Chapter 1
Introduction to Modeling of Complex Systems Using Cellular Automata

Alfons G. Hoekstra, Jiří Kroc, and Peter M.A. Sloot

"The real purpose of scientific method is to make sure Nature hasn't misled you into thinking you know something you don't actually know."

Robert M. Pirsig
Zen and the Art of Motorcycle Maintenance, 1974.

Since the sixteenth century there have been two main paradigms in the methodology of doing science. The first one is referred to as "the experimental" paradigm. During an experiment we observe, measure, and quantify natural phenomena in order to solve a specific problem, answer a question, or to decide whether a hypothesis is true or false. The second paradigm is known as "the theoretical" paradigm. A theory is generally understood as a fundamental, for instance logical and/or mathematical explanation of an observed natural phenomenon. Theory can be supported or falsified through experimentation.

The roots of the experimental and theoretical paradigms occurred much earlier and were already used by Pythagoras, Euclid, Archimedes and others (e.g. during ancient times in Greece, China, Egypt and other cultures thousands years BC). Since that time, the systematic use of those two paradigms has enabled us to understand and quantify some bits and pieces of Nature.

Since the Second World War, a third scientific paradigm appeared on the scene. This one is usually referred to as "the computational" paradigm, in which we study Nature through computer simulations. The first theoretical results dealing with this paradigm can be attributed to Alan Turing [24] in the 1930s. Computation is neither theory nor experiment. Computations can be implemented by many means: mechanically, electro-mechanically (for example the Bombe machine – used to decipher the German Enigma coding-machine), using electrical circuits (the first two programmable digital electronic computers Colossus and ENIAC), electronically (nowadays built in silico computers), chemically, biochemically, using DNA, quantum mechanically and in many other ways not mentioned here.

A.G. Hoekstra (✉)
Computational Science, Faculty of Science, University of Amsterdam,
Science Park 107, 1098 XG, Amsterdam, The Netherlands
e-mail: a.g.hoekstra@uva.nl

1.1 The Computational Paradigm

The computational paradigm uses computation to describe systems and natural phenomena employing a computer. The computational paradigm plays a fundamental role in situations where analytical descriptions of the observed phenomena are not tractable and/or out of reach of direct experimentation. Outputs from computations are often validated against experimental data and against simplified analytical models. The power of the computational paradigm has been demonstrated in physics, mathematics, chemistry, engineering and its importance is continuously increasing in such fields such as biology, medicine, sociology and psychology.

The invention of the computer enabled the solution of analytical models in terms of their numerical implementations, where by numerical we usually mean discretization of space and time and keeping continuous variables. In this way, for example, Partial Differential Equations (PDEs) were solved by the Finite Element Method (FEM). It brought a big breakthrough in scientific and engineering solutions of various problems. Finally, scientists realized that sometimes it is not necessary, or even possible, to design an analytical model describing a given natural phenomenon. In such cases, the observed phenomena can often be directly implemented using a discrete model.

1.2 Modeling

An abstract model is a construct incorporating a number of variables, processes and relationships among them. A model in this sense usually means that details about the original, modeled phenomenon are excluded from the model itself. An abstract model employing mathematical methods and tools to describe a natural phenomenon or system is called a mathematical model. Mathematical models are used within physics, engineering, biology (natural sciences), sociology, psychology, political science, economics (social sciences). Mathematical models employ many different types of mathematical disciplines and methods as statistics, differential equations, integro-difference equations, dynamical systems, game theory, particle systems, systems of difference equations, cellular automata, and many other not mentioned here.

A computer model (often called a computer simulation or computational model) is an abstract model implemented into a computer program. Computer models of natural systems (as a part of mathematical modeling) are widely used in physics, biology, social sciences, economics and engineering.

Mathematical and computational models can be grouped according to the use of continuous or discrete values in *space*, *state variables* and *time* into several computationally different classes (for details see Table 1.1).

Toffoli and others [22, 25] pointed out the following sequence of approximations typically used during modeling of natural phenomena where differential equations are used. Initially, there is a naturally observed phenomenon. This phenomenon is

Table 1.1 Analytical and numerical methods used to model natural phenomena where C stands for continuous and D for discrete *state*, *space*, or *time* [3, 7, 21]

Type of model	State	Space	Time
Partial differential equations (PDEs)	C	C	C
Integro-difference equations	C	C	D
Coupled ordinary differential equations (ODEs)	C	D	C
Interacting particle systems	D	D	C
Coupled map lattices (CMLs) and systems of difference equations lattice Boltzmann equations (LBEs),	C	D	D
Cellular automata (CAs) and lattice gas automata (LGAs)	D	D	D

observed and studied through experimentation. Then a mathematical model is build. In physics, it often means that suitable (set of) differential equations is used (e.g. partial differential equations). This represents the first-level approximation. In the second-level of approximation, the mathematical model is discretized into a numerical one. Typically, such numerical schemes end up with a numerical error.

The third-level approximation is associated with the representation of real numbers on computers. In general computers approximate most real numbers by using nearby rational numbers. In some cases irrational numbers can be treated exactly by computers (e.g. sqrt(2) function). Computers use either floating point or fixed-point number representations, resulting in rounding errors.

Contrary to the three previously listed approximations, we identify only one approximation in computational modeling with cellular automata [22, 25]. The natural phenomenon is directly implemented in the form of a cellular automaton. This is a potential advantage compared to the previous approaches. Due to the presence of only one approximation, the expressivity of cellular automata is in many cases higher as compared to other techniques.

1.3 Complex Systems

In complex systems we observe group or macroscopic behavior emerging from individual actions and interactions. One of the first biological observations of complex systems, if not the first one, is linked to ant-colony behavior [9], see Fig. 1.1. Each ant is simply following its internally encoded reactions to external stimuli. It is a well known fact that ant species may have a set of 20 up to 40 reactions on any given external stimulus. There is no ant "leader" which tells other ants what they should do nor a hierarchy of such leaders. Despite the lack of controlling entities, an ant-colony builds its ant-hill, feeds it, protects it, attacks other colonies, follows a foraging strategy, etc. This emergent behavior observed in ant-colonies is prototypical for many other complex systems.

We now know that a carefully selected set of reactions on external stimuli of large number of identical copies of several generic, mutually interacting entities creates a complex system often displaying self-organization and/or emergent behavior [1].

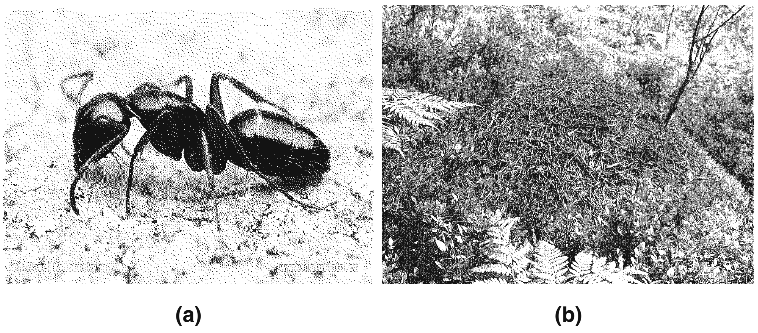

Fig. 1.1 One of the biggest challenges of the current level of scientific understanding of Nature is to find out principles behind self-organizing and emergent systems. An excellent example is the ant-colony. One ant (**a**) (species *Formica*) in isolation is a "simple" system, 1000s ants working together are capable to build complex structures like an ant-colony without any central control (**b**). The solution of the backward problem is "relatively" easy (i.e. if we know ant-colony behavior then we could find out local rules through the observation of ants). The forward problem is in general untractable by all currently known techniques. The question is which set of local rules will lead to the desired global response

Unfortunately, we do not know a generic procedure or algorithm to design such systems. It is a tremendous task to find a set of local interactions which produce a desired global response. The opposite direction, which might be called a backward (or deconstructive) one, is relatively easy to perform.

Another striking biological example of self-organization is the existence of "V-formations" spontaneously occurring within flocks of flying birds (e.g. geese) [15]. The model describing this phenomenon takes into account line of sight and aerodynamic advantages of flocking birds due to an upwash behind their wing tips creating extra lift. Each bird controls its actual position according to its nearest neighbors with respect to aerodynamics and vision. The final "V-like formation" a flock spontaneously occurs due to this self-organization, regardless of the initial positions of birds.

These two examples lead us to the concept of complex systems. Complex systems were independently and often simultaneously (re)discovered in many scientific fields [1, 4, 6, 17]. This is an indirect indication of their universality. A typical complex systems consists of a vast number of identical copies of several generic and mutually interacting processes. Despite the lack of global control complex systems often express (macroscopic) self-organization and emergence, which are typically driven by dissipation of energy and/or information.

In general, self-organization often results from a competition between counteracting processes. A good example is provided by self-organized criticality observed in Earthquakes where the system is constantly fed by elastic energy which is released abruptly in the form of avalanches of various size and intensity. Self organised criticality has been observed in many natural and man-made systems [19]. Self-organization generates very robust solutions, which are intrinsically resistant to any kind of external and/or internal noise and perturbations. Self-organization is often accompanied by emergence and vice versa.

Emergence is defined as the occurrence of a new quality operating at a higher level than where the system units are operating. Going back to the example of ant-colonies, an ant hill is an emergent property arising from the local interactions of ants. A whole hierarchy of emergents can exist, like e.g. in the human body where we see a cascade from DNA, amino acids, polypeptides, proteins, cell structures, cells, tissues, organs, bodies, societies, ecosystems, etc.

Complex system behaviour is observed within Earthquakes, volcano activities, stock market behavior, social networks, traffic flow, etc. Complex systems like that express self-organized criticallity. There is a group of complex systems that show percolation behavior of some property such as, e.g., conductivity of a mixture of powders of an isolator and a metal, forest fires, opinion development within societies, and so on. It is worth stressing that no generally accepted definitions exist for complex systems, self-organization, and emergent behavior. There is a variety of techniques to model complex systems. In the following parts of the introduction as well as in the whole book, the attention is focussed to one of those computational techniques called "cellular automata".

1.4 Cellular Automata

The notion of *cellular automata* has a long, living history going back to Stanislav Ulam and John von Neumann, see for instance [21]. Von Neumann's cellular automaton (in plural cellular automata) [16] represents a direct predecessor of cellular automata (CAs). Shortly, von Neumann introduced cellular automata as a computational medium for machine self-replication motivated by a simple question: "What kind of logical organization is sufficient for an automaton to be able to reproduce itself?". In another words "Can we reproduce computationally and in silico what living cells do?" Unluckily, he left his work unfinished. In principle, self-replication is possible to solve but, so far, the problem is far from being finished. Self-replication still attracts the attention of many researchers. There are two main streams in the design of self-replicating structures. The first one uses hand made local rules and topologies whereas the second one employs evolutionary algorithms to explore self-replicating structures and rules. You will find examples of both approaches in this book.

After the very promising start of cellular automata many theoretical and practical applications and models of natural phenomena were successfully described and solved by computational models using classical cellular automata. Some of them are presented in this book. There are still a vast number of prospective applications of cellular automata in almost all scientific fields. As you will see in this book, cellular automata themselves are undergoing a development as a computational method as well.

1.5 Classical Cellular Automata

The oldest versions of cellular automata have a simple and straightforward implementation. It is based on the use of a regular array of cells (often implemented as a matrix in computer simulations), their local variables, and a transition function working over a neighborhood. Those components of classical cellular automaton employing a von Neumann neighborhood are depicted in Fig. 1.2.

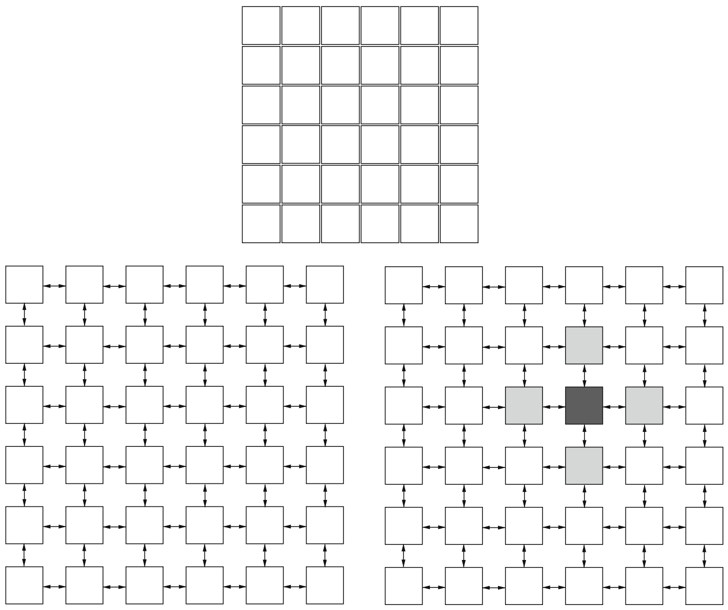

Fig. 1.2 A two-dimensional lattice defining cellular automaton having a size of 6 × 6 cells is shown (*top*). The spatially expanded lattice with explicitly displayed links (*lines with arrows*) between neighboring cells (*bottom-left*). Such topology, where only the nearest neighbors are linked, defines a von Neumann neighborhood with a radius $r = 1$ (*bottom-right*), an updated cell (*dark gray*) and neighbors (*light gray*). In general, different neighborhoods are defined by different sets of links

Let us define cellular automata as follows:

- *A discrete cellular state space* $\mathcal{L}$ is a set of locally interconnected finite state automata (FSAs) that is typically created by a regular d-dimensional lattice of FSAs. For example, in two-dimensions, a cellular automaton consists of a lattice of $\mathcal{L} = m \times n$ squares where each square, in the literature called a cell, is represented by one FSA.
- *A local value space* $\sum$ defines all possible states for each FSA. The state σ of each FSA can be in one of the finite number of states $\sigma \in \sum \equiv \{0, 1, 2, 3, 4, \ldots, k-1, k\}$. The composition of all possible states of all cells create a state space of the whole cellular automaton.

- A *neighborhood* $\mathcal{N}$ is created by a set of N topologically neighboring cells, which are influencing a change of the state of each updated cell in the next simulation step. Typically, homogeneous neighborhoods (composed of nearest neighbors) are used, see Fig. 1.3 for examples of von Neumann and Moore neighborhoods.
- *Boundary conditions* can be periodic, fixed or reflecting among others. The most important are periodic boundary conditions, which are used to simulate the infinite lattice using a finite one. Periodic boundary conditions are implemented as a torus in two dimensions.
- A *transition rule* ϕ: $\overbrace{\sum \times \sum \times \cdots \times \sum}^{\text{N}} \to \sum$ describing the change of each updated cell from its current state value to a new one, operating over the neighborhood (having size N) – is defined.

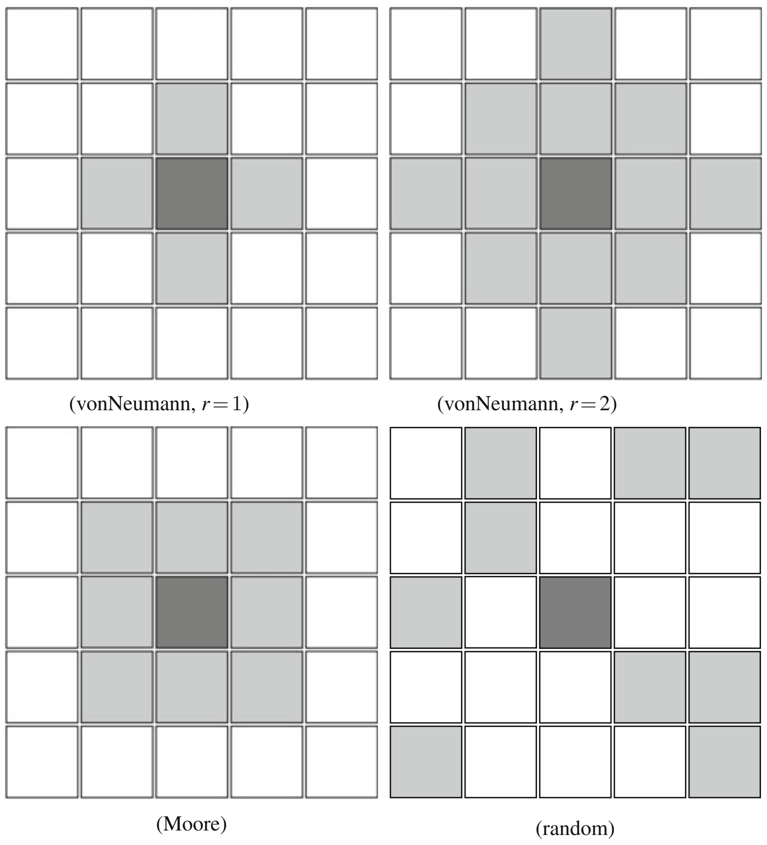

Fig. 1.3 Various types of neighborhoods are shown: von Neumann (radius 1 and 2), Moore, and a random one. Variables of cells within a given neighborhood (*light gray* plus *dark gray*) are used to evaluate new values of the updated cell (*dark gray*) using transition function

- All cells change their state synchronously at an externally provided clock step. This is usually called one iteration step.

A definition of a neighborhood is often provided in the following way. In the case of a von Neumann neighborhood having radius equal to one

$$\begin{aligned}\mathcal{N}_{\mathrm{N}}^{1}(i,j) &= \{\sigma_{k,l} \mid |i-k| + |j-l| \leq 1\} \\ &= \{\sigma_{i,j}, \sigma_{i-1,j}, \sigma_{i,j-1}, \sigma_{i+1,j}, \sigma_{i,j+1}\}\,.\end{aligned} \tag{1.1}$$

This von Neumann neighborhood (with radius equal to one) is depicted in Fig. 1.3. Only the nearest neighbors of the updated cell are involved in this neighborhood (i.e. four plus one in total). It is worth to mention out that the updated cell itself (i.e. the one with indexes (i, j)) is involved in the neighborhood as well. The other type of neighborhood is Moore neighborhood having a radius r equal to one

$$\begin{aligned}\mathcal{N}_{\mathrm{M}}^{1}(i,j) &= \{\sigma_{k,l} \mid |i-k| \leq 1, |j-l| \leq 1\} \\ &= \{\sigma_{i,j}, \sigma_{i-1,j}, \sigma_{i-1,j-1}, \sigma_{i,j-1}, \sigma_{i+1,j-1}, \\ &\quad \sigma_{i+1,j}, \sigma_{i+1,j+1}, \sigma_{i,j+1}, \sigma_{i-1,j+1}\}\,.\end{aligned} \tag{1.2}$$

This Moore neighborhood is depicted in Fig. 1.3. It is composed from the first and second nearest neighbors and the updated cell itself (i.e. eight plus one in total). All members of the neighborhood are numbered anti-clockwise in both cases. In general, the size of the lattice must be much larger than the size of the neighborhood otherwise every cell becomes dependent on other cells.

A commonly used definition of cellular automata states that they represent dynamical systems where space, variables, and time are discrete. Such definition enables to employ techniques developed in statistical physics that are used to predict average, asymptotic and other properties of simulated natural phenomena. Results of cellular automata models are often compared to results of analytical models using a mean field approximation. It is a known fact that the behavior of cellular automata is in general unpredictable. This property is often applied in the design cellular automata used for encryption.

Any transition rule $\phi(t)$ can be written in the form of a function $\phi(t)$ using states $\sigma(t)$ of all cells in the neighborhood

$$\sigma_{i,j}(t+1) = \phi(\sigma_{k,l}(t) \mid \sigma_{k,l}(t) \in \mathcal{N}), \tag{1.3}$$

where $\mathcal{N}$ defines the neighborhood including the cell itself, see Eqs. (1.1) and (1.2), and Fig. 1.3 for examples of neighborhoods. The transition rule ϕ can be any combination of arithmetical and logical operations, including functions. In real applications, the typical transition rule has either the form of a transition table (defining for which input values a certain output value is taken) or the form of a computer program.

In order to provide an illustrative example, the general form of a transition rule for a von Neumann neighborhood $\mathcal{N}_{\mathrm{N}}^{1}$ is given by

$$\sigma_{i,j}(t+1) = \phi(\sigma_{i,j}(t), \sigma_{i-1,j}(t), \sigma_{i,j-1}(t), \sigma_{i+1,j}(t), \sigma_{i,j+1}(t)) , \tag{1.4}$$

which contains all five neighbors including the updated cell itself.

The number of all possible local rules N_r depends on the number of states σ and the number of neighbors n for a given cellular automaton in the following way

$$N_r = \sigma^{\sigma^n} . \tag{1.5}$$

As shown in Table 1.2, the number of rules dramatically increases with the number of neighbors n and states σ.

In general, the most difficult task in the design of cellular automata is to find a transition rule that will describe the temporal evolution of a modeled system, i.e. which leads to desired global response of the system. As the number of possible rules dramatically increases with a number of states σ and the size of the neighborhood n, it is usually non-trivial to find correct transition rules describing the system being modeled.

In order to narrow down the huge space of all possible rules of a cellular automaton for a given number of states σ and the size of neighborhood n, attempts were made to classify rules according to their complexity. Wolfram [26] proposed four classes of cellular automata behavior: (1) almost all configurations relax to a fixed configuration, (2) almost all configurations relax to either a fixed point or a periodic cycle according to the initial configuration, (3) almost all configurations relax to chaotic behavior, (4) sometimes initial configurations produce complex structures that might be persisting or long-living.

Table 1.2 The total number of local rules for a cellular automaton having a number of states σ and number of neighbors n. It is evident that even for automata with a relatively small number of states and neighbors the number of all possible rules increases dramatically with the number of states and neighbors

Number of states σ	Number of neighbors n	σ^{σ^n}	Number of rules N_r
2	2	2^{2^2}	16
2	3	2^{2^3}	256
2	5	2^{2^5}	4 294 967 296
2	10	$2^{2^{10}}$	$1.797 \cdot 10^{308}$
5	2	5^{5^2}	$2.98 \cdot 10^{17}$
5	3	5^{5^3}	$2.35 \cdot 10^{87}$
5	5	5^{5^5}	$1.91 \cdot 10^{2184}$
10	2	10^{10^2}	10^{100}
10	3	10^{10^3}	10^{1000}
10	5	10^{10^5}	10^{100000}

Langton studied the average type of behavior with respect to a statistic provided by the parameter λ of each rule table [13]. In a binary cellular automata, the parameter λ is the fraction of "ones" within the output column of a transition table. Advantages and disadvantages of both classification methods are discussed in detail elsewhere [14]. There have been a wide number of attempts to classify cellular automata rules without substantial success. All known classification techniques are not reliable (they misclassify rules quite easily). A search for a reliable classification technique is still needed for the design of new cellular automata rules having desired properties. Currently, the lack of such classification techniques is typically overcome by use of, e.g., genetic programming, genetic algorithms or any other evolutionary algorithm.

1.6 The Game of Life

Before we proceed let us emphasize that not all complex systems can be described mathematically. One of the best known examples of such a complex system is the Game of Life [8]. Due to its simplicity, it became a prototypical example of a complex system formulated by a cellular automaton.

The Game of Life represents a simple, vital, widely studied example displaying an extremely complex behavior arising from an extraordinary simple local rule [8]. This example is, either directly or indirectly, motivating many applications within such diverse fields as game theory, sociology, ecology, fire spreading, market behavior, etc. The transition rule consists of the evaluation of four independent logical conditions for each updated cell separately:

1. a living cell having less than two living neighbors dies (loneliness);
2. a living cell having more than three living neighbors dies (overcrowding);
3. a living cell having two or three living neighbors stay living (ideal living conditions);
4. a dead cell having exactly three living neighbors becomes alive (offspring).

When we take these simple rules and apply them to a randomly chosen initial configuration of living and dead cells within a lattice of $N \times M$ cells, amazing patterns start to appear. We observe configurations made of living cells transversing through space and persisting over time. Such configurations are called "gliders", see Fig. 1.4 for a sequence of snapshots. We might also find another configuration generating those "gliders" (called "glider-guns") or configurations destroying them (called "glider-eaters"), or oscillating structures. A lot of effort has been spent on the discovery of various structures emerging from the Game of Life.

Figure 1.5 shows two types of structures, a "glider-gun" and "gliders" produced by this "glider-gun". The "glider-gun" located at the bottom-left is continuously creating persistent, moving, living structures called "gliders" also shown in Fig. 1.4.

It is possible to create a whole set of logical and arithmetic operations using NOT, AND, and OR gates implemented within the Game of Life employing such building

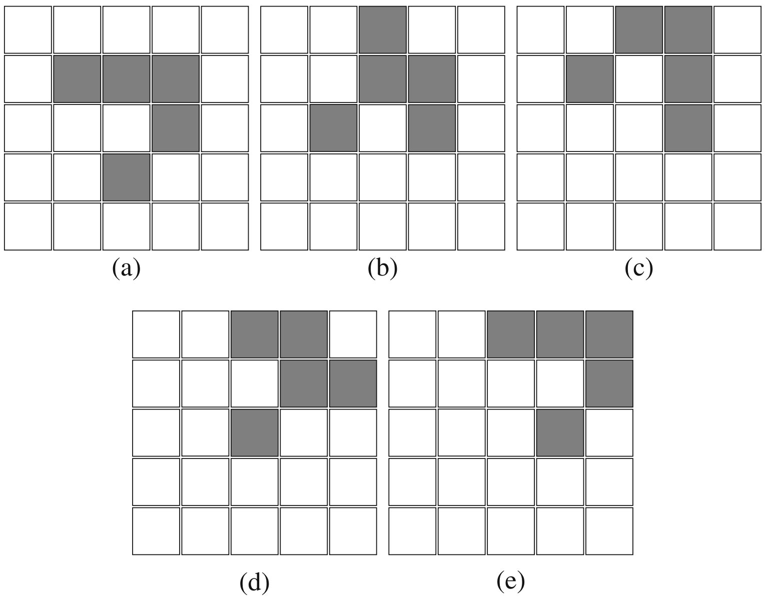

Fig. 1.4 A sequence of snapshots (**a**)–(**d**) depicting the propagation of initial configuration of one "glider" through a cellular space observed within the Game of Life. Coordinates of all cells are kept the same within all sub-figures. The "glider" moves by shifting itself one cell upwards and one cell to the right within four time steps and so on. There are four diagonally symmetrical directions of glider propagation

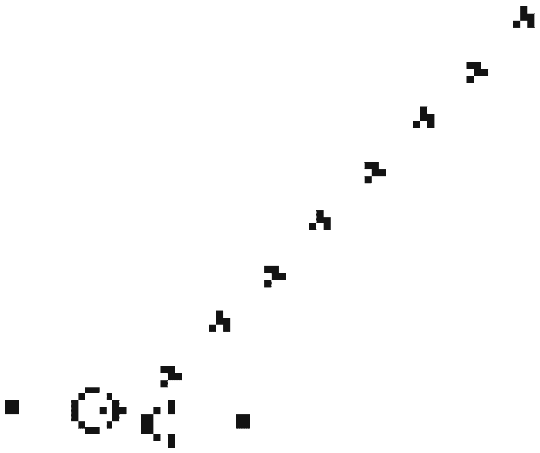

Fig. 1.5 One type of glider-gun (**bottom**) creating gliders (moving from the **left-bottom** corner to the **right-top** one) observed within the Game of Life. The whole logics and arithmetics can be build from them (such operations as NOT, AND, OR, etc.). Any recursive function can be build using such operations. Therefore the Game of Life is universal

units as gliders, glider-guns, and glider-eaters. Any recursive function can be build using those elementary building units, see Fig. 1.5.

The occurrence of real complex structures within a randomly initiated simulation of the Game of Life is very likely. It is shown that the Game of Life has the universality of a Turing machine. Surprisingly, even such a simple model motivated by the behavior of living matter displays self-organization and emergent behavior. These two fundamental processes operate within many complex systems.

There is no efficient way to "simulate" the game of life through a set of equations. In this case the efficiency of the cellular automata is much higher than any classical method, since the use of cellular automata decreases the level of dissipation of local information within the system. This is equivalent to the well-known Church-Turing hypothesis, stating that the final configuration can only be obtained through explicit simulation.

1.7 Advanced Cellular Automata

Classical cellular automata can be generalized in many ways. The best known and the most important generalizations, like the lattice Boltzmann method, networked automata, complex automata, asynchronous automata, quantum cellular automata, wetware, and real valued cellular automata are briefly introduced.

The lattice Boltzmann method was developed to simulate fluid flow as an alternative to the Navier–Stokes equations. It is based on the use of a discrete Boltzmann equation operating on a lattice of edges where virtual particles are moving. Particles have a distribution of velocities at each lattice point. They move along those edges and undergo collisions. Collisions are designed in a such way that time-averaged motion of particles is consistent with the Navier-Stokes equations. The lattice Boltzmann method is discussed in more detail in the chapter by Kusumaatmaja and Yeomans in this book. The lattice Boltzmann method is used to model a wide range of applications for single- and multiphase flows within complex geometries including porous media [5].

Networked automata appeared on the scene along with understanding that the lattice itself on which a particular cellular automaton operates can be understood as a network. The question is how a change of a regular lattice (i.e. a regular network) to more general types of networks influence the overall behavior of a specific cellular automaton under study. The answer is in many cases surprising. Such complicated networks as scale-free and small-world networks have in most cases a large, positive impact on the efficiency and robustness of the computation. What is even more important, such networks more precisely reflect natural phenomena which are described by cellular automata (e.g. social networks, gene regulatory networks, the Internet, etc.).

Complex automata (CxA) employ the idea that a multi-scale system can be decomposed into a number of single-scale automata, which are mutually interacting. The evolution of the system is driven by evolution at single-scale levels which are influenced by evolution at the other scale levels. There is a whole number of possible implementations of multi-scale models using cellular automata. Updating of the multi-scale model could be done in a variety of ways. This new type of cellular automata are introduced in this book by Hoekstra et al.

For a long time it was assumed that synchronous updating of the whole lattice of cells is fundamental for computations employing cellular automata. This assumption is proven to be too strict and can be weakened to *asynchronous automata*. Simply said, it is possible to develop asynchronous cellular automata rules, which perform the same tasks as synchronous cellular automata. The asynchronous updating mode of computation occurs in many real applications. It is known that an asynchronous CA for a particular task is more robust and error resistant than a synchronous equivalent.

Quantum cellular automata (QCA) proposed by Richard P. Feynman employ quantum mechanical phenomena in the design of computational models. They represents the quantum mechanical analogy of classical cellular automata. In many cases, they employ completely different computational principles.

Wetware represents one of the prospective media to implement cellular automata. This area of research remains mostly open for future discoveries. Employing chemical and/or biochemical reactions (going even up to the level of cells and tissues) may bring new computational tools (called wetware) potentially having a tremendous computational capacity (e.g. DNA computing). The term wetware stands for an abstraction used for hardware and software. Wetware is build on completely different computational principles compared to the ones currently used in silico circuits.

Some authors are using the notion of a cellular automaton even when the associated variables have real values. This is valuable in many models of naturally observed phenomena (e.g. recrystallization in solid state physics). Cellular automata using real values are often referred to as coupled map lattices (especially in physics).

1.8 Book Organization

The book is organized in three parts: (1) Theory of Cellular Automata; (2) Applications of Cellular Automata; and (3) Cellular Automata Software. The theory part contains fundamental chapters on different aspects of modeling complex systems with cellular automata. The applications part presents a number of representative examples of applications of Cellular Automata in a large range of scientific disciplines. The part on software highlights the important work that has been done to create high quality software environments that allow to quickly and relatively easily implement a Cellular Automata model and run simulations.

The first part on theory contains eight chapters, discussing fundamental issues related to complex systems modeling with Cellular Automata. An important class of complex systems are multi-level systems, where interactions between multiple levels to a large extend dictate their properties. In three chapters different aspects of this class of Cellular Automata Models are discussed. In Chap. 2 Hogeweg introduces multilevel Cellular Automata, where the states and rules of lower levels, that produce patterns on higher levels, can by adjusted by those very patterns in a feedback loop. Applications in Bioinformatic systems are reviewed. Hoekstra et al., in Chap. 3, propose Complex Automata as a formalism to model multi-scale complex systems. The idea is to decompose a multi-scale system into a set of single scale models, where each single scale model is represented by a CA. While Complex Automata assume a form of scale separation, Dunn in Chap. 4 introduces a strongly coupled hierarchical Cellular Automata to model multiscale, multiresolution systems, with special projection operators to formalize the coupling between the levels in the hierarchy. Applications in landscape ecology serve as an example.

Bandman, in Chap. 5, introduces a formalism for Cellular Automata as a model of spatially extended dynamical systems, focussing on composition techniques to develop and formally analyze advanced Cellular Automata models for such dynamical systems.

The great asset of Cellular Automata has always been to use them as minimal models capturing the main features of the dynamics of a system under study. In two chapters examples of such minimal Cellular Automata models are further explored. Chapter 6, by Umeo, discusses in detail 1-bit communication Cellular Automata, where inter-cell communication per step is restricted to one bit, applied to a number of applications. Adamatzky and Grube consider minimal Cellular Automata models for population dynamics, introducing only six very basic types of interaction between two species, and then exploring the behavior of their models as a function of these interactions.

Tomassini, in Chap. 8, introduces Cellular Evolutionary Algorithms, a class of probabilistic Cellular Automata that can be used to solve complex optimization problems, of which a number of examples are shown. Finally, in Chap. 9, Pan and Reggia continue the line of research that actually started of the field of Cellular Automata, that is, Cellular Automata for self-replicating structures. They give a historical overview, and discuss novel results using evolutionary algorithms to discover new self-replicating structures.

Part II contains 5 chapters showing impressive examples of applications of Cellular Automata to model complex systems in a large range of different scientific disciplines. Yu and Helbing discuss in Chap. 10 applications in Sociology, introducing migration games and studying in detail their behavior under a range of assumptions and parameters. On a quite different stance, Kusumaatmaja and Yeomans introduce in Chap. 11 the Lattice Boltzmann Method and its application to modeling complex fluids, specifically dynamics of drops and wetting dynamics.

Chowdhury, Nishinari, Schadschneider introduce in Chap. 12 single-and two-lane CA models for ant trails, and their validation against empirical data. Next, in Chap. 13, Hatzikirou and Deutsch discuss how the Lattice Gas Cellular Automaton can be applied as a model for interacting biological cells. They study pattern formation of growing cellular populations, and apply their models to simulate growth of solid tumors.

Gürdal and Zakhma show how Cellular Automata can be applied in an engineering context. In Chap. 14 they discuss in detail how Cellular Automata can be applied to topology design optimization and provide three challenging examples.

Finally Part III contains one chapter, by Talia and Naumov, on Cellular Automata software. They review a range of problem solving environments for Cellular Automata modeling and simulation, and discuss in some detail systems that facilitate parallel cellular programming.

1.9 Additional Resources

A wide number of high quality sources providing necessary starting information about cellular automata, their implementation, and their use in modeling of complex systems exist; among others there are [10, 23, 22, 25, 27, 18, 21, 12]. In the case you have no prior experience with cellular automata, it might be good to start with reading of the book of Mitchel Resnick [17] and use the StarLogo programable modeling environment [18], which is suitable for description of decentralized systems (e.g. ant colonies and market economies).

It is also recommended to study the book of Toffoli and Margolus [23] which presents a wide number of cellular automata models implemented in their specially dedicated computer with a computational environment called Cellular Automata Machine (CAM). Many more cellular automata software packages are available. We suggest to visit the web page of the IFIP Working Group 1.5 on Cellular Automata and Machines [11] and consult the review provided in Chap. 15 of this book.

A series of bi-annual cellular automata conferences [28, 20, 2] named ACRI presents every second year a wide number a valuable, up-to-date overview of cellular automata models. If you intend to use cellular automata to model a naturally observed phenomenon then this source – beside this book – might be a good starting point of your search for cellular automata models that are solving the same or similar problems.

References

1. P. Bak, *How Nature Works: The Science of Self-Organized Criticality*. (Springer, New York, NY, 1996)
2. S. Bandini, B. Chopard, M. Tomassini (eds.), Cellular Automata, 5th International Conference on Cellular Automata for Research and Industry, ACRI 2002, Geneva, Switzerland, October 9–11, 2002, Proceedings Lecture Notes in Computer Science, vol. 2493 (Springer, Heidelberg, 2002)
3. L. Berec, Techniques of spatially explicit individual-based models: Construction, simulation and mean-field analysis. Ecol. Model. **150**, 55–81 (2002)
4. N. Boccara, *Modeling Complex Systems* (Springer, Heidelberg, 2004)
5. B. Chopard, M. Droz, *Cellular Automata Modeling of Physical Systems*, (Cambridge University Press, Cambridge, 2005)
6. K. Christensen, N. Moloney, *Complexity and Criticality* (Imperial College Press, London, 2005)
7. A. Deutch, S. Dormann, *Cellular Automaton Modeling of Biological Pattern Formation* (Birkhauser, Basel, 2004)

8. M. Gardner, The fantastic combinations of John Conway's new solitaire game "life". Sci. Am. **223**, 120–123 (1970)
9. B. Hölldobler, E. Wilson, *Journey to the Ants: A Story of Scientific Exploration*, 3rd edn. (Harvard University Press, Cambridge, MA, 1995)
10. A. Ilachinski, *Cellular Automata: A Discrete Universe* (World Scientific Publishing Co. Pte. Ltd., London, 2001)
11. International Federation for Information Processing (IFIP), Working Group 1.5 on Cellular Automata and Machines, http://liinwww.ira.uka.de/ca/software/index.html: A list of software packages for Cellular Automata, http://liinwww.ira.uka.de/ca/software/index.html
12. J. Kroc, Special issue on modelling of complex systems by cellular automata 2007: Guest editors' introduction. Adv. Compl. Syst. **10**(1 supp), 1–3 (2007)
13. C. Langton, Computation at the edge of chaos: Phase transitions and emergent computation. Physica D **42**, 12–27 (1990)
14. M. Mitchell, *Nonstandard Computation*, chap. Computation in cellular automata: a selected review (VCH, Weinheim Verlagsgesellschaft, 1998) pp. 95–140
15. A. Nathan, V. Barbosa, V-like formations in flocks of artificial birds. ArXiv Computer Science e-prints (2006), http://arxiv.org/abs/cs/0611032
16. J. von Neumann, A. Burks, *Theory of Self-Reproducing Automata* (University of Illinois Press, Urbana, IL 1966)
17. M. Resnick, *Turtles, Termites, and Traffic Jams – Explorations in Massively Parallel Microworlds* (The MIT Press, Cambridge, MA, 1997)
18. M. Resnick, StarLogo – programmable environment for exploring decentralized systems flocks, traffic jams, termite and ant colonies. Tech. rep., MIT (2006), http://education.mit.edu/starlogo/
19. P.M.A. Sloot, B.J. Overeinder, A. Schoneveld, Self organized criticality in simulated correlated systems. Comput. Phys. Comm. **142**, 66–81 (2001)
20. P. Sloot, B. Chopard, A. Hoekstra (eds.), Cellular Automata, 6th International Conference on Cellular Automata for Research and Industry, ACRI 2004, Amsterdam, The Netherlands, October 25–28, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3305 (Springer, Heidelberg, 2004)
21. P.M.A. Sloot, A.G. Hoekstra, Modeling dynamic systems with cellular automata, ed. by P.A. Fishwick *Handbook of Dynamic System Modelling* chapter 21 (Chapman and Hall London, 2007)
22. T. Toffoli, Cellular automata as an alternative to (rather than an approximation of) differential equations in modelling physics. Physica **D17**, 117–127 (1984)
23. T. Toffoli, N. Margolus, *Cellular Automata Machines: A New Environment for Modeling* (MIT Press, Cambridge, MA 1987)
24. A.M. Turing, *The Essential Turing: Seminal Writings in Computing, Logic, Philosophy, Artificial Intelligence, and Artificial Life plus The Secrets of Enigma* (Oxford University Press, New York, NY, 2004)
25. G. Vichniac, Simulating physics with cellular automata. Physica **D17**, 96–116 (1984)
26. S. Wolfram, Universality and complexity in cellular automata. Physica D 1–35 (1984)
27. S. Wolfram, *A New Kind of Science* (Wolfram Media Inc., Champaign, Il 2002)
28. S.E. Yacoubi, B. Chopard, S. Bandini, (eds.) Cellular Automata, 7th International Conference on Cellular Automata, for Research and Industry, ACRI 2006, Perpignan, France, September 20–23, 2006, Proceedings Lecture Notes in Computer Science, vol. 4173 (Springer, Heidelberg, 2006)