CELLULAR AUTOMATA

Cellular automata (CA) are collections of identical, interacting, finite-state machines. CA were originally proposed by John von Neumann as models of self-reproduction. Since then, CA have been most extensively studied in the context of complex systems modeling where the evolutionary behavior of a complex system is modeled through a large number of simple, identical, interacting cells.

John Conway's game of Life (1) is perhaps the best known example of a CA. Life, which became popular in the early 70s, is a computer program that models the evolutionary interactions among a population of organisms. Each organism is identical to the others. The life and death of an organism at any time is based on some simple rules about the other organisms in its vicinity. Life delighted people around the world by exhibiting complex population wide propagative and extinctive behavior based on simple rules of engagement between an organism and its neighbors.

CELLULAR AUTOMATA EXAMPLES

Consider a one-dimensional array of cells where each cell assumes a value of either zero or one. Values in the cells change over discrete time steps. We assume that the cells wrap around so that every cell has exactly two neighbors. The *neighborhood* of a cell consists of itself and its two neighbors. The new value of a cell is computed using the current values of the cells in its neighborhood. New values of all cells are simultaneously computed using the present values of all the cells and using the same rule for each cell. Consider the following rules:

1. The new value of a cell is one only if exactly one cell in its neighborhood currently has a value of one.
2. The new value of the cell is one only if exactly one cell in its neighborhood currently has a value of one or the cell itself has a zero but both of its neighboring cells have values of one.

Figure 1 shows the evolution of values in 200 cells over 200 time steps when the above rules are used to update the values in the cells in each time step. The first rule, called

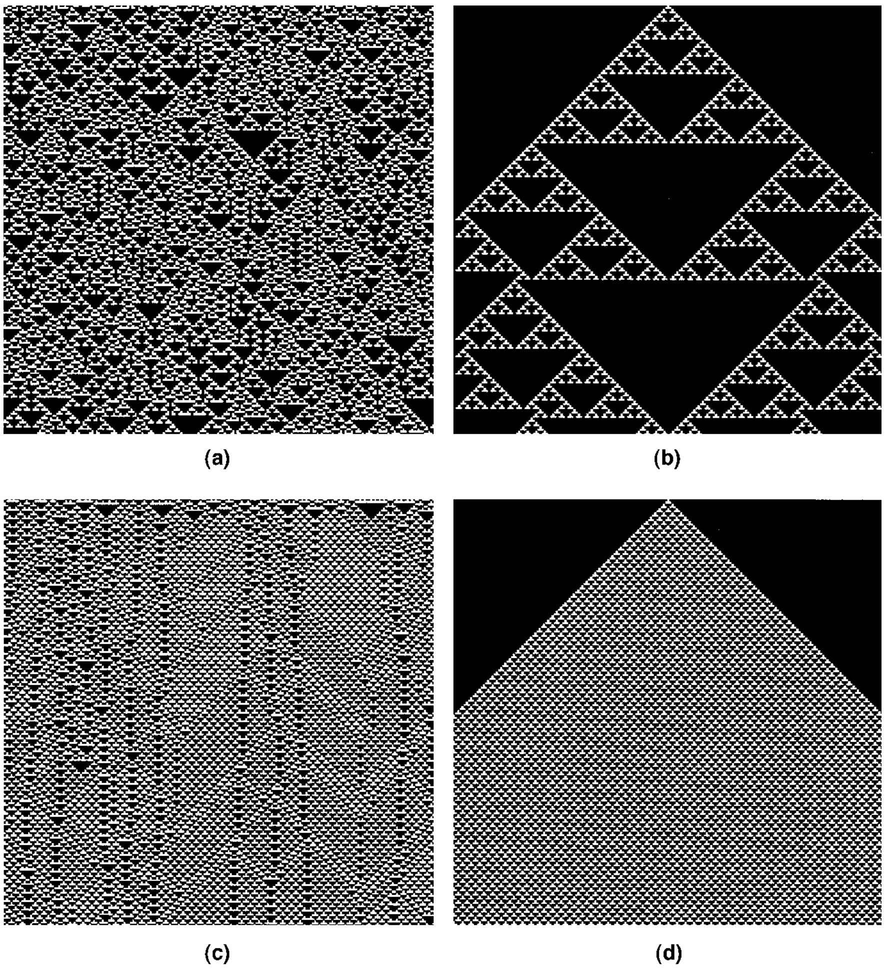

Figure 1. Evolution of a simple linear cellular automaton with 200 cells over 200 time steps. (a) Rule 22 with random initialization of the cells. (b) Rule 22 with only one cell initialized to one. (c) Rule 54 with random initialization. (d) Rule 54 with only one cell initialized to one. These figures are generated using the CA animations accessible on the world wide web at URL http://alife.santafe.edu:80/alife/topics/ca/caweb.

Rule 22, is an example of a *chaotic rule,* and the second rule, called Rule 54, is an example of a *complex rule.* Both types of rules are discussed later. Statistical properties of these cellular of automata have been extensively studied (2).

CELLULAR AUTOMATA THEORY

Basic Definitions

A cellular automaton (3) is a collection of identical *cells* identified by their integral positions in a d-dimensional Euclidean lattice $R = Z^d$, where Z denotes the set of integers. Each cell is a finite-state machine with a set of states S. The global state, also called the *configuration,* of the CA is completely characterized by the individual states of its cells. The global configuration is specified by a mapping c: $R \rightarrow S$.

Given a configuration c, the next configuration of the CA is determined by *simultaneously* determining the new state of each cell. The new state of a cell is determined by its current state and the states of some neighboring cells. The *neighborhood* of a cell is identified by a finite set $N \subset Z^d$, such that the elements of N denote the relative coordinates of the neighboring cells. By this convention, a cell at position $p \in R$ receives as its inputs the states of the cells at positions $p + n$ for all $n \in N$.

A mapping l: $N \rightarrow S$ denotes the configuration of the neighborhood of a cell and is called a *local configuration.* When the CA is in a global configuration, the local configuration observed by a cell at position $p \in R$ is denoted by $l_{c,p}$ and is given by $l_{c,p}(n) = c(p + n)$, for all $n \in N$.

Let $\{N \rightarrow S\}$ denote the set of all possible local configurations of the neighborhood of a cell. The state-transition function of a cell is specified as a mapping f: $\{N \rightarrow S\} \rightarrow S$. This mapping is called the *local rule* or the *evolution rule* of the CA.

All cells in the CA use the same neighborhood definition and the same local rule and make state transitions synchronously. A global state-transition function, also called the *global rule,* F: $\{R \rightarrow S\} \rightarrow \{R \rightarrow S\}$ is constructed from the local rule as follows: $F(c_1) = c_2$ where for all $p \in R$, $c_2(p) = f(l_{c_1,p})$. Given the local rule, the CA evolves synchronously from configuration to configuration in discrete time steps.

Variations and Generalizations

It is possible to generalize the definition of the basic CA in many different ways. Some of these generalizations are mentioned here.

Nondeterministic Cellular Automata. Specification of the local rule as a relationship instead of a function leads to the *nondeterministic CA* in which each cell selects one among several possible next states nondeterministically. In a *probabilistic CA,* the next state is chosen from among a set of possible states based on probabilities (4). In *asynchronous* cellular automata (5), a variation of the nondeterministic CA, each cell makes an arbitrary choice whether its state should be updated in an evolution step.

Tessellation Automata. Specification of a set of rules, one of which may be chosen as the local rule to use at any time step, leads to *tessellation automata* (6). The choice of the rule may be arbitrary or may depend on the time step. Another variation allows the possibility of different cells using different rules leading to *hybrid CA*.

Arrangement of Cells. The common practice is to consider finite lattice structures for the arrangement of the CA cells. Both wraparound (*periodic-boundary CA*) and nonwraparound (*null-boundary CA*) structures have been studied. Nonlattice structures considered include tori, trees, and trellises.

Classification

Wolfram (7) categorized cellular automata into four classes. Although not formally defined, this classification continues to provide an informal way to discuss properties of cellular automata.

1. *Class 1, Homogeneous CA:* Evolution in Class 1 CA leads to a unique homogeneous configuration where all cells enter and remain in the same state. Initial state information is completely lost upon reaching this homogeneous state.
2. *Class 2, Filtering CA:* Evolution in Class 2 CA leads to separated, simple, persistent structures from particular initial-state sequences. The evolved simple structure may be stable or periodic, typically with a small period.
3. *Class 3, Chaotic CA:* After a finite number of time steps, Class 3 CA exhibit chaotic or aperiodic patterns. Some chaotic CA may evolve into highly regular fractal patterns and others exhibit irregular pseudorandom configurational sequences.
4. *Class 4, Complex CA:* Class 4 automata exhibit complicated localized and propagating configurations. Although most cells reach a stable "death" state, gliding periodic structures emerge in some localities. Class 4 CA are universal computing machines, capable of exhibiting arbitrary algorithmic behavior when correctly programmed with suitable initial configurations.

Problems Studied

Local and Global Properties. Global properties are properties of the global rule F. Exploring the relationships between local rules and global properties has been a major area of CA studies. Because CA exhibit complicated overall behavior based on simple local interactions, determination of whether a global property is a consequence of a local rule (*forward or analysis problem*) and discovery of a local property to ensure a global property (*backward or synthesis problem*) are both important CA topics.

Gardens of Eden. A *Garden of Eden* of a CA is a configuration which the CA never reach except as the initial configuration. Garden of Eden states exist when the global rule is surjective. Given a local rule, deciding whether a Garden of Eden state exists for the CA is a well-studied decision problem. This problem is decidable for one-dimensional CA and undecidable for two-dimensional CA.

Reversibility. A CA is said to be reversible if for any configuration it is possible to determine correctly and uniquely what the previous configuration was. Not all CA are reversible. For example, the one-dimensional automata discussed earlier are not generally reversible. However, reversible automata are important models of computation because reversible machines avoid the fundamental lower bounds on power dissipation associated with irreversibility of conventional logic. For this reason reversible CA received special attention (8).

Self-Reproduction. CA were originally proposed (9) as models of self-replicating machines. One common way of studying self-reproduction is to observe whether the initial configuration repeats itself after a finite number of evolutionary steps. Another way is to observe whether certain local patterns repeat themselves infinitely often. For example, if the states of a cell are represented by integers modulo k for some prime number k and module-k addition of the neighborhood states is used as the local rule, then any finite pattern of nonzero integers embedded in a "sea" of zeros repeats itself after k^m steps for some large m (10).

Universality. A computing device is said to be *universal* if it can be programmed to compute any arbitrary computable function including reproducing itself. Some cellular automata are universal computing devices and are equivalent to Turing machines. For example, Conway's game of Life CA have the property of computational universality. A very simple two-dimensional CA with two states and nine neighborhood cells is shown to be universal in Ref. 11.

Algorithms and Complexity. One of the original themes of CA research has been to study CA as models for analyzing computability. Can a CA implement an algorithm for a computable function and how much time/space does CA require to solve a problem? These questions have been studied extensively, especially in the area of formal language recognition using CA (12,13).

APPLICATIONS OF CELLULAR AUTOMATA

CA have been studied extensively in numerous application domains ranging from biological evolution to computer arithmetic. Some of these application areas are reviewed here.

Complex Systems Modeling

Modeling complex dynamical systems through a large number of simple interacting agents continues to be one of the most important applications of CA (14,15). For example, CA have been used to model urban development and ecological systems. In Ref. 16, the effect of forest fires and dispersal on spatial patterns is modeled. In Ref. 17, a CA model of competition between species of grass is modeled.

Image Processing

Two-dimensional cellular automata have been used for image processing applications (18,19). Class 2 CA are considered suitable for filtering applications in digital signal processing. Several picture processing applications are described in Ref. 20. Many CA-motivated array processing computers were developed over the years. Notable among these are the Cellular Logic Image Processor (21), Massively Parallel Processor (22), and the Connection Machine (23). Many *systolic architectures*—a class of VLSI architectures with regular structure—are also inspired by the CA (24).

Biology

Cellular automata have become an indispensable tool in studying biology (25). Growth and behavior of various organisms have been modeled using CA. For example, vertebrate skin patterns (26), heart fibrillation (27), tumor growth (28), pigmentation patterns on shells (29), and the dynamics of task-switching in ants (30) are among the topics studied using CA.

Fluid Flow and Diffusion

Lattice gas automata (LGA) are a class of probabilistic CA used to study viscous fluid flow (31). Reaction diffusion systems modeling is discussed in Ref. 32. Lattice gas models show that it is possible to effectively model continuous behavior, traditionally expressed by partial differential equations, by using discrete evolutionary structures.

VLSI Hardware Design and Test

CA have been extensively studied in the context of VLSI testing. Test pattern generation (pseudorandom, pseudoexhaustive, and deterministic test patterns), signature analysis, testable logic synthesis have all been tackled using CA. Ref. 33 gives an in-depth study and survey of cellular automata for VLSI testing and related applications. With the advent of field-programmable gate arrays (FPGA) suitable for implementing evolving computations, new interest in implementing CA in hardware is emerging (34).

TRENDS IN CELLULAR AUTOMATA STUDIES

Study of the relationships between cellular automata and other models of computation such as neural networks, fractals (35), systolic architectures and genetic algorithms is becoming an important aspect of CA studies. There is growing interest in using CA models to implement evolvable hardware (34). CA also are being used as one of the basic tools in studying artificial life.

BIBLIOGRAPHY

1. M. Gardner, The fantastic combinations of John Conway's new solitaire game 'Life,' *Sci. Amer.,* **223** (4): 120–123, 1970.
2. S. Wolfram, *Cellular Automata and Complexity,* Reading, MA: Addison-Wesley, 1994.
3. M. Kutrib, R. Vollmar, and Th. Worsch, Introduction to the special issue on cellular automata, *Parallel Comput.,* **23**: 1567–1576, 1997.
4. G. Grinstein, C. Jayaprakash, and Y. He, Statistical mechanics of probabilistic cellular automata, *Phys. Rev. Lett.,* **55** (23): 2527–2530, 1985.
5. G. Pighizzini, Asynchronous automata versus asynchronous cellular automata, *Theor. Comput. Sci.,* **132**: 179–207, 1994.
6. H. Yamada and S. Amoroso, Tesselation automata, *Inf. Control,* **14**: 299–317, 1969.
7. S. Wolfram, Universality and complexity in cellular automata, *Physica D,* **10**: 1–35, 1984.
8. N. Margolus, Physics-like models of computation, *Physica D,* **10**: 81–95, 1984.
9. J. von Neumann, in A. W. Burks (ed.), *Theory of Self-Reproduction Automata,* Urbana, IL: University of Illinois Press, 1966.
10. S. Amoroso and G. Cooper, Tesselation structure for reproduction of arbitary patterns, *J. Comput. Syst. Sci.,* **5**: 455–464, 1971.
11. E. R. Banks, Universality in cellular automata, *Proc. Symp. Found. Comput. Syst. Sci.,* 1970, pp. 194–215.
12. A. R. Smith, III, Cellular automata and formal languages, *Proc. 11th Annu. IEEE Symp. Switch. Autom. Theory,* 1970, pp. 216–224.
13. M. Matamala, Alternation on cellular automata, *Theor. Comput. Sci.,* **180**: 229–241, 1997.
14. P. Manneville et al. (eds.), *Cellular Automata and Modeling of Complex Physical Systems,* Berlin: Springer, 1989.
15. J. Demongeot, E. Goles, and M. Tchuente, *Dynamical Systems and Cellular Automata,* London: Academic Press, 1985.
16. D. G. Green, Cellular automata models of crown-of-thorns outbreaks, *Lect. Notes Biomath.,* **88**: 169–188, 1990.
17. J. Silvertown et al., Cellular automaton models of interspecific competition for space—the effect of pattern on process, *J. Ecol.,* **80**: 527–534, 1992.
18. A. Rosenfeld, *Picture Languages,* New York: Academic Press, 1979.
19. S. R. Sternberg, *Language and Architecture for Parallel Image Processing,* Amsterdam: North Holland, 1980.
20. K. Preston and M. J. B. Duff, *Modern Cellular Automata: Theory and Applications,* New York: Plenum, 1984.
21. M. J. B. Duff et al., A cellular logic array for image processing, *Pattern Recognition,* **5**: 229–234, 1973.
22. J. L. Potter, Image processing on a massively parallel processor, *Computer,* **16** (1): 62–67, 1983.
23. W. D. Hillis, The connection machine: A computer architecture based on cellular automata, *Physica D,* **10**: 213, 1984.
24. S. Y. Kung, *VLSI Array Processors,* Englewood Cliffs, NJ: Prentice-Hall, 1988.
25. B. Ostrovsky, M. A. Smith, and Y. Bar-Yam, Applications of parallel computing to biological problems, *Annu. Rev. Biophys. Biomol. Struct.,* **24**: 239–267, 1995.
26. D. Young, A local activator-inhibitor model of vertebrate skin patterns, *Math. Biosci.,* **72**: 51, 1984.
27. A. Burks, Cellular automata and natural systems, in *Cybernetics and Bionics,* Munich: Oldenbourg, 1974.
28. W. Duchting and T. Vogelsaenger, Aspects of modeling and simulating tumor growth and treatment, *J. Cancer Res. Clin. Oncol.,* **105**: 1983.
29. H. Meinhardt, *The Algorithmic Beauty of Sea Shells,* Berlin: Springer, 1995.
30. D. M. Gordon, B. C. Goodwin, and L. E. H. Trainor, A parallel distributed model of the behavior of ant colonies, *J. Theor. Biol.,* **156**: 293–307, 1992.
31. U. Frisch, B. Haslacher, and Y. Pomeau, Lattice-gas automata for the Navier-Stokes equation, *Phys. Rev. Lett.,* **56**: 1505–1508, 1986.

32. J. R. Weimar, Cellular automata for reaction diffusion systems, *Parallel Comput.,* **23** (11): 1699–1715, 1997.

33. P. Pal Chaudhuri et al., *Additive Cellular Automata: Theory and Applications,* Los Alamitos, CA: IEEE Computer Society Press, 1997, Vol. 1.

34. T. Higuchi, M. Iwata, and W. Liu (eds.), *Evolvable Systems: From Biology to Hardware,* Lect. Notes Comput. Sci., No. 1259, Berlin: Springer, 1996.

35. H.-O. Peitgen and D. Saupe (eds.), *The Science of Fractal Images,* New York: Springer, 1988.

Reading List

A. Adamatzky, *Identification of Cellular Automata,* London: Taylor and Francis, 1994.

E. F. Codd, *Cellular Automata,* New York: Academic Press, 1968.

S. Forrest (ed.), *Emergent Computation: Self-Organizing, Collective and Cooperative Phenomena in Natural and Artificial Computing Networks,* Cambridge, MA: MIT Press, 1991.

T. Toffoli and N. Margolus, *Cellular Automata Machines,* Cambridge, MA: MIT Press, 1987.

S. Wolfram, *Theory and Applications of Cellular Automata,* Singapore: World Scientific, 1986.

RANGA R. VEMURI
University of Cincinnati