL

CA are good examples of the paradigms of AL: complex systems made of similar (or identical) entities and local rules, parallel computation and thus local determination of behavior.

In the next few sections we will encounter some simple examples of 1-dimensional CA and explore the terms and concepts introduced. In section 2.2 we will see an example of a 2-dimensional CA.

1-dimensional Cellular Automata

Let's start with some very simple CA: a 1-dimensional CA with one variable at each cell taking only k possible values, say 0, 1, ..., $k-1$. The value of the cell at position i at time t is denoted as $a_i(t)$. We will assume that the neighborhood consists always of the r nearest neighbors on each side¹ and the cell itself, thus the neighborhood consists of $2r+1$ cells. Each cell updates its state at every time step according to some set of rules Φ , and thus

$$a_i(t+1) = \Phi[a_{i-r}(t), a_{i-r+1}(t), \dots, a_{i+r-1}(t), a_{i+r}(t)]$$

Let's look at a simple example. Assume that we have 256 cells in a row, and that each cell can take the values 0 or 1. Each cell updates its state depending on its own state and the state of its two immediate neighbors according to the following rule table:

Table 2.2: Example of a simple local rule (rule table).

$a_{i-1}(t)$	$a_i(t)$	$a_{i+1}(t)$	$a_i(t+1)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

This rule can be re written in a much more compact form using predicate calculus

$$a_i(t+1) = a_{i-1}(t) \oplus a_i(t) \oplus a_{i+1}(t)$$

where \oplus denotes the addition modulo 2 (XOR). The graphical representation shown in figure 2.4 is much more intuitive. We will assume that the row of cells of the CA is closed to a circle (see figure 2.3), thus the leftmost cell is the neighbor of the rightmost cell.

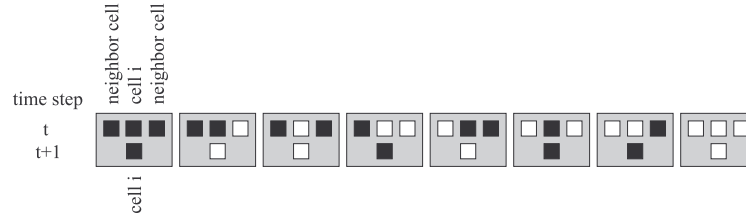


Figure 2.4: Graphical representation of a CA rule. The top row (in the gray boxes) corresponds to the configuration of a cell and its immediate neighbors. In our example there are eight possible configurations of cells (3 cells, 2 states (on, off) each). For each of these eight configurations the bottom row specifies the state of the cell in the next time step.

Figure 2.5 shows the pattern that is generated by the above rule when we start with one black cell in the middle of the CA array. (Remember that the rows correspond to the CA cells at subsequent time steps. Note the self-similarity of the patterns². Although this figure is not a fractal³ in the strict sense (because it has no infinitely fine structures) it is indeed very fractal-like. You can imagine that in an infinite CA array this pattern would grow forever, thereby generating bigger and bigger triangles, and repeat the patterns it has

¹ r: radius of neighbors

² Self-similarity at multiple levels is a key feature of fractality (see section 2.4).

generated before. A very different picture is observed when we start the same CA (with the same rules) from a random initial configuration (figure 2.6). Note that the regular pattern observed before has gone. Still, the pattern is not a random one. Triangles and other structures appear over and over again, although at irregular times and at unforeseen places.

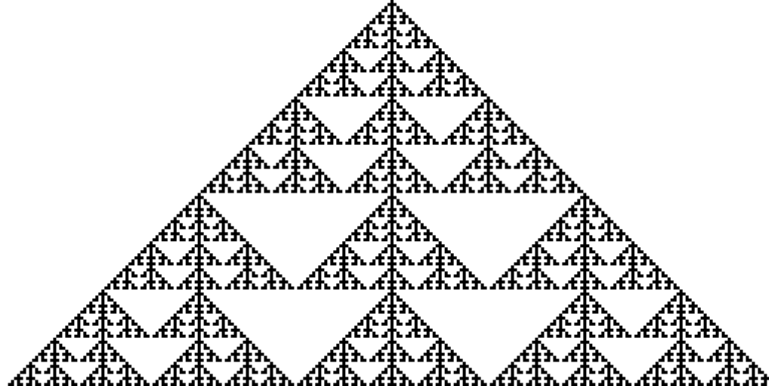


Figure 2.5: Pattern generated by 1-dimensional CA. The pattern is generated by the 1-dimensional CA rule introduced in the text. The top row corresponds to the initial configuration (one black cell in the middle) and the bottom one to the state of the CA after 128 time steps. Note the self-similarity of the patterns.

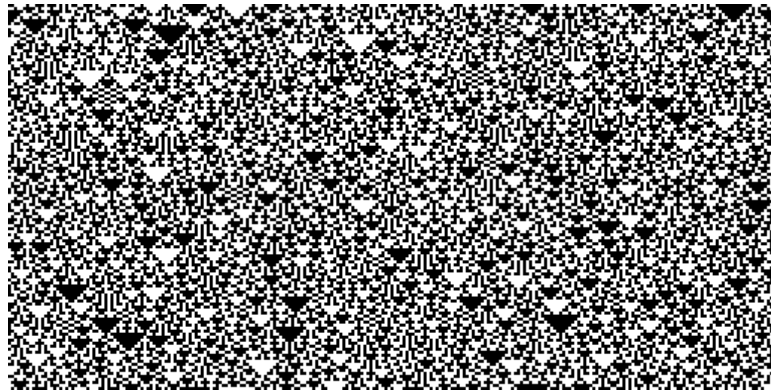


Figure 2.6: Pattern generated by a 1-dimensional CA. The same CA as in figure 2.5 is used, but in contrast to figure 2.5, the initial configuration (top row) is a random one. At first sight the patterns seems random too but at closer inspection many small structures that appear over and over again can be discerned.

We conclude from these two examples that even such simple deterministic systems as 1-dimensional CA can produce astonishingly complex patterns. These patterns are very regular if the initial configuration is regular too (figure 2.5). If the initial configuration is random the generated pattern is much less regular (figure 2.6). In both cases the patterns are complex but they reveal simple higher-level structures, the triangles. Note that by simply looking at the rules (figure 2.4) it is not at all obvious that such triangles will emerge. It is very common to find emergent structures in CA.

Number of Possible CA Rules

There are many different possibilities for CA rules. In the previous we had two states per cell, and three neighbors. Therefore there are $2^3 = 8$ entries in the rule table of a CA (the top row of figure 2.4) and thus 2^8

³ see section 2.4

= 256 possible rule tables. So there are 256 different possible cellular automata of this type. This number grows drastically if we increase the number of states k per cell and the range r of the neighborhood (or the number of neighbors $2r + 1$). For k states per cell and $2r + 1$ neighbors we have

$$k^{2r+1}$$

entries in the rule table and

$$k^{k^{2r+1}}$$

possibilities for rule tables or CA. For $k = 10$ and $r = 5$ we have 10^{11} entries in the rule table and $10^{100000000000}$ different possible CA. To put this number into perspective, there are only about 10^{80} molecules in the universe. Thus we will never be able to examine all or even a significant fraction of all possible CA.

*The Four Classes of Cellular Automata**

In this section we will consider again 1-dimensional CA with 256 cells. Each cell can take the values 0 or 1 ($k = 2$). But this time we will use neighborhoods of a varying number of cells ($r = 1$; $r = 2$, that is the cell itself and the two nearest neighbors on each side; $r = 3$). Although this types of CA are, again, very simple they exhibit a wide variety of qualitatively different phenomena. In figure 2.7 typical examples of the evolution of such cellular automata from random initial conditions are depicted. Structures of different quality and complexity are formed. Wolfram (1984a) divided the CA rules into four classes, according to what quality of structures they give rise. These are:

- Class I:** Tends to spatially homogeneous state (all cells are in the same state). Patterns disappear with time.
- Class II:** Yields a sequence of simple stable or periodic structures (endless cycle of same states). Pattern evolves to a fixed finite size.
- Class III:** Exhibits chaotic aperiodic behavior. Pattern grows indefinitely at a fixed rate.
- Class IV:** Yields complicated localized structures, some propagating. Pattern grows and contracts with time.

The classes II and III correspond to the different types of attractors (see appendix A):

- Class II:** Point attractor or periodic attractor.
- Class III:** Strange or chaotic attractor.

The four classes can also be distinguished by the effects of small changes in the initial conditions (Wolfram, 1984a):

- Class I:** No change in final state.
- Class II:** Changes only in a region of finite size.
- Class III:** Changes over a region of ever-increasing size.
- Class IV:** Irregular changes.

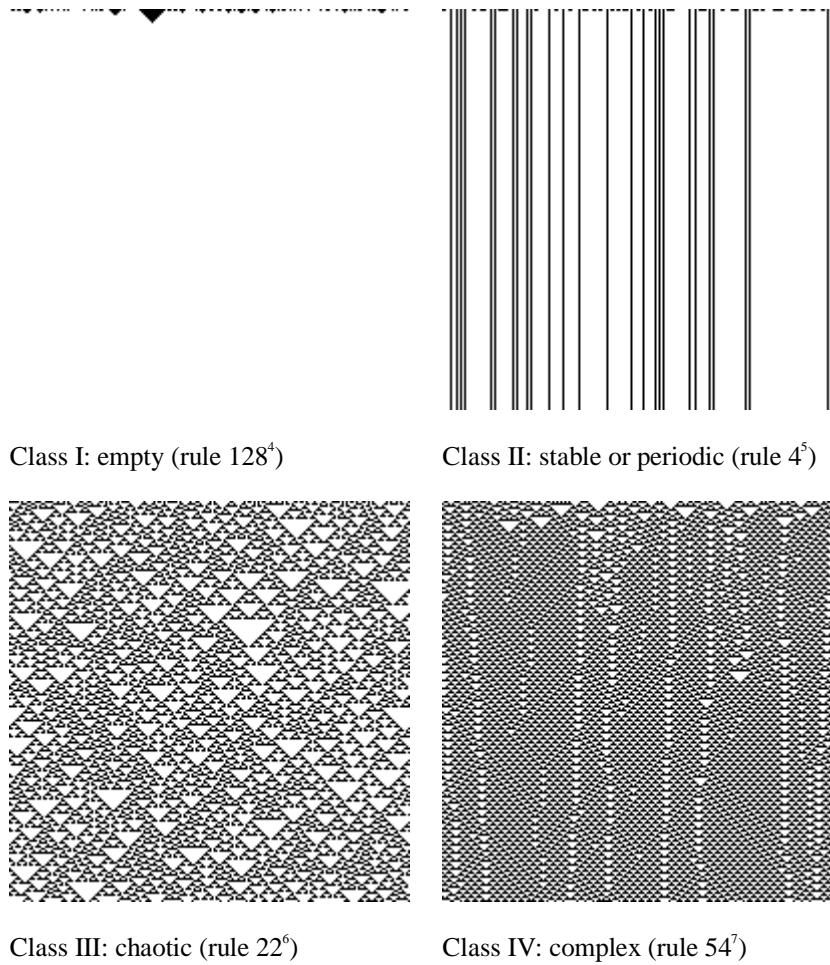


Figure 2.7: Typical examples of CA ($k = 2$, $r = 1$) starting from a random initial configuration. Depicted here are examples of the four classes of CA as introduced by Wolfram (1984a).

Figures 2.8 and 2.9 show the behavior of class IV CA. Their behavior is difficult to describe. It is not regular, not periodic, but also not random. It contains a bit of each. Class IV CA remain at the boundary between periodicity and chaos. Moreover, the behavior is not predictable without explicit calculation. That is very little information on the behavior of a class IV CA can be deduced directly from properties of its rules.

It seems likely, in fact, that the consequences of infinite evolution in many dynamical systems may not be described in finite mathematical terms, so that many questions concerning their limiting behavior are formally undecidable. Many features of the behavior of such systems may be determined effectively only by explicit simulation: no general predictions are possible. (Wolfram, 1984a, p. 23)

⁴ Rule 128: '111' goes to '1', else '0'.

⁵ Rule 4: '010' goes to '1', else '0'.

⁶ Rule 22: '001', '100', and '010' go to '1', else '0'.

⁷ Rule 54: '001', '100', '010', and '101' go to '1', else '0'.

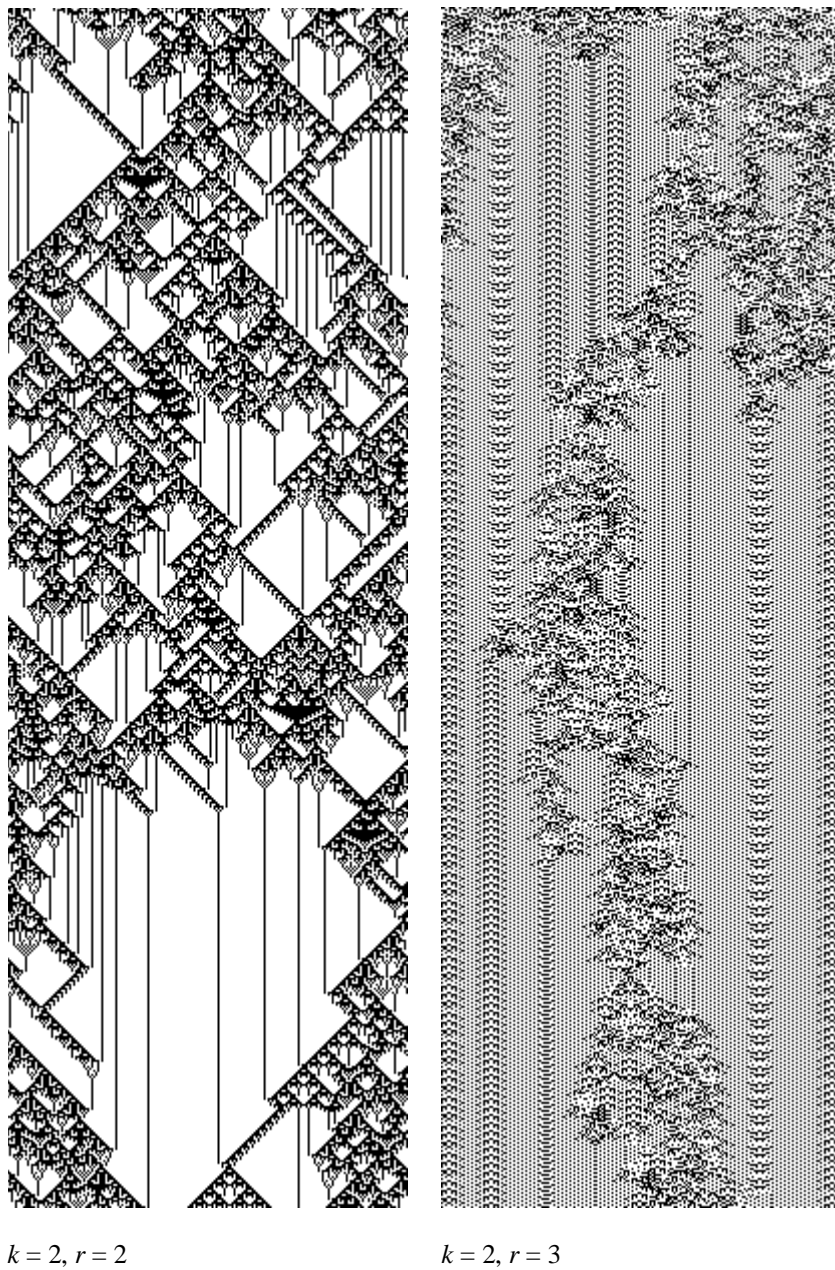


Figure 2.8: Two examples of class IV CA.

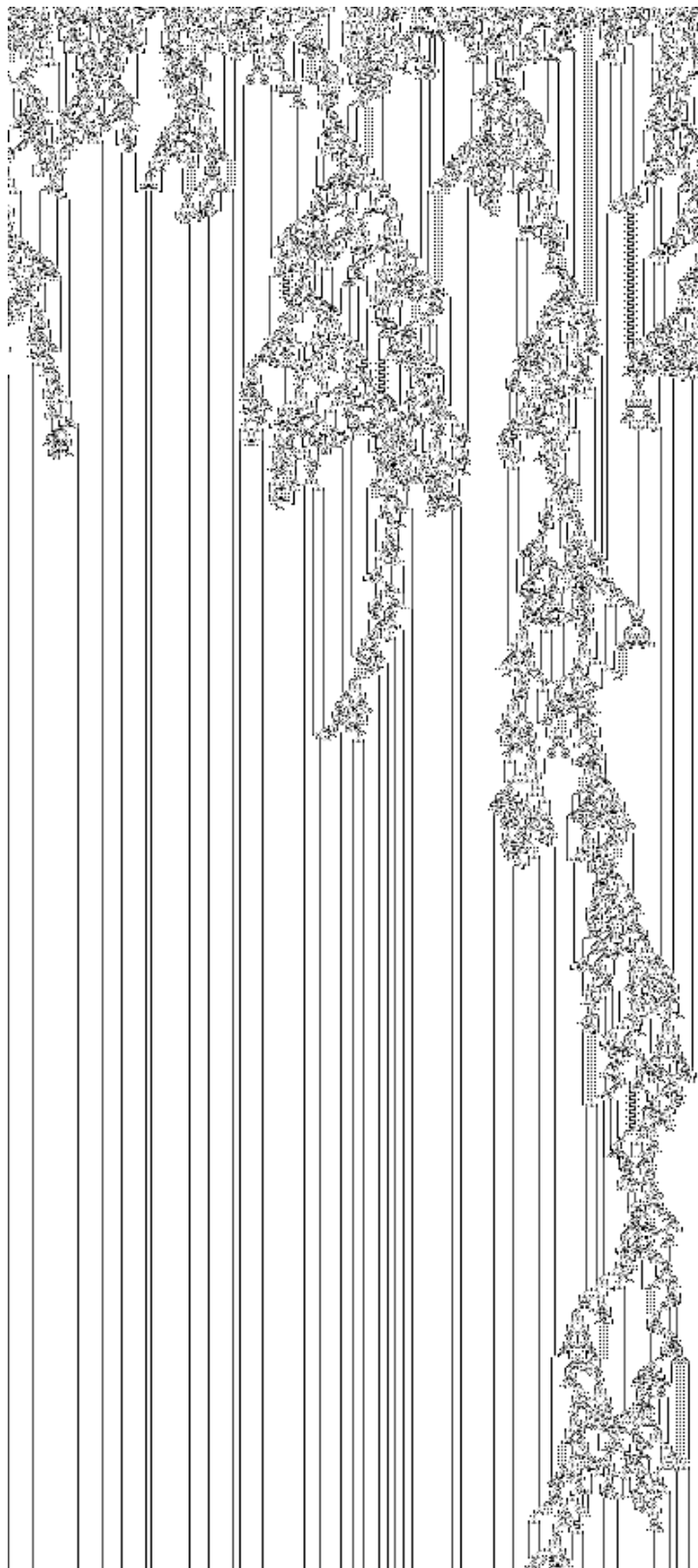


Figure 2.9: Another example of a class IV CA ($k=5$, $r=2$).

2.2 Game of Life

In the late 1960s John Conway, motivated by the work of von Neumann, used simple 2-dimensional CA which he called the “game of life”. Each cell has two possible states of each cell 0 and 1 (or “dead” and “alive”, thus the name of the CA), and a very simple set of rules (Flake, 1998):

Loneliness: If a live cell has less than two neighbors, then it dies.

Overcrowding: If a live cell has more than three neighbors, then it dies.

Reproduction: If a dead cell has three live neighbors, then it comes to life.

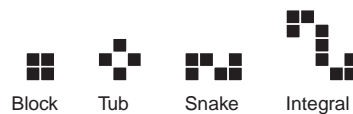
Stasis: Otherwise, a cell stays as it is.

In 1970 Martin Gardner described the Game of Life and Conway’s work in the Scientific American (Gardner, 1970). This article inspired many people around the world to experiment with Conway’s CA. Many interesting configurations were found. We will encounter some of them in the following discussion.

Patterns in the Game of Life are usually characterized by their behavior. There are several categories (of increasing complexity)⁸:

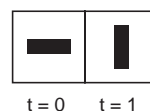
Type I (still-lives): Patterns that do not change, that are static.

Examples:



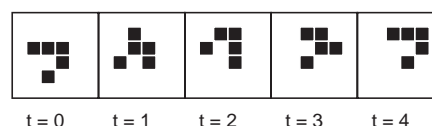
Type II (oscillators): Patterns that repeat themselves after a fixed sequence of states and return to their original state; periodic patterns.

The ‘blinker’ is an example of a 2-periodic oscillator:



Type III (spaceships): Patterns that repeat themselves after a fixed sequence of states and return to their original state, but translated in space, in patterns that move at a constant velocity.

The ‘glider’ is one simple example:

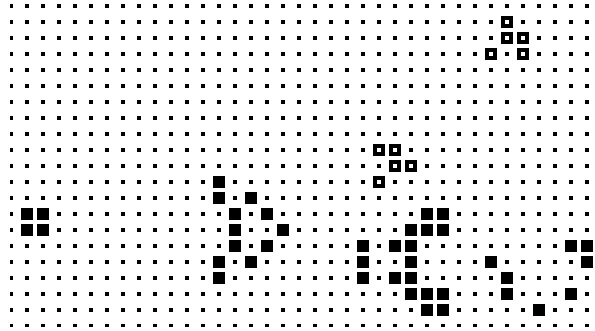


Type IV: Patterns that constantly increase in population size (living cells).

⁸ For exhaustive collection of life patterns and animations see for example home.interserv.com/~mniemiec/lifeterm.htm and www.cs.jhu.edu/~callahan/lifepage.html.

Type IVa (guns): Oscillators that emit spaceships in each cycle.

Example: A glider gun (black squares) that emits gliders (empty squares):



Type IVb (puffers): Spaceships that leave behind still-life, oscillators, and/or spaceships.

Type IVc (breeders): Patterns that increase their population size quadratically (or even faster). For example, a breeder may be a spaceship that emits glider guns.

Type V (unstable): Patterns that evolve through a sequence of states which never return to the original state. Small patterns that last a long time before stabilizing are called “Methuselahs”.

Again, the message is that despite the simplicity of the rules, amazingly complex and sophisticated structures can emerge in the Game of Life.

Universal Computation

Universal Computation means that there is the capability of computing anything that can be computed. Such can be done by a universal computer, i.e. a computer capable of universal computation. The best known example of such a universal computer is a Turing Machine, an imaginary machine proposed in 1936 by Alan Turing, an English mathematician. A Turing Machine has a read/write head mounted to a tape of infinite length, i.e. consisting of an infinite number of cells. The action performed by the head (read, write, move forward, move backward or no action/movement) depends on the current state of the head and of the cell underneath. Due to the infinite length of the tape and the lack of any limitations regarding the number of possible states, every computable problem can be solved by the Turing machine and it is able of universal computation.

Looking again at the Game of Life from a computational point of view we can say that type I objects, static objects, can be looked at as sort of memory needed in every computer, type II objects, periodic patterns, can fulfill the task of counting or synchronizing parallel processes and type III objects which repeat themselves regularly but move in space are required for moving information in a computer, thus the Game of Life includes the basic elements necessary for a computer. Through repeated collisions of moving objects with static objects the latter get altered and increase in size, i.e. new objects get created. Such process of recursively assembling pieces to make larger and more complicated objects can be carried to the extreme of building a self-reproducing machine (Flake, 1999). Based on the above and since operations in computers are usually implemented in terms of logical primitives (AND, OR, NOT) we can say that it is possible to build a general-purpose computer in the Game of Life and that it can emulate any Turing

machine. As a consequence the Game of Life is unpredictable (in the same sense as the Class IV CA of Wolfram, see above)

There are important limitations on predictions, which may be made for the behavior of systems capable of universal computation. The behavior of such systems may in general be determined in detail essentially only by explicit simulation. [...] No finite algorithm or procedure may be devised capable of predicting detailed behavior in a computationally universal system. Hence, for example, no general finite algorithm can predict whether a particular initial configuration in a computationally universal cellular automaton will evolve to the null configuration after a finite time, or will generate persistent structures, so that sites with nonzero values will exist at arbitrarily large times. (This is analogous to the insolubility of the halting problem for universal Turing machines [see for example Beckmann, 1980].) (Wolfram, 1984b, p. 31)

Another way in which sophisticated structures can emerge from systems of simple rules is the so-called Lindenmeyer Systems.

2.3 Lindenmeyer systems

In 1968 the biologist Aristid Lindenmeyer invented a mathematical formalism for modeling the growth of plants. This formalism, known as Lindenmeyer system or L-system, is essentially a traditional production system. Production, or rewriting rules, are rules which state how new symbols, or cells grow from old symbols, or cells. A production system as a whole states how at each time step its production rules get applied to symbols in such a way that as many old symbols as possible are simultaneously substituted by new symbols.

Consider the following L-system as a simple example:

Axiom: B (starting cell or starting seed of the L-system)

Rule 1: $B \rightarrow F[-B] + B$

Rule 2: $F \rightarrow FF$

If we like, we can interpret the effect of the individual rules in this rule system as follows.

Axiom: Initially, we start with a lone B -cell (see figure 2.10).

Rule 1: Each B cell divides, producing an F cell and two B cells arranged as depicted in figure 2.10. The brackets and the “+” and “-” signs indicate the arrangement of the cells (“+” rotate right, “-” rotate left).

Rule 2: Each F cell divides, producing two F cells arranged as shown in figure 2.10.

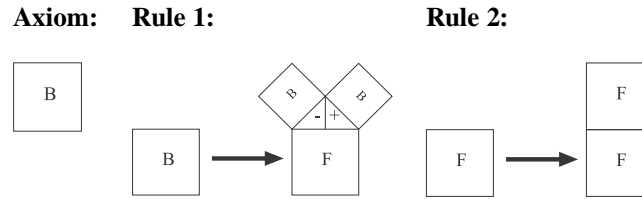


Figure 2.10: Effect of individual rules.

Note that the rules are applied in parallel, that is, all possible deductions (i.e. all possible applications of the rules) are performed simultaneously. Now let's look at the strings produced by the above production system. The axiom tells us to start with a single B cell. Therefore the initial string is simply

$$B$$

Now we apply the rules to this string and obtain (only Rule 1 matches)

$$F[-B] + B$$

Note that the rules of the L-system are used as substitution rules. In the next step both rules match and are applied resulting in the following string:

$$FF[-F[-B] + B] + F[-B] + B$$

As can be seen, the length of the string grows dramatically and gets increasingly confusing:

$$FFFF[-FF[-F[-B] + B] + F[-B] + B] + FF[-F[-B] + B] + F[-B] + B$$

Let us perform one more step:

$$FFFFFFFF[-FFFF[-FF[-F[-B] + B] + F[-B] + B] + FF[-F[-B] + B] + F[-B] + B] + FFFF[-FF[-F[-B] + B] + F[-B] + B] + FF[-F[-B] + B] + F[-B] + B$$

Much more intuitive is the graphical representation of the same process as depicted in figure 2.11.

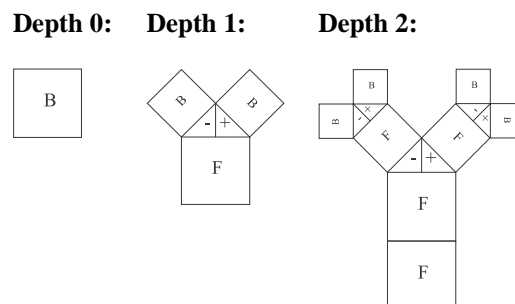


Figure 2.11: Effect of joint action of rule system. What emerges is a kind of tree structure.

Turtle Graphics

L-systems in and of themselves do not generate any images, they merely produce large sequences of symbols. In order to obtain a picture, these strings have to be interpreted in some way. In figures 2.10 and 2.11 we have already seen a possibility. More generally, these L-systems can be interpreted by turtle graphics.

The concept of *turtle graphics* originated with Seymour Papert. Intended originally as a simple computer language for children to draw pictures (LOGO), a modified turtle graphics language is well suited for drawing L-systems. Plotting is performed by a (virtual) turtle. The turtle sits at some position looking in some direction on the computer screen and can move forward, either with or without drawing a line, and turn left or right. A brief summary of commands used for drawing L-systems is given in table 2.3. Note that without the brackets the drawing of branching structures were impossible.

Table 2.3: Turtle graphics commands

command	turtle action
<i>F</i>	draw forward (for a fixed length)
<i>/</i>	draw forward (for a length computed from the execution depth)
<i>G</i>	go forward (for a fixed length)
<i>+</i>	turn right (by a fixed angle)
<i>-</i>	turn left (by a fixed angle)
<i>[</i>	save the turtle's current position and angle for later use
<i>]</i>	restore the turtle's position and orientation saved by the most recently applied <i>[</i> command

Figure 2.12 shows the first five stages of the draw process of the following L-system:

Axiom: *F*

Rule: *F*=[*-F*][*+F*]

Angle: 20.

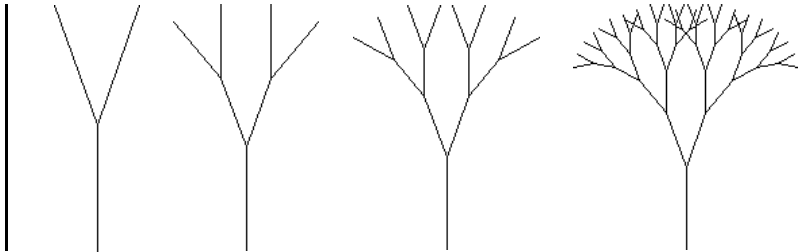


Figure 2.12: The first five L-system stages

The first drawing has an execution depth 0 and the drawing corresponds to the string

$$F$$

The execution depth denotes the number of times the rule is applied. The above string corresponding to execution depth 0 is therefore the axiom. In the second drawing, the rule is applied once, thus corresponds to the string

$$/[-F] [+F]$$

and the third drawing has execution depth 2 resulting in the string

$$/ [-/ [-F] [+F]] [+ / [-F] [+F]]$$

In figure 2.13 two examples of L-systems are given. A sample program to generate these images can be downloaded from mitpress.mit.edu/books/FLAOH/cbnhtml/index.html.

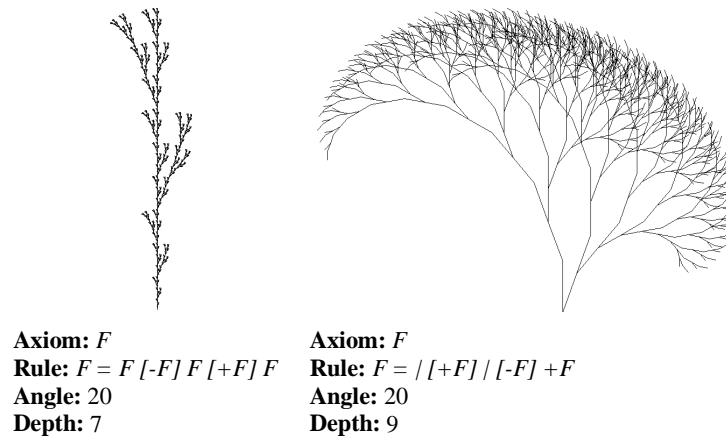


Figure 2.13: A few examples of L-systems (from Flake, 1998).

Development Models

The interpretation of the L-system can be extended to three dimensions by adding a third dimension to the orientation of the turtle. In order to simulate the development of plants additional information has to be included into the production rules. Also here the assumption is applied that plants control the important aspects of their own growth. Such information can include a delay mechanism or the influence of environmental factors but also a stochastic element, so that not all the plants look the same. Some examples of such more complex models of development are depicted in figures 2.14 and 2.15 (from Prusinkiewicz, 1990). Additional information on these can be found on the really beautiful web site www.cpsc.ucalgary.ca/projects/bmv/vmm-deluxe/TitlePage.html. We will discuss additional models of how shapes can grow when we discuss artificial evolution in chapter 6.



Figure 2.14: A sophisticated plan (a mint) grown with L-systems



Figure 2.15: Simulated development of *Capsella bursapastoris*.

2.4 Fractals

The simple L-systems we met in the previous section are instances of the more general structures known as *fractals*. The term “fractal”⁹ was first used by Benoit Mandelbrot (see, for example, Mandelbrot, 1983). Fractals are geometric figures that have one striking quality: They are self-similar on multiple scales, that means a small portion of a fractal often looks similar to the whole object (in theory a fractal is perfectly regular and has infinitely fine structures). A description of a fractal-like object could be something like this: “It has a miniature version of itself embedded inside it, but the smaller version is slightly rotated.” For example, one branch of a particular L-system plant looks exactly like the whole plant, only smaller (e.g. figure 2.18). To be precise, this perfect self-similarity of L-systems holds only if the L-system is calculated to infinite depth, or to infinitely fine details. In this case the somewhat paradoxical statement holds that an arbitrary branch of the L-system plant is exactly the same as the whole plant, only rotated and scaled. In other words, a fractal contains itself. Not only that, a fractal consists of infinitely many copies of itself.

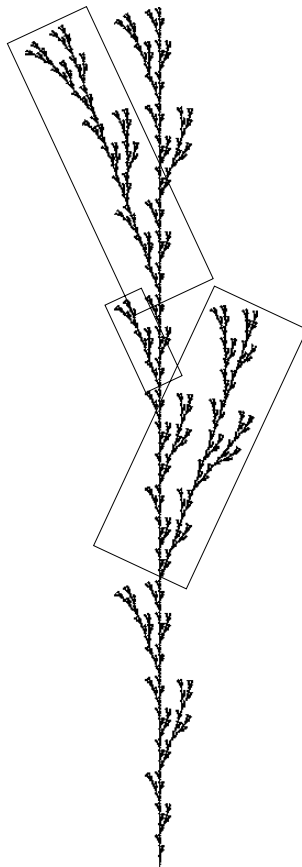


Figure 2.18: The fractal structure of L-system turtle graphics. Each branch in the boxes contains a rotated and re-scaled copy of the whole figure.

⁹ The name fractal has been given based on the fractal dimension of these structures. A fractal can have a non-integer dimension meaning for example that it is “more than a line but less than a plane”.

Fractals in Nature

Fractals often appear in nature. Not only plants like trees or ferns (see figure 2.19) have a fractal structure, but also snow flakes, and blood vessels (see figure 2.20).



Figure 2.19: A fractal fern (from www.mhri.edu.au/~pdb/fractals/fern/)

Fractals are nature's answer to hard "optimization" problems, how to find the optimal solution if there are conflicting goals. In case of the blood vessel system that hard task is to supply every part of the body with blood using as few resources as possible and in the same time minimizing the amount of time used for a single round trip. (Without this last condition one thin long blood vessel visiting every part of the body would do the job.) Because blood vessel systems are the result of millions of years of evolution one may think that they are not just any solution to the problem but a good one, one that is close to an optimal one.

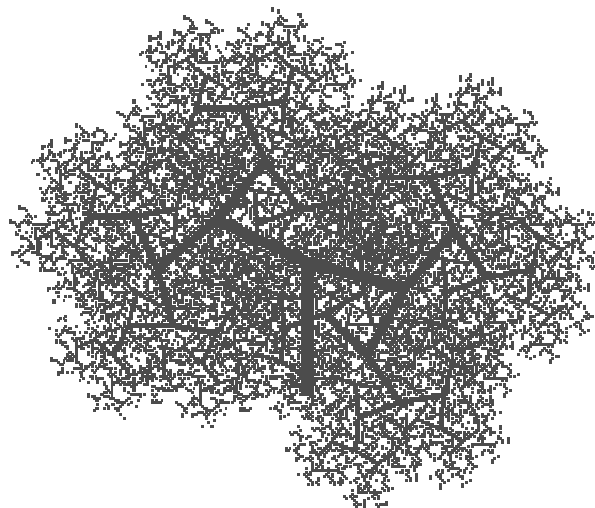


Figure 2.20: A fractal model of the blood vessel system (from www.cs.ioc.ee/ioc/res98/fractal.html).

Of course, fractals in nature are not perfect mathematical fractals, they have no infinitely fine structures and are not perfectly regular. Blood vessels, for example, do not become indefinitely small; there is some minimal diameter. Interestingly, the smallest blood vessels, the capillaries, are always of about the same

size. For example the capillaries of an elephant have the same diameter as those of a mouse. The difference is that the elephant's blood vessel system has a few more branching levels than that of the mouse.

Generating fractals

Above we noted the self-similarity, the fact that fractals contain miniature versions of themselves. The trick in generating fractals is to come up with a way to describe where and how the miniature version of the whole should be placed. In the previous section we have used L-systems and turtle graphics. In general, there are four types of transformations that one could imagine as being useful: translation (move to different place), scaling (alter size), reflection, and rotation. Algorithms for generating fractals are always recursive and based on self-similarity, using combinations of these basic four transformations. An example is the Multiple Reduction Copy Machine Algorithm (MRCM). Figure 2.21 shows a schematic representation of the MRCM algorithm¹⁰.

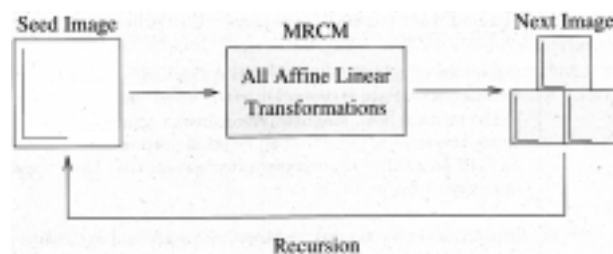


Figure 2.21: A schematic of the MRCM algorithm. The input image is simultaneously transformed by translation and scaling.

There is a vast literature on fractals. It would be beyond the scope of this class to provide extensive coverage. The interested reader is referred to Flake (1998), Chapter 7 (Affine Transformation Fractals), Barnsley (1988), Mandelbrot (1983), and Peitgen et al. (1992).

2.5 Sea shells

Another fascinating case of pattern formation that can be conveniently described by cellular automata is the evolution of the colorful patterns of seashells. We all know the pigment patterns of tropical seashells and are impressed by their beauty and diversity. The fascination comes from their mixture of regularity and irregularity (see figure 2.23). No two shells are identical but we can immediately recognize similarities. The patterns on the shell resemble the patterns we met in the sections on 1-dimensional CA. And this coincidence has a deeper reason.

The patterns on seashells consist of calcified material. A mollusk can enlarge its shell only at the shell margin. Therefore, in most cases the calcified material, and thus the pigmentation patterns, is added at this margin. In this way the shell preserves a time record of the pigmentation process that took place at the its margin. This process is much like the 2-dimensional pictures of 1-dimensional CA that are a time record of

¹⁰ The MRCM algorithm's name is based on the fact that it could at least partly be simulated with a real copy machine (make simultaneously several copies of an object and alter place and size, such process to be repeated several times).

the CA dynamics. In this sense, it is straightforward to simulate pattern growth on a seashell with a 1-dimensional CA. Let us look at this idea in some detail.

As Meinhardt argues in his book “The Algorithmic Beauty of Sea Shells” (Meinhardt, 1995) the process of pattern formation in seashells can be conceived in terms of an activator-inhibitor dynamics (figure 2.22) whereby the activator causes and the inhibitor suppresses pigmentation. This dynamics is often called a reaction-diffusion dynamics. Pattern formation is the result of local self-enhancement (also called autocatalysis) and long-range inhibition.

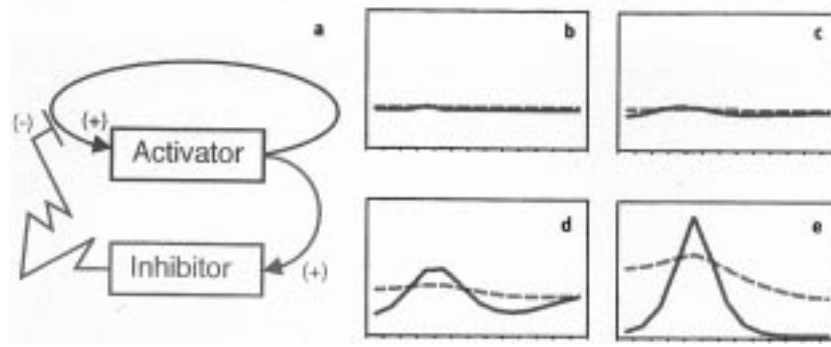


Figure 2.22: Reaction scheme for pattern formation by autocatalysis and long-range inhibition. An activator catalyzes its own production and that of its antagonist (the inhibitor). The diffusion constant of the inhibitor must be much higher than that of the activator. A homogenous distribution of both substances is unstable (b) (the x-axis represents position and the y-axis the concentration). A minute local increase of the activator (—) grows further (c, d) until a steady state is reached in which self-activation and inhibition (---) are balanced (from Meinhardt, 1995).

Activator-inhibitor dynamics can be described either by a set of partial differential equations, or by cellular automata.

Meinhardt (1995) introduces the following differential equations to describe the dynamics of the activator-inhibitor system that relate the concentration change per time unit of both substances a and b as a function of the present concentration.

$$\frac{\partial a}{\partial t} = s \left(\frac{a^2}{b} + b_a \right) - r_a a + D_a \frac{\partial^2 a}{\partial x^2}$$

$$\frac{\partial b}{\partial t} = s a^2 - r_b b + D_b \frac{\partial^2 b}{\partial x^2} + b_b$$

where $a(x,t)$ is the concentration of an auto-catalytic activator at position x at time t , $b(x,t)$ is the concentration of its antagonist, D_a and D_b are the diffusion coefficients and r_a and r_b are the decay rates of a and b .

. Let us briefly outline the main intuitions why the interaction as stated in the above equations can lead to stable patterns. Let's assume all constants, and even the inhibitor concentration, are equal to 1, and disregard diffusion. This leads to the following simplified equations:

$$\frac{\partial a}{\partial t} = a^2 - a$$

Here the activator has a steady state but only at $a = 1$ otherwise the state will be unstable. Simplifying the equation for the inhibitor b leads to

$$\frac{\partial b}{\partial t} = a^2 - b$$

Now the steady state is at $b = a^2$.

Now let us include the action of the inhibitor in the equation for the activator. Under the assumption that the inhibitor reaches the equilibrium rapidly after a change in activator concentration, this can be expressed as function of the activator concentration alone

$$\frac{\partial a}{\partial t} = \frac{a^2}{b} - a \approx \frac{a^2}{a^2} - a = 1 - a$$

The inclusion of the inhibitor leads to a steady state at $a = 1$ which remains stable since for $a > 1$, $(1-a)$ is negative and the concentration returns to $a = 1$ (Meinhardt, 1999).

As seen above the action of the inhibitor leads to stabilization of the autocatalysis and to stable patterns. On shells, stable patterns lead to permanent pigment production in some positions caused by a higher concentration of activator a and its suppression in between (higher concentration of inhibitor b). This leads, for example, to an elementary pattern of stripes parallel to the direction of growth.

The above partial differential equations which represents the continuous change over time can be approximated by a system of difference equations representing change in discrete time steps. Accordingly the discrete i will take the role of x (the position). The differentials $\frac{\partial a}{\partial t}$, $\frac{\partial a}{\partial x}$, $\frac{\partial^2 a}{\partial x^2}$ can be approximated by differences:

$$\frac{\partial a}{\partial t}(x, t) \approx a_i(t+1) - a_i(t),$$

$$\frac{\partial a}{\partial x}(x, t) \approx a_{i+1}(t) - a_i(t), \text{ and}$$

$$\frac{\partial^2 a}{\partial x^2}(x, t) \approx [a_{i+1}(t) - a_i(t)] - [a_i(t) - a_{i-1}(t)].$$

Inserted into the system of differential equations as set out above the concentration of the activator would be

$$a_i(t+1) = a_i(t) + s \left(\frac{a_i^2(t)}{b_i(t)} + b_a \right) - r_a a_i(t) + D_a (a_{i-1}(t) + a_{i+1}(t) - 2a_i(t))$$

Time is now discrete with a time step of $\Delta t=1$. If we interpret i as the number of a cell (in a row) the above equation is in fact a local rule for a 1-dimensional CA. Analogously the equation for the inhibitor $b(x, t)$ can be reformulated and we obtain a local rule for $b_i(t)$.

$$b_i(t+1) = b_i(t) + sa_i^2(t) - r_b b_i(t) + D_b(b_{i-1}(t) + b_{i+1}(t) - 2a_i(t)) + b_b$$

Therefore our resulting CA has *two* variables $a_i(t)$ and $b_i(t)$ for each cell i . The difference to the CA discussed earlier is that the state variables here can take arbitrary values and not just discrete ones.

In figure 2.23 two examples of seashells and their simulated counterparts are shown (from Meinhardt, 1995). The patterns were calculated as discussed above and the mapped onto a 3-dimensional model of a seashell. The results are striking.

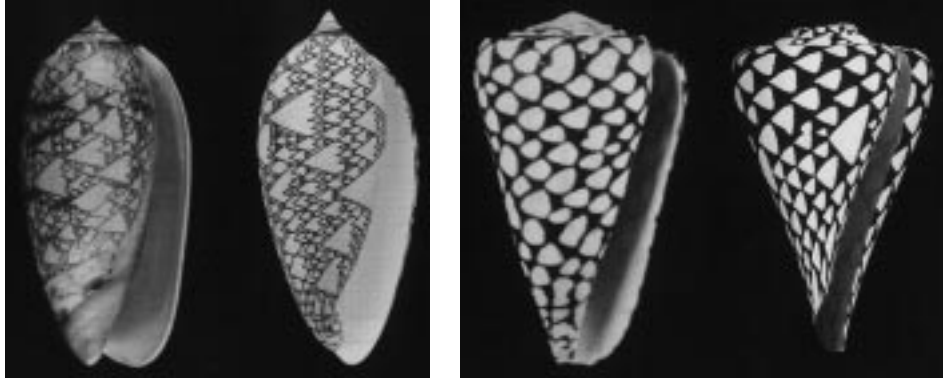


Figure 2.23: Two examples of seashells and the simulated patterns using — in essence — the dynamics described in this section (taken from Meinhardt, 1995, p. 179, 180).

2.6 Sandpiles

While studying the fundamental question why nature is so complex and not as simple as the laws of physics imply the concept of self-organized criticality (SOC), a mathematical theory describing how systems reach dynamical behavior, has been discovered (Bak, 1997). SOC explains some complex patterns that we find everywhere in nature. SOC states that nature is perpetually out of balance, but organized in a poised state — the critical state¹¹ — where anything can happen within well-defined statistical laws.

A good and easily understandable example of SOC is the sandpile model. One can imagine a flat table onto which grains of sand are added randomly one at a time. In the beginning the grains will mostly stay where they land. With more sand added grains start to pile up and sand slides and avalanches occur. First such avalanches only have local effect in one particular region of the table but with more sand added the piles cannot get any higher since the slope is too steep for additional grains of sand. Consequently the avalanches get stronger and do also affect the piles in the other regions of the table or may even cause sand to leave the table (see figure below).

¹¹ Critical in the sense that it is neither stable nor unstable, but near phase transition.

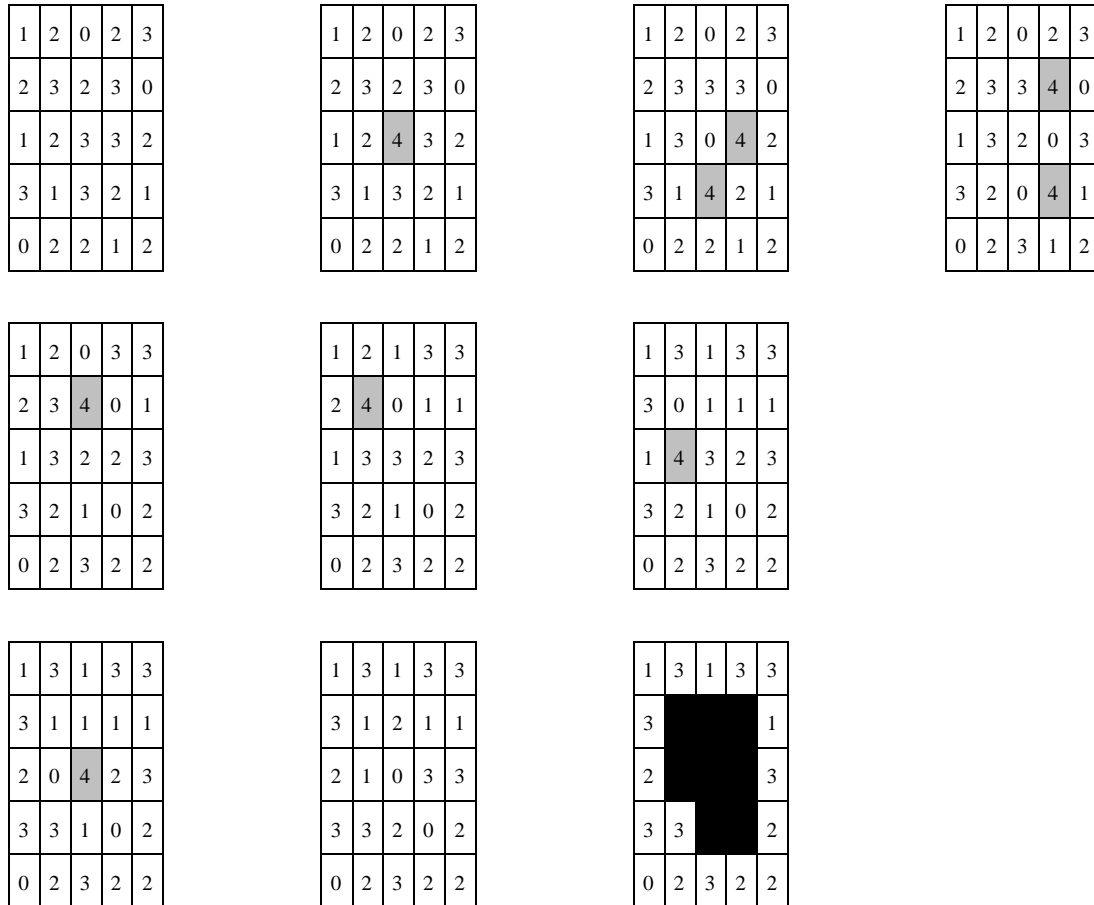


Figure 2.24: Illustrating of toppling avalanche in a small sandpile. A grain falling at the site with height 3 at the center of the grid leads to an avalanche composed of nine toppling events, with a duration of seven update steps. The avalanche has a size $s=9$. The black squares indicate the eight sites that toppled. One site toppled twice. (Bak, 1997, p.53)

In the end new grains added to the pile will result in average in the same number of grains rolling down the pile and falling off the table. In order to achieve such balance between sand added to, and sand leaving the table communication within the system is required. Such state is the self-organized critical (SOC) state.

The number of avalanches of size s can be expressed by the simple power law

$$N(s) = s^{-\tau}$$

(where the exponent τ defines the slope of the curve) and results in a quasi straight line if plotted on a double-logarithmic paper.

The power law states the following: Small avalanches appear more often than big ones.

The addition of grains of sand has transformed the system from a state in which the individual grains follow their own local dynamics to a critical state where the emergent dynamics are global (Bak, 1997). The individual elements obeying their own simple rules have through interaction lead to a unique, delicately

balanced, poised, global situation in which the motion of any element might affect any other element in the system.

Accordingly the sandpile model shows how an open system has naturally organized itself into a critical scale-free state without any external organizing force, thus a simple model for complexity in nature has been developed.

2.7 Conclusion

In this chapter we have looked at a number of examples illustrating basic principles of pattern formation in natural and artificial systems. The essence is that sophisticated patterns can emerge on the basis of simple rules that are based on local interactions. There is no need for a global blueprint. Cellular automata, Lindenmeyer systems (L-systems), fractals and SOC are convenient formalisms to model pattern formation processes.

Another central factor in pattern formation is — almost trivially — the availability of many cells, and that all the cells are processed in parallel: there must be no central control.

Bibliography

- Bak, P. (1997). *How Nature Works*. Oxford University Press
- Barnsley, M. (1988). *Fractals Everywhere*. Academic Press
- Beckmann, F. S. (1980). *Mathematical Foundations of Programming*. Addison-Wesley
- Berlekamp, E. R., J. H. Conway, R. K. Guy (1982). *What is Life?* Chapter 25 in *Winning Ways* (Volume 2), Academic Press
- Flake, G. W. (1998). *The Computational Beauty of Nature*. A Bradford Book
- Gardner, M. (1970). *Mathematical Games*. *The fantastic combinations of John Conway's new solitaire game "life"*. In *Scientific American*, vol. 223, no. 4, p. 120-123
- Gerola, H. and P. Seiden (1978). *Stochastic star formation and spiral structure of galaxies*. *Astrophys. J.*, 223, p. 129
- Greenberg, J. M., B. D. Hassard, and S. P. Hastings (1978). *Pattern formation and periodic structures in systems modeled by reaction-diffusion equations*. *Bull. Am. Math. Soc.*, 84, p. 1296
- Mandelbrot, B. (1983). *The Fractal Geometry of Nature*. Freeman
- Meinhardt, H. (1995). *The Algorithmic Beauty of Sea Shells*. Springer
- Peitgen, H.-O., H. Jürgens, D. Saupe (1992). *Chaos and Fractals*, Springer
- Prusinkiewicz, P. (1990). *The Algorithmic Beauty of Plants*. Springer
- Schewe, P. F (ed.) (1981). *Galaxies, the Game of Life, and Percolation*. *Physics News*, Amer. Inst. Phys. Pub. R-302, p. 61
- Wolfram, S. (1984a). *Computation Theory of Cellular Automata*. *Commun. Math. Phys.* 96, p. 15-57
- Wolfram, S. (1984b). *Universality and Complexity in Cellular Automata*. *Physica D*, Vol. 10, p. 1-35

Chapter 3: Distributed intelligence

In the last chapter we concluded that pattern formation in natural systems occurs as a consequence of simple local rules. We looked at plants, at artificial creatures in the game of life, and at seashells. We now look at emergence of behavioral patterns that can be interpreted by an external observer as some kind of “distributed intelligence”. In this chapter we will proceed by inspecting some examples from robots and natural agents. We start with an experiment in collective robotics. We then discuss self-organizing phenomena in insect societies. Next, we briefly present Craig Reynolds’s famous boids. Finally, we discuss some “guiding heuristics for decentralized thinking”, as outlined by Mitch Resnick.

3.1 An experiment: the Swiss robots

The Didabots are cleaning up

In what follows we summarize experiments conducted by Maris and te Boekhorst (1996) who studied a collective heap building process by a group of simple robots, called Didabots. Instead of predefining “high-level” capacities, Maris and te Boekhorst exploit the physical structure of the robots and the self-organizing properties of group processes. The main idea behind the experiments is that seemingly complex patterns of behavior (such as heap building) can result from a limited set of simple rules that steer the interactions between entities (e.g., robots) and their environment. This idea has, for example, been successfully applied to explain the behavior of social insects (see below).

Look at figure 3.1. There is an arena with a number of Didabots, typically 3 to 5. They are equipped with infrared sensors that can be used to measure proximity: They show high activation if they are close to an object and low or zero activation if they are far away. The range of the infrared sensors is on the order of 5 cm, i.e. relatively short. All the Didabots in this experiment can do is avoid obstacles. They are programmed with the following simple control rule: If there is sensory stimulation on the left, turn (a bit) to the right, if there is sensory stimulation on the right, turn (a bit) to the left.



Figure 3.1: Didabots in their arena. There is an arena with a number of Didabots, typically 3 to 5. All they can do is avoid obstacles.

Now look at the sequence of pictures shown in figure 3.2. Initially the cubes are randomly distributed. Over time, a number of clusters are forming. At the end, there are only two clusters and a number of cubes along the walls of the arena. These experiments were performed many times. The result is very consistent — there are always a few clusters and a few cubes left along the walls. What would you say the robots are doing?

“They are cleaning up”; “They are trying to get the cubes into clusters”; “They are making free space”; these are answers that we often hear. These answers are fine if we are aware of the fact that they represent an observer’s perspective. They describe the behavior. The second one also attributes an intention by using the word “trying”. We are the designers, we can say very clearly what the robots were programmed to do: to avoid obstacles!



Figure 3.2: Example of heap building by Didabots. Initially the cubes are randomly distributed. Over time, a number of clusters are forming. At the end, there are only two clusters and a number of cubes along the walls of the arena.

The complexity of the behavior is a result of a process of self-organization of many simple elements: the Didabots with their simple control rule. The Didabots only use the sensors on the front left and front right parts of the robot. Normally, they move forward. If they get near an obstacle within reach of one of the sensors (about 20cm), they simply turn toward the other side. If they encounter a cube head on, neither the left nor the right sensor detects an obstacle and the Didabot simply continues to move forward. At the same time, it pushes the cube. However, it pushes the cube because it does not “see” it, not because it was programmed to push it. For how long does it push the cube? Until the cube either moves to the side and the Didabot loses it, or until it encounters another cube to the left or the right. It then turns away, thus leaving both cubes together. Now there are already two cubes together, and the chance that yet another cube will be deposited near them is increased. Thus, the robots have changed their environment which in turn influences their behavior. While it is not possible to predict exactly where the clusters will be formed, we can predict with high certainty that only a small number of clusters will be formed in environments with the geometrical proportions used in the experiment.

The kind of self-organization displayed by the Didabots in this experiment is also called self-organization without structural changes: If at the end of the experiments, the cubes are randomly distributed again and the Didabots put to work on the same task, their behavior will be the same — nothing has changed internally. This is also the kind of self-organization displayed by physical systems, as for example the famous Bénard cells (when a heat gradient is applied to a liquid, the individual molecules organize into “rolls”). As soon as the energy input is switched off, the system is back to its original state. We talk about self-organization with structural changes, whenever something changes within the agent so that next time

around the behavior of the agent will be different. Such processes of self-organization with structural changes are found in the artificial chimp societies of Hemelrijk (see chapter 5). They are also found in the ontogenetic development of the brain: it is crucial that the organism changes over time, otherwise it could not improve its behavior.

Similar principles to the ones observed in the Didabot experiments can also be found in natural agents. Let us look at a number of examples.

Whereas in ants seemingly sophisticated group decisions may raise suspicion and induce a search for simpler mechanisms, this is less so for primates (i.e. monkeys and apes).

3.2 Collective intelligence: ants and termites

Self-organization in a “super-organism”

The article on Self-organization in a “super-organisms” by Rüdiger Wehner was distributed in class. Exact reference: Wehner, R. (1998). Selbstorganisation im Superorganismus. Kollektive Intelligenz sozialer Insekten. NZZ Forschung und Technik, 14. Januar 1998, 61.

Abstract: In the societies of social insects thousands of individuals — as if guided by an invisible hand of a central organizer — produce cognitive abilities that transcend the abilities of each individual member by far. However, the central organizer is pure fiction. The “soul of the white ant” does not sit inside the queen, but is decentralized as collective intelligence, distributed over all individuals. This collective intelligence unites biologists, computer scientists, and economists in an interdisciplinary endeavor.

Referring back to our robot experiments in the previous section, we have another instance of sorting behavior. Sorting behavior is also observed in ants.

Deneubourg’s model of sorting behavior in real ants

Examination of an ant nest yields that brood and food are not randomly distributed, but that there are piles of eggs, larvae, cocoons, etc. How can ants do this? If the contents of the nest are distributed onto a surface, very rapidly the workers will gather the brood into a place of shelter and then sort it into different piles as before. Deneubourg and his colleagues show that this sorting behavior can be achieved without explicit communication between the ants.

The model works as follows. Ants can only recognize objects if they are immediately in front of them. If an object is far from other objects, the probability of the ant picking it up is high. If there are other objects present the probability is low. If the ant is carrying an object, the probability of putting it down increases as there are similar objects in its environment. Here are the formulas:

$$p(\text{pick up}) = (k^+ / (k^+ + f))^2$$

where f is an estimation of the fraction of nearby points occupied by objects of the same type and k^+ is a constant. If $f=0$, i.e., there are no similar objects nearby, the object will be picked up with certainty. If $f=k^+$, then $p(\text{pick up})=1/4$ and it gets smaller if f approaches 1. The probability to put down an object is

$$p(\text{put down}) = (f/k^- + f)^2$$

where f is as before and k^- a different constant. $p(\text{put down})$ is 0 as f is 0, i.e., if there are no similar objects nearby the probability of putting the object down approaches 0. The more objects of the same type that are nearby, the larger is $p(\text{put down})$. The development of the clusters for real ants and for a simulation is shown in figure 3.3. Sorting is achieved by these simple probabilistic rules. There is no direct communication between the ants. The sorting behavior is an emergent property.

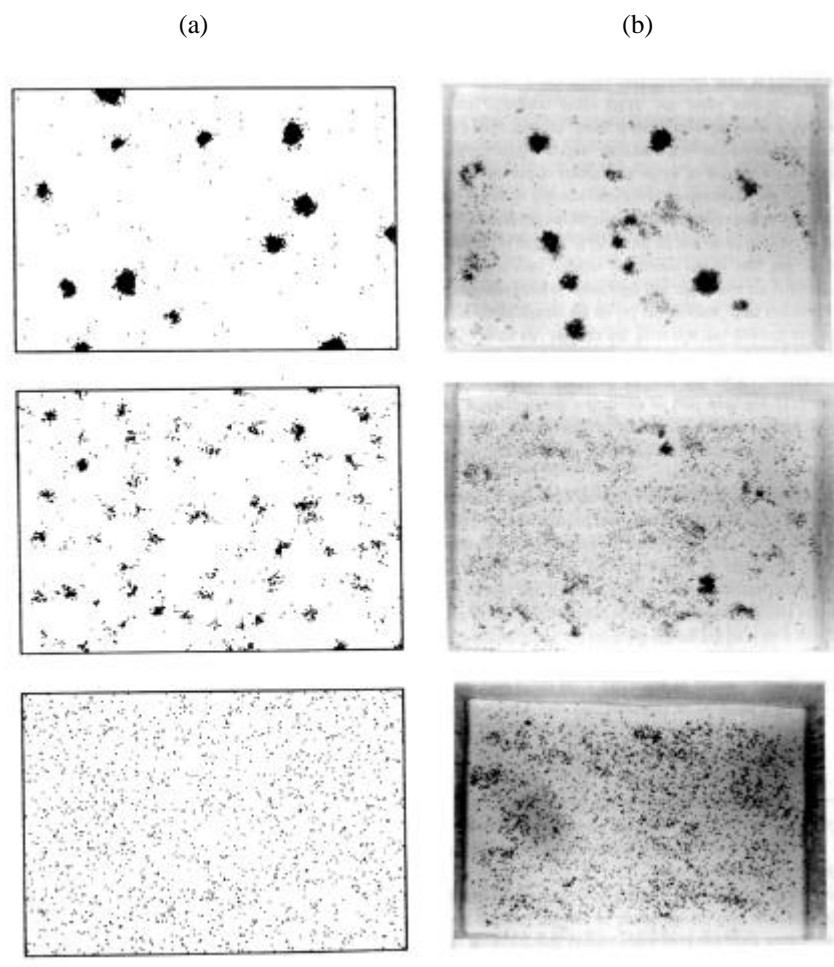


Figure 3.3. Development of clusters of objects in a society of ants. (a) Simulation. (b) Real ants. The simulation is based on local rules only. The simulated ants can only recognize objects if they are immediately in front of them. If an object is far from other objects, the probability of the ant picking it up is high. If there are other objects present the probability is low. If the ant is carrying an object, the probability of putting it down increases as there are similar objects in its environment. This leads to the clustering behavior shown.

While many people would agree that artificial life-like models have explanatory power for ant societies, they would be skeptical about higher animals or humans. Charlotte Hemelrijk and Rene te Boekhorst, two primatologists at the University of Zurich, had gotten interested in artificial life and autonomous agents. They were convinced that this kind of modeling technique could also be applied to societies of very high-level mammals like chimpanzees or orangutans. Hemelrijk used computer simulation models to study emergent phenomena in societies of artificial creatures, which, for her, were abstract simulations of orangutans. In an instructive paper entitled “Cooperation without genes, games, or cognition”, Hemelrijk (1997) demonstrated that cooperation in the sense of helping behavior is entirely emergent from interactive factors. Often, what seems to be a tit-for-tat strategy, as game theorists favor them, turns out to be a side effect of interactions between agents. A tit-for-tat strategy is one where the individuals keep track of what has happened and only give back as much as they have received. The more parsimonious explanations based on local rules of interaction also obviate explanations resorting to high-level cognition. For example, participants in a conflict are thought to keep track of the number of situations in which one individual has helped them and they have helped the other. For more detail, the reader is referred to Hemelrijk (1997). Along similar lines, te Boekhorst did a simulation of artificial “orangutans” demonstrating that travel band formation in orangutans can be explained in very simple ways (te Boekhorst and Hogeweg, 1994). This kind of research is predominantly conducted at the simulation level since often high-level operators like “recognize dominance rank” are used which cannot be translated to real robots in a straightforward manner.

Ants find their way to a food source

In their experiments on ants, Deneubourg and Goss (1989) asked whether the complexity of social interactions might be attributed to the individuals or to their interactions. For instance, colonies of certain species of ants appeared to be able to select the nearest food source among several that were present at varying distances from the nest. Attributing the complexity of this phenomenon to the individual ants would imply that individual ants compare the distances to several food sources and on the basis of this knowledge choose the nearest food source. This would entail ample cognitive calculations. Instead, Deneubourg and Goss clarified this choice as a consequence of the pheromonal marking and following system of the ants. Ants mark their path with pheromone when they leave the nest to search for food as well as on their journey back to the nest. At crossings where several paths intersect, they choose the most heavily marked direction. Ants return sooner from nearer food sources and as a consequence, shorter paths are marked more intensively than those leading to sources further away. Self-reinforced differentiation of the degree of marking of a path is also named an *autocatalytic* process, a particular instantiation of self-organization (remember the autocatalytic processes in pattern formation in seashells). Such autocatalytic processes have been invoked to elucidate several other aspects of insect behavior as well, e.g. the observed strict spatial distribution of honey, pollen and youngsters in the comb of bees (see the article by Wehner, 1998) and the way a comb is built. In all cases, assuming autocatalytic effects form an alternative to the idea that patterns are controlled centrally by a blueprint or high cognitive capacities of the individuals.

3.3 The simulation of distributed systems: Starlogo

Starlogo was developed out of the programming language Logo. Logo was developed originally for preschool children. It was simple enough so that the children could easily program simple robots, the “turtles”. Initially, the turtles were real robots capable of moving around on a flat surface. They were equipped with a pen, which could be in one of two positions, either up, or down. If it was down the turtle would draw a line on the ground as it moved. The Logo turtles were designed so that the children could playfully and intuitively learn concepts from geometry, mathematics, and engineering that are often considered hard (see Papert, 1980). For example they could explore ideas of feedback, geometric figures like polygons and circles, and infinitesimality. Later, the turtles were “virtualized” and became a means for drawing on a computer screen rather than on the physical ground (see also the “turtle graphics” discussed in chapter 2 in the context of Lindenmeyer systems). The commands that could be given to the turtles, like move forward, turn, pen-down remained the same. The language Logo has become very popular over the years and children like to play with it.

Starlogo is an extension of the traditional Logo language. First, there are many more turtles in Starlogo than in Logo. In a typical Starlogo program, there are literally hundreds or thousands of turtles. In other words, it is a massively parallel language. Second, Starlogo turtles have been equipped with sensors. While traditional Logo turtles were mainly used for drawing purposes, Starlogo turtles have to behave in ways that strongly depend on their environment, in particular their local environment. Behavioral rules can be defined for the turtles which makes it possible to model true turtle-environment interactions. Third, Starlogo there is the possibility to define so-called patches which can be used to model the properties of the environment. For example, pheromones deposited by the ants will automatically evaporate, or food that is eaten by the turtles grows back at a particular rate. The patches are much like cellular automata as discussed earlier. In summary, there are two types of rules, those for the turtles (or, more generally, the actors) and those for the environment. We will see precisely the same kind of distinction later on when discussing the Sugerscape model (Chapter 5).

The basic tutorial for Starlogo has been distributed in class.

For further information refer to the StarLogo homepage: lcs.www.media.mit.edu/groups/el/Projects/starlogo

3.4 Flocking — the BOIDS

The boids are among the most famous creatures in artificial life. They were invented by Craig Reynolds who was at the time, in the mid-eighties, working as a computer animator. In Culver City in California where he lived, he would observe flocks of blackbirds. He wondered how he could get virtual creatures to flock in similar ways. His hypothesis was that simple rules would be responsible for this behavior. It was clear to him that the boids would have to be agents: they would have to be situated, viewing the world from their own perspective, rather than from a global one. Their behavior is then controlled by a number of local rules. He came up with the following set (Reynolds, 1987):

- (1) Collision avoidance: avoid collision with nearby flockmates
- (2) Velocity matching: attempt to match velocity with nearby flockmates

(3) Flock centering: attempt to stay close to nearby flockmates.

Collision avoidance is the tendency to steer away from an imminent impact. Static collision avoidance is based on the relative position of the flockmates and ignores their velocity. Conversely, velocity matching is based only on speed. The third rule is about flock centering. It makes a boid want to be near the center of the flock. Because of the situated perspective of the boid, “center of the flock” means the perceived center of gravity of the nearby flockmates. If the boid is already well within the flock, the perceived center of gravity is already at the boid’s position, so there is no pull further towards the center. However, if the boid is at the periphery, flock centering will cause the boid to deflect somewhat from its path towards the center. Together, these three rules lead to surprisingly realistic flocking behavior.

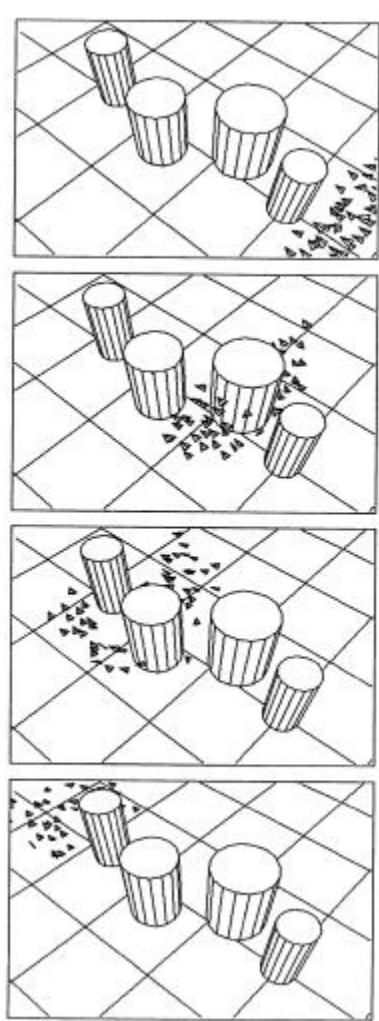


Figure 3.4: Craig Reynolds’s “boids” engaged in flocking behavior. They encounter a cluster of pillars. Amazingly enough, the flock simply splits and rejoins after it has passed the pillars. Note that “splitting” is not contained in the set of rules. It is truly emergent: the result of several parallel processes as the boids interact with the environment. It was to Reynolds’s own surprise: he did not know what was going to happen. The behavior of the boids can be fully explained — although remarkable and beautiful, there is nothing mystical about it.

Reynolds was interested in what would happen when the flock encountered obstacles. Would the boids continue to flock? Would they all move past the obstacle on one side? Or would they split? The latter

happened. Note that splitting was nowhere programmed into the boids. Both, the flocking behavior and the splitting behavior are truly emergent. There are a number of internal processes functioning in parallel (obstacle avoidance, velocity matching, and flock centering). They are based on the boids's situated view of their environment. The flocking behavior is very robust. This is due to the local distributed nature of the mechanism. Another wonderful example of how sophisticated behavior emerges from simple rules.

One last point. Researchers in artificial life claim that their creatures are behaving creatures in their own right. Boids are digital creatures as such — they are not only models of real birds. “Flocking in boids is true flocking, and may be counted as another empirical data point in the study of flocking behavior in general, right up there with flocks of geese and flocks of starlings.” (Langton, 1989, p. 33).

Rodney Brooks, Pattie Maes, Maja Mataric, and Grinnell Moore used rules almost exactly like Reynolds to achieve flocking behavior in real robots. At an IROS conference in 1990 (International Conference on Intelligent Robots and Systems), they suggested the use of a swarm of robots to prepare the lunar surface for a manned mission. Maja Mataric implemented flocking on her robots using a variation of Reynolds's rules. Her robots, just as the boids, exhibit robust flocking behavior. Again, flocking is emergent from local rules. Let us finish with a quote by Dan Dennett, a champion in the philosophy of mind: “Maja's robots are flocking, but that's not what they think they are doing.” (Dennett, 1997, p.251) What they think they are doing is applying Reynolds's rules. Another example of the notorious frame-of-reference issue. For those interested in collective robotics, Maja Mataric has investigated the field for many years. A thorough review would be beyond the scope of this book. The interested reader is referred to some of the review papers (Mataric, 1995, 1997).

Comment: In class, some details concerning the implementation of the boids were discussed. Issues include: (i) Should a situated perspective or an external observer's (or God's) perspective be used? (ii) How should the term “nearby flockmates” be interpreted? Should a sphere be placed around the boid, or should the boid have a particular field of vision? (iii) How is velocity matching performed? How do the boids “realize” that they are slower or faster than their neighbors? And so forth.

Online resources are found at:

- www.cs.toronto.edu/~dt/siggraph97-course/cwr87/ (by Craig Reynolds)
- www.cse.unsw.edu.au/~conradp/java/Boids/example.html (a very nice Java applet)

3.5 Guiding heuristics for decentralized thinking

To conclude our discussion of distributed intelligence, this section provides a brief summary of the major points discussed in this chapter.

(i) Positive feedback isn't always negative

Positive feedback has a negative image problem:

- screeching sounds that result when a microphone is placed near a speaker
- population growth

- a vicious circle (German: Teufelskreis) is based on positive feedback. For example, reduction of service in public transportation, leads to a vicious circle: Reduction in service leads to additional frustration of users, leading to further decreases in users, which in turn necessitates further reduction of services.

However, positive feedback is not always negative but can have positive effects. More neutrally, it underlies a large number of phenomena in nature and society.

Examples:

- the emergence of Silicon Valley: snowball effect
- winning of standards/computers --> Macintosh problem; Windows as the most successful software of all times. Of course, these developments are seen as negative by some.
- formation of ant trails: see above
- in the robot-clustering task there is positive feedback: the more cubes are together in a cluster, the higher the probability that yet another cube will be deposited nearby
- interaction of positive and negative feedback. An instance of this interaction is the evaporation of pheromones in the formation of ant trails.

(ii) Randomness can help create order

Randomness is not only “disorder” but random perturbations can have an important role in self-organizing systems:

- in traffic jams: small fluctuations in traffic density serve as “seeds”; positive feedback accentuates these density fluctuations, making the seeds sprout into full-fledged traffic jams
- randomness is required to achieve adaptivity in pheromone trails. If the ant societies are to remain adaptive there must always be a random component in the ant’s choice behavior (see the shortcut problem, next chapter). This is important if this idea is to be applied to ant-based control, i.e. to message routing in telecommunication networks (see chapter 4).
- clapping of an audience often develops into rhythmic clapping

(iii) A flock isn’t a big bird

- levels should not be confused: often people confuse the behaviors of individuals with the behaviors of groups; e.g. interactions among individual birds give rise to flocks
- in many cases, the individuals on one level behave differently from objects on another level: traffic jams tend to move backward, even though all of the cars within the jams are moving forward
- in the Sugarscape model (see below), there are diagonal migration patterns even though the agents can only move up/down and left/right
- the “leader” in a flock is always changing

(iv) A traffic jam isn't just a collection of cars

- It is important to realize that some objects (“emergent objects”) have an ever-changing composition
- Over an individual's life-time cells are always dying and new cells are being created, but the individual remains the same individual
- the ants that make up an “ant bridge” change continuously
- the water making up the shape of a fountain is always different, but the shape remains

(v) The hills are alive (the environment has an independent dynamics)

- people often focus on the behaviors of individual objects, overlooking the environment that surrounds the objects
- resources are exploited, pollution diffuses to other patches, etc. (see also agent-based simulation in the Sugarscape model, chapter 5)

(adapted from Resnick, 1997)

References

- te Boekhorst, I.J.A., and Hogeweg, P. (1994). Effects of tree size on travelband formation in orangutans: data analysis suggested by model study. In R. Brooks, and P. Maes (eds.): *Artificial Life IV Proceedings*. Cambridge, Mass.: MIT Press, 119-129.
- Deneubourg, J. L. and Goss, S. (1989). Collective patterns and decision-making. *Ethology, ecology and Evolution*, **1**, 295-311.
- Dennett, D.C. (1997). Cog as a thought experiment. *Robotics and Autonomous Systems*, **20**, 251-256.
- Hemelrijk, C. K., (1997). Co-operation without games, genes or cognition. In P. Husbands and I. Harvey (eds.) *Fourth European Conference on Artificial Life*. Cambridge, Mass.: MIT Press, 511-520.
- Langton, Ch. (1989). Artificial Life. In Ch. Langton (ed.). *The proceedings of an interdisciplinary workshop on the synthesis and simulation of living systems*. Redwood City: Addison-Wesley.
- Maris, M., and te Boekhorst, R. (1996). Exploiting physical constraints: heap formation through behavioral error in a group of robots. In *Proc. IROS'96, IEEE/RSJ International Conference on Intelligent Robots and Systems*.
- Mataric, M. (1995). Designing and understanding adaptive group behavior. *Adaptive Behavior*, **4**, 51-80.
- Mataric, M. (1997). Learning social behaviors. In R. Pfeifer, and R. Brooks (eds.). *Robotics and Autonomous Systems*, Special Issue on “Practice and Future of Autonomous Agents”, 191-204.
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.
- Resnick, M. (1997). *Turtles, termites, and traffic jams. Explorations in massively parallel microworlds*. Cambridge, Mass.: MIT Press.

Reynolds, C.W. (1987). Flocks, herds, and schools: a distributed behavioral model. *Computer Graphics*, **21**, 25-34.