Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 334 (2005) 3–33
Theoretical Computer Science
www.elsevier.com/locate/tcs

Theory of cellular automata: A survey☆

Jarkko Kari

Department of Mathematics, University of Turku, FIN-20014, Turku, Finland

Received 1 October 2004; accepted 12 November 2004

Communicated by G. Rozenberg

Abstract

This article surveys some theoretical aspects of cellular automata CA research. In particular, we discuss classical and new results on reversibility, conservation laws, limit sets, decidability questions, universality and topological dynamics of CA. The selection of topics is by no means comprehensive and reflects the research interests of the author. The main goal is to provide a tutorial of CA theory to researchers in other branches of natural computing, to give a compact collection of known results with references to their proofs, and to suggest some open problems.

Keywords: Cellular automata; Reversible computation; Decidability; Discrete dynamical system

1. Introduction

Cellular automata (CA) are among the oldest models of natural computing, dating back over half a century. The first CA studies by John von Neumann in the late 1940s were biologically motivated [10,69]: the goal was to design self-replicating artificial systems that are also computationally universal. Von Neumann clearly wanted to investigate synthetic computing devices analogous to human brain in which the memory and the processing units are not separated from each other, that are massively parallel and that are capable of repairing and building themselves given the necessary raw material. Following suggestions by S. Ulam, he envisioned a discrete universe consisting of a two-dimensional mesh of finite state machines, called cells, interconnected locally with each other. The cells change

☆ Research supported by the Academy of Finland Grant 54102.
E-mail address: jkari@utu.fi.

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2004.11.021

their states synchronously depending on the states of some nearby cells, the neighbors, as determined by a local update rule. All cells use the same update rule so that the system is homogeneous like many physical and biological systems. These cellular universes are now known as CA. Von Neumann's line of research on self-replicating CA was continued later by other authors, see e.g. [14].

CA possess several fundamental properties of the physical world: they are massively parallel, homogeneous and all interactions are local. Other physical properties such as reversibility and conservation laws can be programmed by choosing the local update rule properly. It is therefore not surprising that physical and biological systems have been successfully simulated using CA models [13]. Discrete simulation of fluid flows using CA has even become a field of its own in which CA models are called lattice gases. See, for example, [27,31] for two fundamental lattice gas models. Other classic CA simulations of physical systems include Ising spin models [68] and diffusion phenomena, e.g. [72].

The physical nature of CA may have even much greater practical importance when applied to the opposite direction, that is, when using the physics to simulate CA. Since many CA are computationally universal—and some very simple CA have this property—then perhaps we eventually succeed to harness physical reactions of microscopic scale to execute massively parallel computations by running a computationally universal CA. This requires that the simulated CA obeys the rules of physics, including reversibility and conservation laws. While such truly programmable matter may be decades away, its potential is great [66].

In this tutorial article, we review basic theoretical results concerning CA in computer science. The field is very broad and the research is lively, so it is only possible to cover certain aspects of the field. The selected topics reflect the research interests of the author, and they include reversibility, conservation laws, decidability questions, computational universality and limit behavior. For other topics see, for example, books [13,30,66,75]. We start by defining basic concepts.

2. Preliminaries

In this paper, we consider synchronous CA only, where the underlying topology is an infinite rectangular grid. The cells are hence the squares of an infinite d-dimensional checker board, addressed by $\mathbb{Z}^d$. This is called a d-dimensional CA. The one-, two-and three-dimensional cases are most common, and as we see later, one-dimensional CA behave in some respects differently from the higher-dimensional ones.

2.1. Basic definitions

The states of the automaton come from a finite *state set* S. At any given time, the *configuration* of the automaton is a mapping $c : \mathbb{Z}^d \longrightarrow S$ that specifies the states of all cells. The set $S^{\mathbb{Z}^d}$ of all configurations is denoted by $\mathcal{C}(d, S)$, or briefly $\mathcal{C}$ when d and S are known from the context. Constant functions are called *homogeneous* configurations.

The cells change their states synchronously at discrete time steps. The next state of each cell depends on the current states of the neighboring cells according to an update rule. All cells use the same rule, and the rule is applied to all cells at the same time. The neighboring

cells may be the nearest cells surrounding the cell, but more general neighborhoods can be specified by giving the relative offsets of the neighbors. Let $N = (\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n)$ be a vector of n distinct elements of $\mathbb{Z}^d$. Then the *neighbors* of a cell at location $\vec{x} \in \mathbb{Z}^d$ are the n cells at locations

$$\vec{x} + \vec{x}_i \quad \text{for } i = 1, 2, \ldots, n.$$

The *local rule* is a function $f : S^n \longrightarrow S$ where n is the size of the neighborhood. State $f(a_1, a_2, \ldots, a_n)$ is the state of a cell whose n neighbors were at states $a_1, a_2, \ldots, a_n$ one time step before. This update rule then determines the global dynamics of the CA: Configuration c becomes in one time step the configuration e where, for all $\vec{x} \in \mathbb{Z}^d$,

$$e(\vec{x}) = f(c(\vec{x} + \vec{x}_1), c(\vec{x} + \vec{x}_2), \ldots, c(\vec{x} + \vec{x}_n)).$$

We say that $e = G(c)$, and call $G : \mathcal{C} \longrightarrow \mathcal{C}$ the *global transition function* of the CA.

In summary, CA are dynamical systems that are homogeneous and discrete in both time and space, and that are updated locally in space. A d-dimensional CA is specified by a triple (S, N, f) where S is the state set, $N \in (S^{\mathbb{Z}^d})^n$ is the neighborhood vector, and $f : S^n \longrightarrow S$ is the local update rule. We usually identify a CA with its global transition function G, and talk about CA function G, or simply CA G. In algorithmic questions G is, however, always specified using the three finite items S, N and f.

2.2. Neighborhoods

Neighborhoods commonly used in CA are the *von Neumann* neighborhood N_{vN} and the *Moore* neighborhood N_M. The von Neumann neighborhood contains relative offsets $\vec{y}$ that satisfy $\|\vec{y}\|_1 \leqslant 1$, where

$$\|(y_1, y_2, \ldots, y_d)\|_1 = |y_1| + |y_2| + \cdots + |y_d|$$

is the Manhattan norm. This means that cell in location $\vec{x}$ has $2d + 1$ neighbors: the cell itself and the cells at locations $\vec{x} \pm \vec{e}_i$ where $\vec{e}_i = (0, \ldots, 0, 1, 0, \ldots 0)$ is the ith coordinate unit vector. The Moore neighborhood contains all vectors $\vec{y} = (y_1, y_2, \ldots, y_d)$ where each y_i is -1, 0 or 1, that is, all $\vec{y} \in \mathbb{Z}^d$ such that $\|\vec{y}\|_\infty \leqslant 1$ where

$$\|(y_1, y_2, \ldots, y_d)\|_\infty = \max\{|y_1|, |y_2|, \ldots, |y_d|\}$$

is the max-norm. Every cell has 3^d Moore neighbors. Fig. 1 shows the von Neumann and Moore neighborhoods in the case $d = 2$.

Generalizing the Moore neighborhood, we obtain *radius-r* CA, for any positive integer r. In radius-r CA the relative neighborhood consists of vectors $\vec{y}$ such that $\|\vec{y}\|_\infty \leqslant r$. Moore neighborhood is of radius 1. By *radius-*$\frac{1}{2}$ neighborhood we mean the neighborhood that contains the offsets $\vec{y} = (y_1, y_2, \ldots, y_d)$ where each y_i is 0 or 1. Fig. 2 shows the trellis whose rows are the configurations of a one-dimensional, radius-$\frac{1}{2}$ CA at consecutive time steps, and the rows are shifted to make the neighborhood look symmetric.

A one-dimensional, radius-$\frac{1}{2}$ CA is also called one-way, or OCA for short. Since the neighborhood $(0, 1)$ does not contain negative elements, information cannot flow to the positive direction, unless the cells are shifted as in Fig. 2.

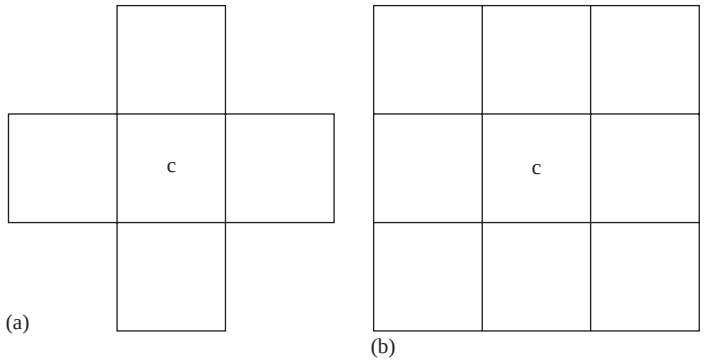


Fig. 1. Two-dimensional (a) von Neumann and (b) Moore neighbors of cell c.

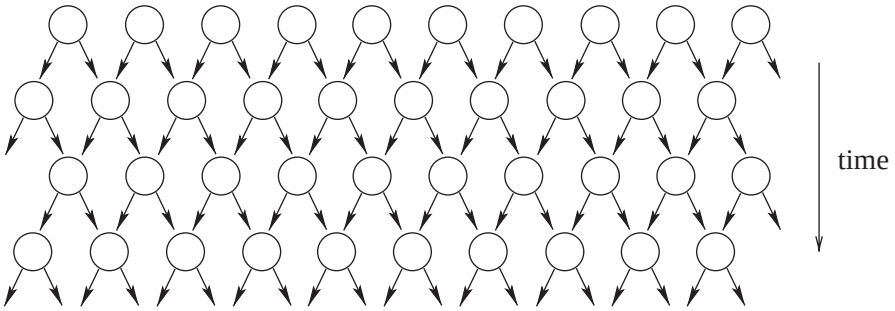


Fig. 2. Dependencies in one-dimensional, radius-$\frac{1}{2}$ CA.

2.3. Finite and periodic configurations

The *shift functions* are particularly simple CA that translate the configurations one cell down in one of the coordinate directions. More precisely, for each dimension $i = 1, 2, \ldots, d$ there is the corresponding shift function σ_i whose neighborhood contains only the unit coordinate vector $\vec{e}_i$ and whose local rule is the identity function. The one-dimensional shift function is the left shift $\sigma = \sigma_1$. *Translations* are compositions of shift functions. Translation $\tau_{\vec{y}}$ by vector $\vec{y}$ is the CA with neighborhood $(\vec{y})$ and the identity local rule.

Sometimes one state $q \in S$ is specified as a *quiescent state*. It should be *stable*, which means that $f(q, q, \ldots, q) = q$. The quiescent configuration Q is the configuration where all cells are quiescent: $Q(\vec{x}) = q$ for all $\vec{x} \in \mathbb{Z}^d$. A configuration $c \in S^{\mathbb{Z}^d}$ is called *finite* if only a finite number of cells are non-quiescent, i.e. the support

$$\{\vec{x} \in \mathbb{Z}^d \mid c(\vec{x}) \neq q\}$$

is finite. Let us denote by $\mathcal{C}_{\mathrm{F}}(d, S)$, or briefly $\mathcal{C}_{\mathrm{F}}$, the subset of $S^{\mathbb{Z}^d}$ that contains only the finite configurations. Because of stability of q, finite configurations remain finite in the evolution of the CA, so the restriction G_{F} of G on the finite configurations is a function $\mathcal{C}_{\mathrm{F}} \longrightarrow \mathcal{C}_{\mathrm{F}}$.

A *periodic configuration*, or more precisely, a spatially periodic configuration is a configuration that is invariant under d linearly independent translations. This is equivalent to the existence of d positive integers $t_1, t_2, \ldots, t_d$ such that $c = \sigma_i^{t_i}(c)$ for every $i = 1, 2, \ldots, d$, that is,

$$c(\vec{x}) = c(\vec{x} + t_i \vec{e}_i)$$

for every $\vec{x} \in \mathbb{Z}^d$ and every $i = 1, 2, \ldots, d$. Let us denote by $\mathcal{C}_P(d, S)$, or briefly $\mathcal{C}_P$, the set of periodic configurations. CA are homogeneous in space and consequently they preserve periodicity of configurations. The restriction G_P of G on the periodic configurations is hence a function $\mathcal{C}_P \longrightarrow \mathcal{C}_P$.

Finite configurations and periodic configurations are used in effective simulations of CA on computers. Periodic configurations are often referred to as the periodic boundary conditions on a finite cellular array. For example, in the case $d = 2$ this is equivalent to running the CA on a torus that is obtained by "gluing" together the opposite sides of a rectangle. One should, however, keep in mind that the behavior of a CA can be quite different on finite, periodic and general configurations, so experiments done with periodic boundary conditions may be misleading.

2.4. Other basic concepts and properties

One must take care not to confuse (spatially) periodic configurations with temporally periodic configurations. Configuration c is *temporally periodic* for CA G if $G^k(c) = c$ for some $k \geqslant 1$. If $G(c) = c$ then c is a *fixed point*. Every CA has a temporally periodic configuration that is homogeneous: this follows from the facts that there are finitely many homogeneous configurations and that CA functions preserve homogeneity. Configuration c is *eventually periodic* if it evolves into a temporally periodic configuration, that is, if $G^m(c) = G^n(c)$ for some $m \neq n$. This is equivalent to the property that the (*forward*) *orbit* $c, G(c), G^2(c), \ldots$ is finite. Every spatially periodic configuration is eventually periodic.

CA G is called *nilpotent* if $G^n(\mathcal{C})$ is a singleton set for sufficiently large n, that is, if there is a configuration c and number n such that $G^n(e) = c$ for all configurations e. Since homogeneous configurations remain homogeneous we immediately see that configuration c has to be homogeneous and a fixed point.

The *phase space* of CA G is the infinite directed graph whose nodes are the configurations, and there is an edge from c to e if and only if $G(c) = e$.

Let G_1 and G_2 be two given CA functions with the same state set and the same dimension d. The *composition* $G_1 \circ G_2$ is also a CA function, and the composition can be formed effectively. If N_1 and N_2 are neighborhoods of G_1 and G_2, respectively, then a neighborhood of $G_1 \circ G_2$ consists of vectors $\vec{x} + \vec{y}$ for all $\vec{x} \in N_1$ and $\vec{y} \in N_2$.

The equivalence of two given CA G_1 and G_2 is decidable: if the neighborhoods N_1 and N_2 are the same then the local rules must be identical, and if the neighborhoods are different then one can take the union of the two neighborhoods and test whether the two CA agree on the expanded neighborhood.

Sometimes it happens that $G_1 \circ G_2 = G_2 \circ G_1 = id$ where id is the identity function. Then CA G_1 and G_2 are called *reversible* and G_1 and G_2 are the *inverse automata* of each other. One can effectively decide whether two given CA are inverses of each other.

This follows from the effectiveness of the composition and the decidability of the CA equivalence. Reversible CA are studied in depth Section 4.

2.5. Elementary CA

One-dimensional CA with state set $\{0, 1\}$ and the nearest neighbor neighborhood $(-1, 0, 1)$ are called *elementary*. There are $2^3 = 8$ possible patterns inside the neighborhood of a cell, each of which may be mapped into 0 or 1. Hence, there are $2^8 = 256$ different elementary CA. Some of them are identical up to renaming the states or reversing right and left, so the number of essentially different elementary rules is smaller, only 88.

Elementary rules were extensively studied and empirically classified by Wolfram [73–75]. He introduced a naming scheme that has since become standard: Each elementary rule is specified by an eight-bit sequence

$$f(111)\ f(110)\ f(101)\ f(100)\ f(011)\ f(010)\ f(001)\ f(000),$$

where f is the local update rule of the CA. The bit sequence is the binary expansion of an integer in the interval $0 \ldots 255$, called the *Wolfram number* of the CA [73]. For example, the famous rule 110 is the elementary CA where

$$f(111) = 0, \quad f(110) = 1, \quad f(101) = 1, \quad f(100) = 0,$$

$$f(011) = 1, \quad f(010) = 1, \quad f(001) = 1, \quad f(000) = 0,$$

obtained from the binary expansion $110 = (01101110)_b$.

The numbering scheme is easily generalized to larger neighborhoods and state sets.

Examples of one-dimensional CA dynamics are often depicted as *space–time* diagrams. Horizontal rows of a space–time diagram are consecutive configurations. The top row is the initial configuration. For example, Fig. 3 shows space–time diagrams of rule 110 at two different scales, where black denotes state 1 and white denotes state 0. Notice that the diagrams drawn are finite portions of a diagram that extends to infinity to the left, to the right and down. At the fine-grained plot one can clearly observe signals and collisions of signals creating new signals.

In [74], Wolfram experimented with elementary CA rules starting from random initial configurations, and based on the types of space–time diagrams observed he classified the rules into four categories:

(W1) almost all initial configurations lead to the same uniform fixed point configuration,
(W2) almost all initial configurations lead to a periodically repeating configuration,
(W3) almost all initial configurations lead to essentially random looking behavior,
(W4) Localized structures with complex interactions emerge.

Fig. 4 shows examples of typical space–time diagrams in each class. Wolfram conjectured that class 4 CA are computationally universal.

The classification due to Wolfram is vague, and it was later formalized by Culik and Yu [17], and they also proved that their classification is undecidable: there is no algorithm to determine in which class a given one-dimensional CA belongs. Consider CA with quiescent

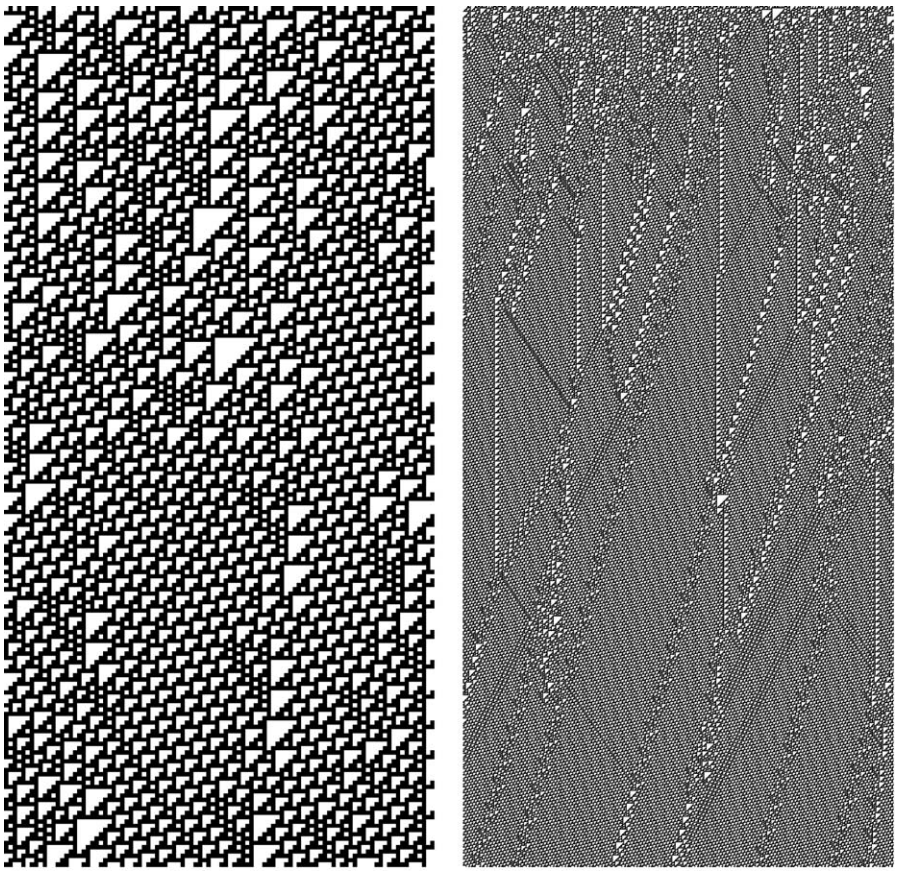


Fig. 3. Space–time diagrams of rule 110 at two different levels of magnification.

state q. The Culik–Yu classes are defined by the following properties. CA belongs to the lowest class whose defining property is satisfied:

(CY1) all finite configurations evolve into the quiescent configuration Q, that is, for every $c \in \mathcal{C}_F$ there exists $n \geqslant 1$ such that $G^n(c) = Q$,

(CY2) all finite configurations are eventually periodic, that is, for every $c \in \mathcal{C}_F$ there exist m and n, $m \neq n$, such that $G^m(c) = G^n(c)$,

(CY3) there exists an algorithm that determines if a given finite configuration belongs to the orbit of another given finite configuration, that is, it is decidable for given $c, e \in \mathcal{C}_F$ whether $G^n(c) = e$ for some $n \geqslant 1$,

(CY4) no restriction.

Note class (1) is not equivalent to nilpotency: every nilpotent CA belongs to class (1) but it has also non-nilpotent members, e.g. rule 128 in which the quiescent state 0 spreads killing all 1's.

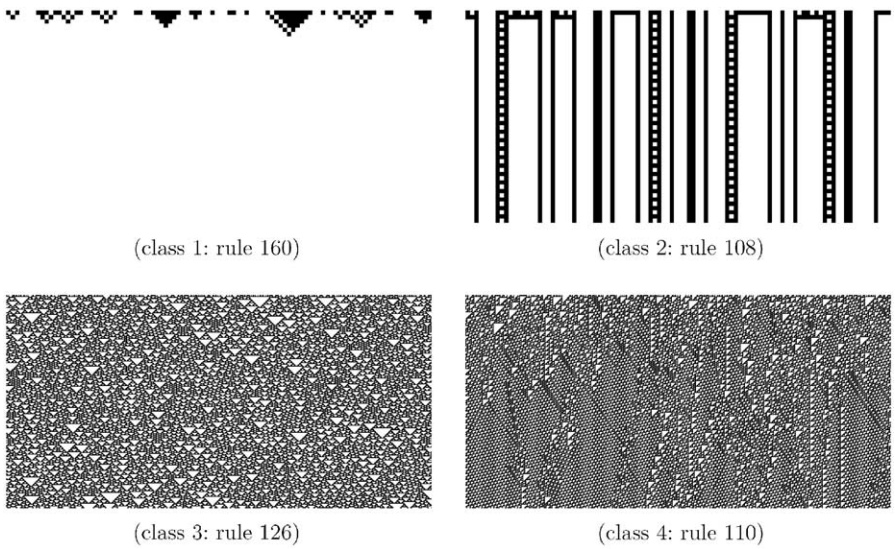


Fig. 4. Space–time diagrams of sample CA from each of the four Wolfram classes.

2.6. Well-known examples: Game of Life and rule 110

The most famous CA of all times is the *Game of Life* by Conway [6,15,29]. It is two-dimensional, uses the Moore neighborhood and has only two states called "life" and "no-life". Cells with "life" are called living. The local update rule simulates artificial life: a living cell stays alive if and only if there are exactly two or three living cells in the eight surrounding cells. Fewer than two living neighbors causes death by isolation, more than three living neighbors by overcrowding. A non-living cell becomes alive if it has precisely three living neighbors—each organism has three parents in this strange artificial life form. Let GOL denote the global transition function of the Game of Life.

What makes the Game of Life exciting are the various life-like objects it supports. These objects quickly emerge when the Game of Life is started at a random initial configuration. Different categories of objects include

- *Still life*: Finite fixed points of the update rule, that is, configurations $c \in \mathcal{C}_F$ such that $GOL(c) = c$.
- *Oscillator*: Temporally periodic finite configurations, that is, configurations $c \in \mathcal{C}_F$ such that $GOL^k(c) = c$ for some $k \geqslant 2$.
- *Gliders*: Finite configurations $c \in \mathcal{C}_F$ such that $GOF^k(c) = \tau(c)$ for some $k \geqslant 1$ and translation τ.
- *Glider guns*: Finite configurations that—like oscillators—periodically return back to the initial state, and in addition, during each period one or more gliders are emitted.

Fig. 5 shows an example from each object category. Emerging GOL configurations typically consist of several of such objects, and during the evolution the objects interact with each other through collisions with gliders and other moving structures. Collisions

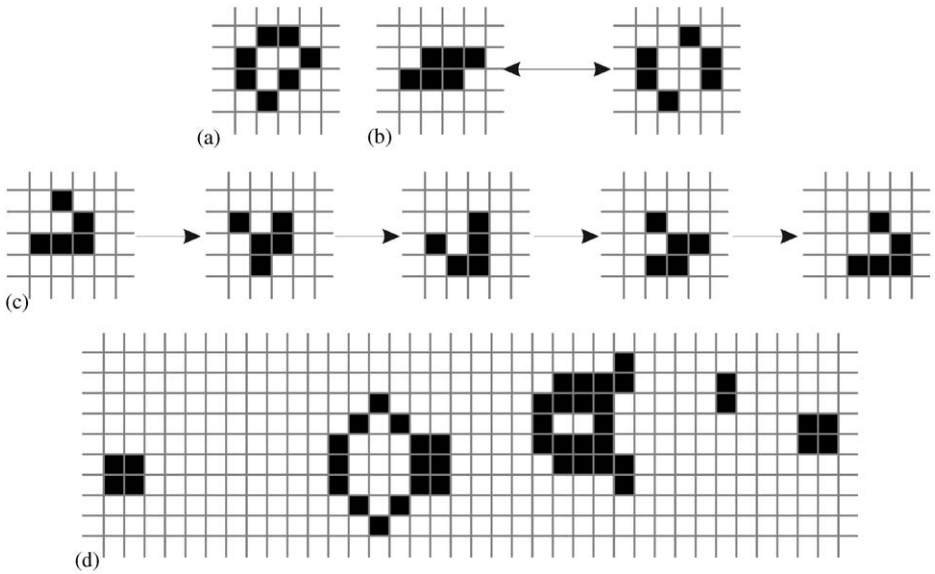


Fig. 5. Sample Game of Life objects: (a) still life, (b) period two oscillator, (c) period four glider and (d) period 30 glider gun.

create new objects which in turn participate in interactions, leading to extraordinary complexity.

We say that a finite configuration dies if it eventually becomes the quiescent configuration Q. Note that gliders in GOL are analogous to the complicated localized structures, or signals, that emerge in class 4 elementary CA. In line with Wolfram's conjecture, it has been proved that for any given Turing machine M one can effectively construct a finite GOL configuration that dies if and only if machine M halts on the blank tape. This proves the following theorem:

Theorem 1 (*Berlekamp et al. [6]*). *Game of Life is computationally universal. It is undecidable whether a given finite configuration dies.*

Let us next briefly discuss the elementary rule 110, examples of whose space–time diagrams were depicted in Fig. 3. Rule 110 is in Wolfram class 4, and Wolfram conjectured in the 1980s that it is computationally universal [74]. Recently this result was established by him and Cook [76]. In the proof a variety of localized structures, or signals, are used to encode information. Collisions of these signals are then employed to perform logic operations in the same spirit in which the gliders and their collisions were used in GOL. One should, however, note that in one-dimensional CA such as rule 110 there are technical difficulties not present in GOL due to the fact that in two dimensions it is much easier to make signals cross each other.

Theorem 2 (*M. Cook, S.Wolfram [76]*). *Rule* 110 *is computationally universal.*

We leave the exact form of universality of rule 110 undefined here. We return to different forms of universality later in Section 6. We finish this section by posing the open problem about the universality status of another elementary CA that is in Wolfram class 4:

Open problem 1. *Is the elementary CA rule* 54 *computationally universal*?

2.7. Topology and CA dynamics

A seminal paper in the topological investigation of CA is by Hedlund [33]. This paper is remarkable in several ways. It also marks the beginning of symbolic dynamics, the study of bi-infinite words and the shift function.

Let us define a topology on the configuration space $S^{\mathbb{Z}^d}$. The topology we use is the Cantor topology, obtained as the infinite power of the discrete topological space S. This topology is compact by Tychonoff's theorem. The topology is also metric: it is induced by the metric where the distance between two different configurations c and e is

$$d(c, e) = \alpha\left(\min\{\|\vec{x}\|_\infty \mid c(\vec{x}) \neq e(\vec{x})\}\right),$$

where $\alpha(x) = (\frac{1}{2})^x$. Replacing the max-norm $\|\cdot\|_\infty$ by the Manhattan norm $\|\cdot\|_1$ or the Euclidean norm $\|\cdot\|_2$ does not change the topology, nor does replacing $\alpha(x)$ by any other strictly decreasing function. In these metrics two configurations are close to each other if they agree with each other within a large region around the origin.

Balls in the metric d are called *cylinder sets*. They form a basis of the topology. Radius r cylinder containing configuration c is the set

$$\mathrm{Cyl}(c, r) = \{e \in S^{\mathbb{Z}^d} \mid e(\vec{x}) = c(\vec{x}) \text{ whenever } \|\vec{x}\|_\infty \leqslant r\}$$

of configurations that agree with c at all cells whose coordinates are within distance r from 0. For every fixed r there are finitely many radius r cylinders, and these cylinders are disjoint. Hence the radius r cylinders partition $S^{\mathbb{Z}^d}$. It follows that each cylinder is *clopen*, that is, both open and closed in the topology. The complement of a radius r cylinder is namely the union of the other radius r cylinders.

It is easy to see that CA are continuous in this topology. Trivially they commute with the shift functions σ_j, that is,

$$\sigma_j \circ G = G \circ \sigma_j$$

for every CA G and $j = 1, 2, \ldots, d$. The converse also holds. This is the Curtis–Hedlund–Lyndon theorem:

Theorem 3 (*Hedlund [33]*). *A function $G : S^{\mathbb{Z}^d} \longrightarrow S^{\mathbb{Z}^d}$ is the global transition function of a* CA *if and only if*
(i) *G is continuous, and*
(ii) *G commutes with the shifts σ_j.*

If G is a reversible CA function then $G : \mathcal{C} \longrightarrow \mathcal{C}$ is by definition a bijection. Conversely, every CA function G that is bijective is reversible. Indeed, it's inverse function clearly

commutes with the shift. The inverse function is also continuous because the space $\mathcal{C}$ is compact, and therefore it is a CA function. We have

Corollary 1 (*Hedlund [33]*). *A CA G is reversible if and only if it is a bijection.*

In symbolic dynamics literature it is therefore customary to call reversible CA *automorphisms* of the shift dynamical system. CA are termed *endomorphisms*.

2.8. Wang tiles

Wang tiles were introduced by logician Wang in 1961 [70]. They are relevant to CA theory for several reasons. Some decision problems concerning CA can be formulated as tiling problems, and the famous undecidability results concerning Wang tiles can then be employed to establish undecidability results in CA. Aperiodic tiles can be used to provide interesting examples of two-dimensional CA. Wang tiles are also used in one-dimensional CA where space–time diagrams can be viewed as tilings and this provides insight to the dynamics of the CA.

A *Wang tile* t is a unit square with colored edges. Let us denote by t_{N}, t_{E}, t_{S} and t_{W} the colors of the north, east, south and west edges of tile t, respectively. A *tile set* T is a finite collection of Wang tiles. A *Wang tiling* with T is a mapping $t : \mathbb{Z}^2 \longrightarrow T$, that is, copies of tiles in T are placed at integer lattice points, without rotating or flipping the tiles. Tiling t is valid at point $(x, y) \in \mathbb{Z}^2$ if the colors of tile $t(x, y)$ match with the colors of the neighboring tiles, that is, if

$$t(x, y)_{\mathrm{N}} = t(x, y+1)_{\mathrm{S}},$$

$$t(x, y)_{\mathrm{S}} = t(x, y-1)_{\mathrm{N}},$$

$$t(x, y)_{\mathrm{E}} = t(x+1, y)_{\mathrm{W}},$$

and

$$t(x, y)_{\mathrm{W}} = t(x-1, y)_{\mathrm{E}}.$$

Tiling t is *valid* if it is valid at every point $(x, y) \in \mathbb{Z}^2$. We say that tile set T admits a valid tiling if at least one valid tiling exists. It follows from the compactness discussed in the previous section that if a tile set admits valid tilings of arbitrarily large squares then it admits a valid tiling of the entire infinite plane.

The tiling question is the decision problem to determine if a given tile set T admits at least one valid tiling. The question was proved undecidable by R. Berger in 1966:

Theorem 4 (*Berger [5] and Robinson [58]*). *It is undecidable whether a given finite tile set T admits a valid tiling.*

Analogously to configurations, a tiling t is called *periodic* if it is invariant under two non-parallel translations. These translation can be chosen horizontal and vertical, which means that a periodic tiling consists of horizontal and vertical repetitions of a tiled rectangle. A tile

set T that admits valid tilings is called *aperiodic* if it does not admit a valid periodic tiling. Already Wang observed [70] that if no aperiodic tile sets existed then the tiling problem would be decidable. One could namely try to tile larger and larger rectangles until one of the following two things happens: a rectangle is found that cannot be tiled, or a period of a periodic tiling is found. In the first case the tile set does not admit a tiling, in the second case it does. Only aperiodic tile sets fail to halt. So, as a corollary to Theorem 4, we have

Corollary 2. *Aperiodic tile sets exist.*

In fact, Berger's proof of Theorem 4 contains a construction of one aperiodic tile set. Currently, the smallest aperiodic set of Wang tiles contains 13 tiles [16,44], and it is an open problem whether one of the tiles in this set is in fact superfluous.

When space–time diagrams of one-dimensional, radius-$\frac{1}{2}$ CA are described using Wang tiles it turns out to be natural to consider tile sets in which the colors of two edges uniquely determine each tile. We call tile set T *NW-deterministic* if for every $s, t \in T$ we have

$$\left.\begin{array}{l} s_{\mathrm{N}} = t_{\mathrm{N}} \\ s_{\mathrm{W}} = t_{\mathrm{W}} \end{array}\right\} \Longrightarrow s = t.$$

This means that in a valid tiling each tile is uniquely determined by its neighbors to the north and to the west. The idea of NW-tile sets is that any valid tiling can be viewed as a space–time diagram of a CA with state set T, where the configurations are read along the infinite SW/NE-diagonals of the tiling.

The following observations were made in [40]:

Theorem 5 (*Kari [40]*). *The tiling problem is undecidable when restricted to NW-deterministic tile sets. There are NW-deterministic, aperiodic tile sets.*

An other undecidable variant of the tiling problem is the *finite tiling problem*, in which we are given a Wang tile set that contains a particular blank tile B. The blank tile has all its edges colored identically. Blank tiling of the plane is the trivial tiling where all tiles are blank. A finite tiling is a tiling that contains only a finite number of non-blank tiles. The finite tiling question asks whether a given tile set with the blank tile admits valid finite tilings other than the trivial blank tiling, and this problem is seen undecidable through a simple reduction from the halting problem of Turing machines [41]. Note that the undecidability of the finite tiling problem is much easier to establish than Theorem 4. Note also a fundamental difference between the two tiling problems: In the finite tiling problem there is a semi-algorithm to detect if a non-trivial finite tiling exists, while in the general tiling problem it is semi-decidable if a tiling does not exist.

Finally, we discuss one particular tile set called SNAKES from [41]. This tile set is aperiodic. The tiles have also an arrow printed on them. The arrow is horizontal or vertical and it points to one of the four neighbors of the tile. Given any tiling, valid or invalid, the arrows determine paths, obtained by following the arrows printed on the tiles. The tile that follows tile $t(x, y)$ on a path is the neighbor of $t(x, y)$ in the direction indicated by the arrow on $t(x, y)$. Note that a path may enter a loop, or it may visit new tiles indefinitely.

The tile set SNAKES has the following plane filling property: consider a tiling t and a path P that indefinitely follows the arrows as discussed above. If the tiling is valid at all tiles that P visits, then the path covers arbitrarily large squares. In other words, for every $N \geqslant 1$ there is a square of $N \times N$ tiles on the plane, all of whose tiles are visited by path P. Note that the tiling may be invalid outside path P, yet the path is forced to snake through larger and larger squares. In fact, SNAKES forces the paths to follow the well-known Hilbert-curve.

3. Garden of Eden

One of the earliest discovered properties of CA were the Garden-of-Eden theorems by Moore and Myhill in 1962 and 1963, respectively. These results relate injectivity and surjectivity of CA with each other. A CA is called *injective* if the global transition function G is one-to-one. It is *surjective* if G is onto. A CA is *bijective* if G is a both onto and one-to-one. We have seen in Corollary 1 that bijectivity is equivalent to reversibility.

If G is not surjective then there exist *Garden-of-Eden* configurations, that is, configurations without a pre-image. A trivial property of finite sets is that a function from a set into itself is injective if and only if it is surjective. In CA the same is true only in one-direction: an injective CA is always surjective, but the converse is not true. However, finite configurations behave more like finite sets: G_{F} is injective if and only if G is surjective. This result, the two directions of which are due to Moore [53] and Myhill [55], is one of the oldest results in the theory of CA:

Theorem 6 (*Garden-of-Eden theorem, Moore [53] and Myhill [55]*). *G_{F} is injective if and only if G is surjective*.

It is trivial that the injectivity of the full function G implies the injectivity of its restrictions G_{F} and G_{P}, so we immediately get the following corollary:

Corollary 3. *Injective CA are also surjective. Hence injectivity, bijectivity and reversibility are equivalent*.

It is also easy to see that the surjectivity of G_{F} or G_{P} implies the surjectivity of G. This is a direct consequence of the compactness of the configuration space $\mathcal{C}$. The next theorem summarizes these and other known relations. The proofs are straightforward and can be found, for example, in [22]. The results are summarized in Figs. 6 and 7.

Theorem 7. *The following implications are true in every dimension d*:
- *If G is injective then G_{P} and G_{F} are injective.*
- *If G_{P} or G_{F} is surjective then G is surjective.*
- *If G_{P} is injective then G_{P} is surjective.*
- *If G is injective then G_{F} is surjective.*

In addition, the following implications are true for one-dimensional CA:
- *If G_{P} is injective then G is injective.*
- *If G is surjective then G_{P} is surjective.*

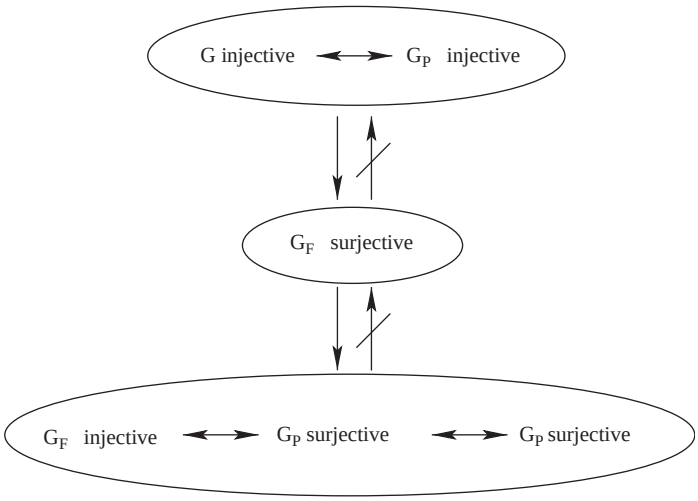


Fig. 6. Implications between injectivity and surjectivity properties in one-dimensional CA.

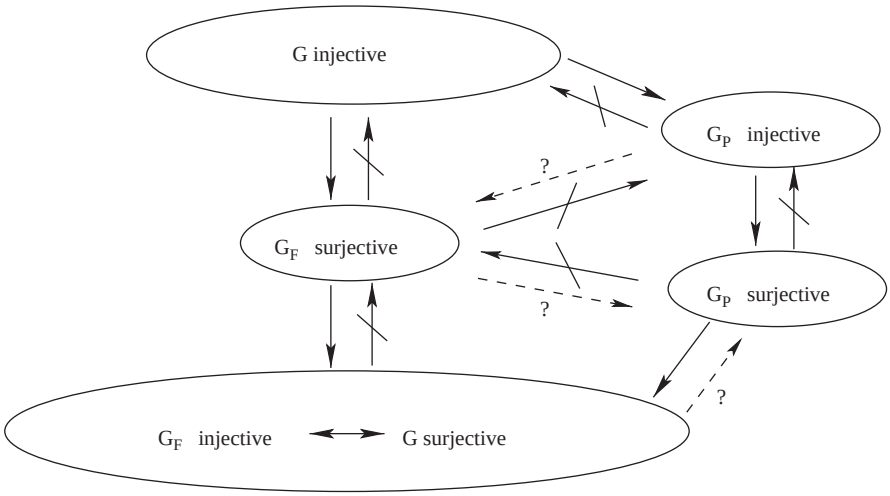


Fig. 7. Implications between injectivity and surjectivity properties in two- and higher-dimensional CA.

Finally, to establish that some implications are not true we use three CA: one-dimensional automata XOR and CONTROLLED-XOR, and two-dimensional SNAKE-XOR.

XOR is a one-dimensional radius-$\frac{1}{2}$ CA with state set $\{0, 1\}$ and the local rule

$$f(x, y) = x + y \pmod 2.$$

State 0 is the quiescent state. XOR is easily seen injective on finite configurations. It is not surjective on finite configurations: for example a configuration with a single state 1 has two infinite predecessors but no finite predecessors.

CONTROLLED-XOR is also a one-dimensional radius-$\frac{1}{2}$ CA. It has four states 00, 01, 10 and 11. The first bit of each state is a control symbol that does not change. If the control symbol of a cell is 0 then the cell is inactive and does not change its state. If the control symbol is 1 then the cell is active and applies the XOR rule on the second bit. In other words,

$$f(ab, cd) = \begin{cases} ab & \text{if } a = 0, \\ a(b + d \pmod 2) & \text{if } a = 1. \end{cases}$$

State 00 is the quiescent state. CONTROLLED-XOR is surjective on finite configurations. It is not injective on unrestricted configurations as two configurations, all of whose cells are active, have the same image if their second bits are complements of each other.

XOR and CONTROLLED-XOR prove the two non-implications in Fig. 6. In higher-dimensional spaces the rules are applied in one of the dimensions only. Then XOR and CONTROLLED-XOR prove five of the six non-implications in Fig. 7. For the remaining

$$G_{\mathrm{P}} \quad \text{injective} \implies G \text{ injective}$$

we need SNAKE-XOR, a two-dimensional CA that uses the SNAKES tile set described in Section 2.8.

SNAKE-XOR is similar to CONTROLLED-XOR. The states consist of two layers: a control layer and a xor layer. The control layer does not change: it only indicates which cells are active and which neighbor cell provides the bit to the XOR operation. In SNAKE-XOR the control layer consist of SNAKES-tiles. Only cells where the tiling on the control layer is valid are active. Active cells execute the modulo two addition on their xor layer. The arrow of the tile tells which neighbor provides the second bit to the XOR operation.

SNAKE-XOR is not injective: two configurations c_0 and c_1 whose control layer consist of the same valid tiling have the same image if their xor layers are complementary to each other. However, SNAKE-XOR is injective on periodic configurations, as the plane filling property ensures that on periodic configurations any infinite path that follows the arrows must contain non-active cells.

Notice that Fig. 7 contains three implications whose status is unknown:

Open problem 2. *In two- and higher-dimensional cellular spaces,*
- *does the injectivity of G_{P} imply the surjectivity of G_{F}?*
- *does the surjectivity of G_{F} imply the surjectivity of G_{P}?*
- *does the surjectivity of G imply the surjectivity of G_{P}?*

4. Reversibility

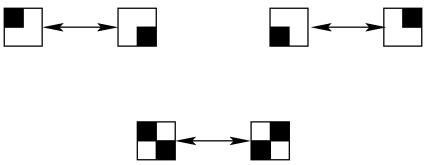
Reversibility is a fundamental property of microscopic physical systems, implied by the laws of quantum mechanics. CA simulating such systems should obey the same laws, hence be reversible. Moreover, a massively parallel computer that optimally uses physics to compute must itself be reversible. Non-reversibility always implies energy dissipation, in practice in the form of heat. It is therefore not surprising that reversible CA have received particular attention since the early days of CA investigation.

Hedlund [33] and Richardson [57] independently proved that all one-to-one CA are reversible (Theorem 1). At that point of time it was not know whether interesting reversible CA exist. But soon T. Toffoli demonstrated that any d-dimensional CA can be simulated by a $d+1$-dimensional reversible CA [65]. Since, it is easy to simulate any Turing machine on a one-dimensional CA, Toffoli's result implies the existence of computationally universal, two-dimensional reversible CA. The result was improved in [54] where it was shown that reversible Turing machines can be simulated by one-dimensional reversible CA. This establishes the existence of universal one-dimensional reversible CA, since reversible Turing machines can be computationally universal [4].

Theorem 8 (*Morita and Harao [54]*). *One-dimensional reversible CA exist that are computationally universal.*

An especially elegant two-dimensional solution is the billiard-ball computer by Margolus [51]. He also introduced the technique of space partitioning as a way to ensure reversibility. In this technique the update is done in two steps (see Fig. 8): in the first step the plane is partitioned into 2×2 blocks along, say, odd coordinates. A permutation π_1 of S^4 is applied inside each block, where S is the state set. In the second step the partitioning is shifted so that the plane is partitioned along even coordinates, and another permutation π_2 of S^4 is applied. This technique became known as the Margolus neighborhood.

In the billiard ball computer we have $\pi_1 = \pi_2$. The state set is binary, and we denote the states as white and black. The permutations π_1 and π_2 only make the following exchanges:



All other 2×2 blocks are unchanged. Because of the alternating partitioning, a single black state propagates on the plane in one of the four diagonal directions that depends on the parity of its location. With such a simple update rule it is possible to simulate the motion and collisions of billiard balls of positive diameter. Walls from which the balls bounce can also be created, and all these constructs can be combined to perform arbitrary computation [51].

A CA that uses the Margolus neighborhood is trivially reversible—the inverse automaton applies the inverse permutations. Also other physical constraints can be easily programmed into a CA that uses such a neighborhood. For example, the number of black cells in finite configurations is automatically preserved if the permutations are such that they conserve black states.

Strictly speaking Margolus neighborhood is not a CA neighborhood in the sense of our definitions. The local update rule is different for even and odd cells. But if we consider "supercells" that consist of 2×2 blocks then all cells are identical. Also time steps become identical if we combine the even and odd clock cycles into one update step. In this strict sense the billiard ball computer by Margolus has $2^4 = 16$ states.

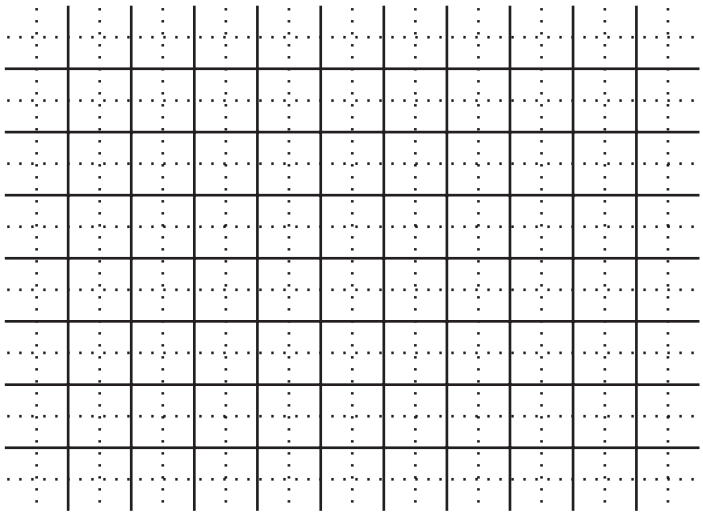


Fig. 8. The Margolus neighborhood. Odd updates use the solid partitioning, even updates the dashed partitioning.

We have seen how space partitioning can be used to guarantee reversibility. A natural question that arises is what CA are reversible when other neighborhoods are used. If each cell has n neighbors then each cell receives as input n elements of S and its output is a single element of S. If $n > 1$ then individual cells erase information. On the other hand, the new state of each cell is transmitted to n cells. By a clever choice of the local update rule the cells can cooperate in a way that preserves information.

It was determined already in 1972 by S. Amoroso and Y. Patt that it is possible to decide if a given one-dimensional CA is reversible [2]. In the same paper, they also provided an algorithm to determine if a given CA is surjective:

Theorem 9 (*Amoroso and Patt [2]*). *There exist algorithms to determine if a given one-dimensional CA is injective or surjective.*

Elegant decision algorithms based on de Bruijn graphs were later designed by Sutner [62]. In higher-dimensional spaces the questions are, however much harder. It was shown in [38,41] that

Theorem 10 (*Kari [41]*). *There are no algorithms to determine if a given two-dimensional CA is injective or surjective.*

The proof for injectivity is a reduction from the tiling problem using the tile set SNAKES from Section 2.8. For a given set T of Wang tiles we construct a two-dimensional CA, similar to SNAKE-XOR of Section 3. The CA has a control layer and a xor layer. The control layers in turn consists of two layers: one with tiles T and one with tiles SNAKES. A cell is active if and only if the tiling is valid at the cell on both tile components. Active cells execute the modulo two addition on their xor layer, and the arrow on the SNAKES tile tells which neighbor provides the second bit to the sum. The plane filling property of SNAKES guarantees

that if two different configurations have the same successor then arbitrarily large squares must have a valid tiling. Conversely, if a valid tiling exists then two different configurations with identical control layers can have the same successor. Hence, the CA we constructed is injective if and only if T does not admit a valid tiling, and this completes the proof.

The proof concerning the surjectivity is an analogous reduction from the finite tiling problem introduced in Section 2.8. The analogy is based on the fact that surjectivity is equivalent to injectivity on finite configurations (Theorem 6). But note the following fundamental difference in the injectivity problems of G and G_{F}: a semi-algorithm exists for the injectivity of G (based on an exhaustive search for the inverse CA) and for the non-injectivity of G_{F} (based on looking for two finite configurations with the same image).

Even though Theorem 1 guarantees that the inverse function of every injective CA is a CA, Theorem 10 implies that the neighborhood of the inverse CA can be very large: there can be no computable upper bound, as otherwise we could test all candidate inverses one-by-one. In contrast, in the one-dimensional space the inverse automaton can only have a relatively small neighborhood. In one-dimensional radius-$\frac{1}{2}$ CA the inverse neighborhood consists of most $s-1$ consecutive cells where s is the number of states [20], and this bound is tight [39].

In view of our motivation in physics and computation—namely implementing reversible massively parallel computers using some reversible physical systems—reversibility of the global transition function G does not seem like the most relevant property to study. Even though $G : \mathcal{C} \longrightarrow \mathcal{C}$ is one-to-one on infinite configurations, the local rule f that produces G is not one-to-one. Function f is then useless if we want to implement G in reversible logic. Rather, we prefer local rules that, unlike f, are themselves reversible, such as the permutations in the Margolus neighborhood. In [67], the natural question was asked whether all reversible CA can be implemented using reversible local update rules. This question was answered affirmatively in [43] for one- and two-dimensional CA.

The Margolus neighborhood can be generalized in a natural way to use partitions into larger blocks, clock cycles longer than two, and to any dimension d. Different phases of the clock cycle may use different partitions. But at each step some permutation is applied on the blocks. We say that such CA use the *generalized Margolus neighborhoods*, and call them GMN-CA. In the literature, GMN-CA are also known as lattice gases.

Theorem 11 (*Kari [43,45]*). *All one- and two-dimensional reversible CA are a composition of a GMN-CA and a "translation-type" CA. Every d-dimensional GMN-CA can be modified into a GMN-CA whose clock cycle has length at most* $d+1$.

Note that the translation-type component is needed in the previous theorem. For example, left shift σ cannot be implemented as a GMN-CA alone [43].

In higher dimensions, we only know that it is possible to add new states to any reversible CA so that the extended reversible CA uses the generalized Margolus neighborhood [24]. The strict variant is open:

Open problem 3. *Is every reversible three- and higher-dimensional CA a composition of a GMN-CA and a "translation-type" CA? Are there examples of d-dimensional GMN-CA that cannot be implemented using a clock cycle shorter than* $d+1$?

5. Conservation laws

Reversible CA preserve information. There are also other conservation laws in physics that CA may obey, e.g. conservation of energy, momentum, etc. The Margolus neighborhood is a particularly useful tool in programming conservation laws into CA.

In [32], Hattori and Takesue introduced and investigated *additive conserved quantities* in CA. These are invariants of the CA evolution that are obtained as sums of locally computed numerical values. The same definitions and results were discovered independently by other researchers, e.g. [9], and a special case called number conservation was later introduced in [8].

A *range-one additive quantity* is a function $\varphi : S \longrightarrow \mathbb{R}$ that assigns a real number to each state. We require that $\varphi(q) = 0$ if q is the quiescent state. Function φ can then be extended into a function $\hat{\varphi}$ that assigns a real number to each finite configuration, obtained by summing up φ over all cells:

$$\hat{\varphi}(c) = \sum_{\vec{x}\in\mathbb{Z}^d} \varphi(c(\vec{x})).$$

The sum is finite for all $c \in \mathcal{C}_F$. We say that CA G *conserves* φ if $\hat{\varphi}(G(c)) = \hat{\varphi}(c)$ for all $c \in \mathcal{C}_F$.

More generally, a d-dimensional *additive quantity* consists of a neighborhood vector $N = (\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n)$ of n distinct elements of $\mathbb{Z}^d$, and a *density function* $\varphi : S^n \longrightarrow \mathbb{R}$. Note that the neighborhood vector is not the same as the neighborhood vector of the CA under consideration. It is required that $\varphi(q, q, \ldots, q) = 0$ if q is the quiescent state. The density function is applied at all cells to obtain the density $\Phi_c : \mathbb{Z}^d \longrightarrow \mathbb{R}$ of configuration c, that is,

$$\Phi_c(\vec{x}) = \varphi(c(\vec{x} + \vec{x}_1), c(\vec{x} + \vec{x}_2), \ldots, c(\vec{x} + \vec{x}_n))$$

for all $\vec{x} \in \mathbb{Z}^d$. The sum

$$\hat{\varphi}(c) = \sum_{\vec{x}\in\mathbb{Z}^d} \Phi_c(\vec{x})$$

of the densities at all cells is finite when c is a finite configuration. We say that CA G conserves the additive quantity if $\hat{\varphi}(G(c)) = \hat{\varphi}(c)$ for all $c \in \mathcal{C}_F$.

The notion defined above can be generalized further in various ways. For example, one can replace $\mathbb{R}$ with any abelian group. Also, conserved quantities of G^t can be considered, where $t \geqslant 1$ is arbitrary. More general staggered invariants were studied in [63]. Note also that the conservation of φ can be equivalently defined in terms of periodic configurations instead of finite configurations. Then the sum of the densities should be taken over one period of the configuration. Definitions using finite and periodic configurations are equivalent.

There is an algorithm to test whether a given additive quantity is conserved by a given CA G. Indeed, it is enough to test for all configurations whether changing the state of one cell has an equal effect before and after applying G, that is, we must have $\hat{\varphi}(c') - \hat{\varphi}(c) = \hat{\varphi}(G(c')) - \hat{\varphi}(G(c))$ where c' is obtained from c by changing one cell $\vec{x}$. This is sufficient because any two finite configurations can be changed into each other by a sequence of single cell changes. Because c' and c, as well as $G(c')$ and $G(c)$, are equal on all cells

except a finite neighborhood of $\vec{x}$, it is enough to perform the test on a finite number of configurations. Hence the test is effective. For practical purposes, the number of tests can be reduced further.

An interesting question is to determine all conserved quantities of a given CA. This question is important in designing CA models for physical systems as incorrect conserved quantities in the model mean that the model obeys incorrect conservation laws that may affect the simulation results. For a given CA G and a fixed neighborhood vector N the local density functions $\varphi : S^n \longrightarrow \mathbb{R}$ that G conserves form a vector space. Based on the condition derived in the previous paragraph, one easily finds the orthogonal complement of the vector space, and therefore all conserved quantities of G with neighborhood N can be effectively found.

Theorem 12 (*Hattori and Takesue [32]*). *There is an algorithm to determine if a given additive quantity is conserved by a given CA. For a given CA and neighborhood vector N one can effectively find all conserved quantities of G that use neighborhood N.*

Notice that there exist some additive quantities that every CA conserves: for example, if $d = 1$, $N = (0, 1)$, and $S = \{0, 1\}$ then the local density function

$$\varphi(01) = 1, \varphi(10) = -1, \varphi(00) = \varphi(11) = 0$$

assigns value 0 to all finite configurations and is therefore conserved by every CA. We say that an additive quantity is trivial if every finite configuration gets value zero.

An interesting and important problem is to find all non-trivial conserved quantities of a given CA, when the neighborhood vector N can be arbitrary.

Open problem 4. *Is there an algorithm to determine if a given one-dimensional CA has any non-trivial conserved quantities*? *Is there an effective method to find all conserved quantities of a given one-dimensional CA*?

Note that for two-dimensional CA the question is easily seen undecidable, using a simple reduction from the finite tiling problem, introduced in Section 2.8: for any tile set T with the blank tile B construct the CA that keeps the state of a cell unchanged if the tiling is valid at the cell and alters the state to B in all other cases. It is easy to see that a non-trivial conserved quantity exists if and only if a non-trivial valid finite tiling exists.

6. Intrinsic universality

What makes CA so interesting is the complexity of the dynamics when the CA rule is iterated. This complexity is apparent already in elementary rules such as rule 110. It is not difficult to simulate an arbitrary Turing machine by a CA, so universal computations are clearly possible in CA. Even very simple rules such as GOL, rule 110 and the billiard ball computer are computationally universal. It seems that computational universality is a very common property in CA, a rule rather than an exception [76].

The universality of GOL and rule 110 is based on performing Turing machine simulations in the CA. But also a stronger form of universality exists that is inherent to CA. Intrinsically universal CA can simulate any other CA, including its evolutions on infinite configurations. We say that CA A simulates CA B if, after a suitable coloring of blocks of states, all space–time diagrams of B are among the space–time diagrams of A, modulo a shift. See [56] for a more detailed definition. CA A is called intrinsically universal if it can simulate all CA of the same dimension. This means that any computation by any CA is among the evolutions of the intrinsically universal CA, after a suitable blocking, shifting and coloring of the states.

A simple intrinsically universal CA was presented already in 1987 by Albert and Culik [1]. Currently, the smallest known one-dimensional intrinsically universal CA is due to Ollinger [56]:

Theorem 13 (*Ollinger [56]*). *There exists an intrinsically universal one-dimensional CA that uses* 6 *states and the nearest-neighbor neighborhood* $N = (-1, 0, 1)$.

It is an open problem whether smaller ones exist.

Open problem 5. *Is rule* 110 *intrinsically universal*?

7. Limit sets

In CA dynamics there are configurations that are transient in the sense that they can only exist early in the evolution. For example, Garden of Eden configurations cannot appear after the first update. The concept of a limit set captures the configurations that are important in the long run, that is, configurations that are not transient.

The limit set $\Lambda = \Lambda[G]$ of CA G consists of all the configurations that can occur after arbitrarily many computation steps. In other words, it consists of those configurations that are not Garden-of-Eden configurations for any G^n. Define $\Lambda^{(n)} = G^n(\mathcal{C})$ for every $n \geqslant 1$.Then the limit set of G is

$$\Lambda = \bigcap_{n=1}^{\infty} \Lambda^{(n)}.$$

The finite time sets form a decreasing chain

$$\Lambda^{(1)} \supseteq \Lambda^{(2)} \supseteq \Lambda^{(3)} \supseteq \cdots$$

Each $\Lambda^{(n)}$ is compact as an image of the compact set $\mathcal{C}$ under continuous mapping G^n. Consequently, the limit set is compact as an intersection of compact sets.

The limit set can never be empty: it must contain at least one homogeneous configuration. This follows from the fact that every CA has temporally periodic homogeneous configurations. In fact, the limit set is either a singleton set (containing only the quiescent configuration) or it is infinite and contains some non-periodic configurations. It was shown in [19] that if the limit set is a singleton set then $\Lambda = \Lambda^{(n)}$ for some n, that is, all configurations become quiescent in at most n steps, which means that the CA is nilpotent.

Every configuration $c \in \Lambda$ has a pre-image in Λ. This fact has an easy topological proof: $G^{-1}(c)$ is topologically closed and therefore also $G^{-1}(c) \cap \Lambda^{(n)}$ is closed for every $n \geqslant 1$. Since $c \in \Lambda$, every $G^{-1}(c) \cap \Lambda^{(n)}$ is non-empty. The sets $G^{-1}(c) \cap \Lambda^{(n)}$, $n = 1, 2, 3, \ldots$ form a decreasing chain of non-empty compact sets, so their intersection is non-empty. But the intersection is $G^{-1}(c) \cap \Lambda$, which means that c has a pre-image in the limit set. We have proved that $G(\Lambda) = \Lambda$, and clearly Λ is the largest set with this property. Note also that for every configuration c in the limit set there exists an infinite pre-image sequence $c_0, c_1, \ldots$ such that $c = c_0$ and $G(c_{i+1}) = c_i$ for every $i \geqslant 0$.

The following theorem summarizes the observations made so far.

Theorem 14 (*Culik et al. [20] and Hurd [34]*). *Limit set Λ is compact and non-empty. It is the largest invariant set, that is, the largest set such that $G(\Lambda) = \Lambda$. The limit set is finite if and only if the CA is nilpotent, in which case Λ is a singleton set and $\Lambda = \Lambda^{(n)}$ for some n.*

It is a natural question to ask what kind of local rules make CA nilpotent. It turns out that no easy characterization exist:

Theorem 15 (*Culik et al. [19] and Kari [40]*). *For every $d \geqslant 1$, it is undecidable whether a given d-dimensional CA is nilpotent.*

In two-dimensional case this can be seen as follows [19]: for any given finite set T of Wang tiles we construct a two-dimensional CA whose state set is $T \cup \{q\}$ and the local update rule keeps the state of a cell unchanged if the tiling is correct at the cell, otherwise the state becomes q. This CA is nilpotent if and only if T does not admit a valid tiling of the plane. The undecidability of the nilpotency problem now follows from the undecidability of the tiling problem (Theorem 4).

To show the undecidability in the one-dimensional case [40] we use NW-deterministic tiles defined in Section 2.8. For any given NW-deterministic tile set T we construct a one-dimensional CA with state set $T \cup \{q\}$. The neighborhood of the CA is $(0, 1)$ and the local update rule f is defined so that $f(a, b) = c$ if $a, b, c \in T$ and c is a tile that match in color when a and b are placed on its western and northern side, respectively. Since T is NW-deterministic there is at most one matching c for every a and b. In all other cases $f(a, b) = q$. A valid tiling by T is then a valid space–time diagram where the SW/NE diagonals are the configurations. So if a valid tiling exists then the CA is not nilpotent. Conversely, if no valid tiling exist then every starting configuration will produce state q at bounded intervals, and these q's spread to cover the entire line in a finite number of steps. We see that the nilpotency problem must be undecidable as otherwise we could solve the NW-deterministic tiling problem, contradicting Theorem 5.

Using Theorem 15 one can show that the topological entropy of a given CA is uncomputable [35]. In [19] several properties of the limit sets were proved undecidable. Soon it was discovered that in fact all non-trivial properties of limit sets are undecidable [42]. A property of limit sets simply means any family of CA such that any two CA that have the same limit set (regardless of their state set) either both are in the family or not in the family. The property is non-trivial if the family is not empty but does not contain all CA either. In

[42], the nilpotency problem was successfully reduced to all non-trivial properties of limit sets, proving that there is no algorithm do determine if the limit set a given CA has the property or not.

Theorem 16 (*Rice's theorem for limit sets, Kari [42]*). *For every $d \geqslant 1$, all non-trivial properties of d-dimensional limit sets are undecidable.*

Note that in the previous theorem the state set of the input CA can be arbitrary. It is an interesting open question to determine what happens when the state set S is fixed. Then surjectivity becomes a property of the limit set: a CA is surjective if and only if its limit set contains all configurations over the state set S. Since surjectivity is decidable in dimension one (Theorem 9), there is at least one decidable property of limit sets. Other such properties are not known:

Open problem 6. *Is the surjectivity question of one-dimensional CA the only decidable property of limit sets when the input is restricted to CA over a fixed state set S.*

8. Dynamical systems approach

CA G is a continuous function on the compact metric space $S^{\mathbb{Z}^d}$. It is then an example of a dynamical system studied by topological dynamics and chaos theory. Important notions in chaos are related to the behavior under the iteration of G of points that are close to each other. We start by defining a few of these concepts.

Configuration c is an *equicontinuity* point of G if for every $\varepsilon > 0$ there exists $\delta > 0$ such that

$$\forall e \in \mathcal{C} : d(c, e) < \delta \Longrightarrow d(G^t(c), G^t(e)) < \varepsilon \quad \text{for all } t \geqslant 0.$$

In other words, configurations sufficiently close to c remain as close to the orbit of c as we want. Let $Eq(G)$ denote the set of equicontinuity points of G. If all configurations are equicontinuity points then the CA is called *equicontinuous*. It was proved in [48] that G is equicontinuous if and only if $G^n = G^m$ for some $n \neq m$.

CA G is called *sensitive to initial conditions*, or simply sensitive, if there are no equicontinuity points, and moreover, the number $\varepsilon > 0$ that contradicts the equicontinuity of c can be chosen uniformly, independent of c. In other words, G is sensitive iff there exists $\varepsilon > 0$ such that

$$(\forall c \in \mathcal{C})(\forall \delta > 0)(\exists e \in \mathcal{C} \text{ and } t \geqslant 0) : d(c, e) < \delta \quad \text{and } d(G^t(c), G^t(e)) > \varepsilon.$$

CA have the property that if there are no equicontinuity points then the CA is sensitive [48].

A stronger form of sensitivity is *positive expansivity*. A CA is called positively expansive if there exists $\varepsilon > 0$ such that for any two different configurations c and e we have $d(G^t(c), G^t(e)) > \varepsilon$ for some $t \geqslant 0$. In other words, no matter how close the configuration c and e are to each other, their orbits will eventually be separated by distance ε.

In [48], Kurka proposed the following equicontinuity classification of CA:

(K1) equicontinuous CA, that is, $Eq(G) = \mathcal{C}$,

(K2) CA with some equicontinuity points, that is, $\emptyset \subsetneq Eq(G) \subsetneq \mathcal{C}$,
(K3) sensitive but not positively expansive CA,
(K4) positively expansive CA

Classes (K3) and (K4) contain all CA that do not have any equicontinuity points. Hence every CA belongs to exactly one class. In two- and higher-dimensional cases class (K4) is empty: there are no positively expansive two-dimensional CA [26,60].

The decision problem to determine if a given CA belongs to a given class was studied in [23]. It was proved that even for one-dimensional CA it is undecidable if a given CA belongs to class (K1), (K2) or (K3). The membership problem for class (K4) remains an open problem.

Open problem 7. *Is it decidable whether a given one-dimensional CA is positively expansive*?

Let us define two more concepts that are related to chaos. CA is called *transitive* if for any two open sets U and V there exists $t \geqslant 0$ such that $G^t(U) \cap V \neq \emptyset$, and it is *mixing* if for every open U and V there exists t_0 such that $G^t(U) \cap V \neq \emptyset$ for all $t \geqslant t_0$. Transitivity can be equivalently defined in terms of orbits: a CA is transitive if and only if some configuration c has a dense orbit. The two definitions of transitivity are seen equivalent using the Baire category theorem.

Trivially all mixing CA are also transitive. More interesting implications are

$$G \text{ positively expansive} \Longrightarrow G \text{ mixing}$$

by Blanchard and Maass [7] and

$$G \text{ transitive} \Longrightarrow G \text{ surjective and sensitive to initial conditions}$$

by Kurka [48]. Now we are ready to define chaos in CA. One popular definition of chaotic dynamical systems is due to Devaney [21]: a dynamical system is called *chaotic* if
(1) it is transitive,
(2) temporally periodic points are dense, and
(3) it is sensitive to initial conditions.

In CA transitivity implies sensitivity so only conditions (1) and (2) remain. A challenging open problem concerns the second condition:

Open problem 8. *Are the temporally periodic configurations dense when G is surjective*?

If the answer is affirmative then chaos in CA becomes equivalent to transitivity.

9. Linear local rules

If the local rule of a CA is a linear function, where the state set S is a commutative finite ring with identity, then the CA behaves particularly nicely. Such CA are called *linear* or *additive*. One should note that in some CA literature all one-dimensional CA are called

linear, referring to the organization of the cells on the line. In our vocabulary linear is used to refer to the linearity of the update rule. More precisely, the local rule must take the form

$$f(a_1, a_2, \ldots, a_n) = c_1a_1 + c_2a_2 + \cdots + c_na_n$$

for some constants $c_1, c_2, \ldots, c_n \in S$. We denote the identity of the ring by $1 \in S$. An especially important case is $S = \mathbb{Z}_m$ where $\mathbb{Z}_m$ is the set of integers modulo m. Linearity simplifies analysis of CA, and some problems that are undecidable for general CA have polynomial time algorithms on linear rules.

The set $\mathcal{C}$ of configurations forms an S-module, in which any two configurations are added and multiplied by elements of S cell wise, that is, for any configurations $c_1, c_2 \in \mathcal{C}$ and elements $a, b \in S$ the configuration $e = ac_1 + bc_2$ is given by $e(\vec{x}) = ac_1(\vec{x}) + bc_2(\vec{x})$ for all $\vec{x} \in \mathbb{Z}^d$. Linearity implies the *superposition principle*: If c_1 and c_2 are two configurations, then for any $a, b \in S$ we have $G(ac_1 + bc_2) = aG(c_1) + bG(c_2)$.

Actually the superposition principle can be taken as the defining condition of linear CA, and the linearity can then be generalized to arbitrary finite abelian groups S. We say that CA G over finite abelian group S is linear if $G(c_1 + c_2) = G(c_1) + G(c_2)$ for any two configurations $c_1, c_2 \in \mathcal{C}$. However, in the following we always assume the commutative ring structure and the identity element 1.

A useful representation of linear local rules uses Laurent polynomials, that is, "polynomials" with negative and positive powers of the variables. A d-dimensional CA with local rule $f(a_1, a_2, \ldots, a_n) = c_1a_1 + c_2a_2 + \cdots + c_na_n$ and neighborhood $N = (\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n)$ is represented as the Laurent polynomial

$$p(Z) = c_1Z^{-\vec{x}_1} + c_2Z^{-\vec{x}_2} + \cdots + c_nZ^{-\vec{x}_n}$$

with d variables $z_1, z_2, \ldots, z_d$ where we have used the notation

$$Z^{(y_1, y_2, \ldots, y_n)} = z_1^{y_1} z_2^{y_2} \ldots z_d^{y_d}$$

for every $(y_1, y_2, \ldots, y_d) \in \mathbb{Z}^d$. In other words, the coefficients of the Laurent polynomial are the coefficients of the local rule while the exponents of the variables express the neighborhood. The polynomial is Laurent because both negative and positive powers of the variables are possible. Letter Z will be used to refer to the vector $(z_1, z_2, \ldots, z_d)$ of variables.

Notice that if Laurent polynomials $p(Z)$ and $q(Z)$ over ring S define global functions G and H, respectively, then the product $p(Z)q(Z)$ represents the composition $G \circ H$. Consequently, for any $k \geqslant 1$, $p^k(Z)$ represents G^k.

Also, it is useful to represent configurations as Laurent power series: let d-dimensional configuration c correspond to the formal power series

$$s(Z) = \sum_{\vec{x} \in \mathbb{Z}^d} c(\vec{x}) Z^{\vec{x}}.$$

Then the product $p(Z)s(Z)$ is the power series that represents the configuration $G(c)$, and $p^k(Z)s(Z)$ represents $G^k(c)$.

Let us denote by $S[Z, Z^{-1}]$ the set of Laurent polynomials on variables $Z = (z_1, z_2, \ldots, z_d)$ over the coefficient ring S. The set $S[Z, Z^{-1}]$ itself is an (infinite) commutative ring. Let $S[\![Z, Z^{-1}]\!]$ denote the set of Laurent series over S. It is an S-module.

Let us first investigate the problem of determining which linear CA functions are injective and surjective. The inverse of a linear injective function is also a linear function, so $p(Z)$ defines an injective CA if and only if there exists a Laurent polynomial $q(Z)$ such that $p(Z)q(Z) = 1$. This $q(Z)$ is the local rule of the inverse automaton. Such $q(Z)$ exists if and only if $p(Z)$ is a unit of the ring $S[Z, Z^{-1}]$.

A CA is surjective if and only if it is not injective on finite configurations. Because of the superposition principle this is equivalent to saying that no non-zero configuration is mapped to the zero configuration. This means that Laurent polynomial $p(Z)$ defines a non-surjective CA if and only if there exists a Laurent polynomial $q(Z) \neq 0$ such that $p(Z)q(Z) = 0$, in other words, if and only if $p(Z)$ is a zero divisor in the ring $S[Z, Z^{-1}]$.

It turns out that inclusions of the coefficients of $p(Z)$ in the maximal ideals of ring S determines the injectivity and surjectivity status of the CA, as proved in [59]:

Theorem 17 (*Sato [59]*). *The linear CA represented by the Laurent polynomial $p(Z)$ over ring S is*

(a) *surjective if and only if no maximal ideal of S contains all coefficients of $p(Z)$.*
(b) *injective if and only if for every maximal ideal of S exactly one coefficient of $p(Z)$ is outside the ideal.*

We have the following easy to check formulations of the conditions in Theorem 17: the CA defined by $p(Z)$ is

(a) surjective if and only if $a \cdot p(Z) \neq 0$ for every $a \in S \setminus \{0\}$,
(b) injective if and only if for every $a \in S \setminus \{0\}$ there exists $b \in S$ such that $ab \cdot p(Z)$ is a monomial.

In the special case $S = \mathbb{Z}_m$ we obtain the following well-known result [37]:

Corollary 4 (*Ito et al. [37]*). *Let G be a linear CA over ring $\mathbb{Z}_m$, and let*

$$f(a_1, a_2, \ldots, a_n) = c_1a_1 + c_2a_2 + \cdots + c_na_n$$

be its local rule. Then G is

(a) *surjective if and only* $\gcd(m, c_1, c_2, \ldots, c_n) = 1$, *and*
(b) *injective if and only if every prime factor p of m divides all but exactly one coefficient $c_1, c_2, \ldots, c_n$.*

When G is injective, efficient algorithms exist for constructing the inverse CA [49]. See also [46] for a more general case.

Let us next consider dynamical properties defined in Section 8, and let us assume for the rest of this section that $S = \mathbb{Z}_m$. There exist simple conditions for topological properties such as equicontinuity, sensitivity to initial conditions, transitivity and positive expansivity:

Theorem 18 (*Cattaneo et al. [11] and Manzini and Margara [50]*). *Let G be a linear CA over ring $\mathbb{Z}_m$, and let*

$$f(a_1, a_2, \ldots, a_n) = c_1a_1 + c_2a_2 + \cdots + c_na_n$$

be its local rule. Let us assume, without loss of generality, that the first coefficient corresponds to relative neighborhood offset zero, that is, $\vec{x}_1 = \vec{0}$.

- *G is equicontinuous if and only if it is equicontinuous at some point, which is equivalent to the condition: all prime factors of m divide all coefficients* $c_2, c_3, \ldots, c_n$. (*Note that the prime factors do not need to divide the coefficient* c_1 *corresponding to the offset* $\vec{0}$.)
- *G is sensitive if and only if it is not equicontinuous. This is equivalent to: m has a prime factor that does not divide some coefficient* $c_2, c_3, \ldots, c_n$.
- *G is transitive if and only if* $\gcd(m, c_2, c_3, \ldots, c_n) = 1$.
- *One-dimensional G is positively expansive if and only if*

$$\gcd(m, d_1, d_2, \ldots, d_k) = \gcd(m, e_1, e_2, \ldots, e_{n-k-1}) = 1$$

where elements $d_1, d_2, \ldots, d_k$ *and* $e_1, e_2, \ldots, e_{n-k-1}$ *are the coefficients of the local rule that correspond to negative and positive relative neighborhood offsets, respectively. Higher dimensional positively expansive CA do not exist.*

The previous theorem gives a complete picture of the topological properties in linear CA over $\mathbb{Z}_m$. Note that all the given conditions are fast to test because the greatest common divisor is easy to compute. Even the tests for equicontinuity and sensitivity are fast because all we need to do is to compute $g = \gcd(m, c_2, c_3, \ldots, c_n)$ and check whether $g^{\lfloor \log_2 m \rfloor}$ is divisible by m.

10. Language recognition

Language recognition by space-bounded one-dimensional CA is among the classical research topics in CA theory. In this computation model the active part of the CA is bounded by the length of the input word. Some problems posed in the early 1970s remain unsolved even today. Real-time and linear-time recognition is of particular interest.

In this section, CA will be one-dimensional with the nearest-neighbor neighborhood $(-1, 0, 1)$. Such a CA is called one-way (OCA for short) if it is equivalent to a CA that uses the neighborhood $(0, 1)$. The state set is $S \cup \{\#\}$ where $\# \notin S$ is the boundary symbol. The symbol # has the property that it is never destroyed or created, that is, $f(a, b, c) = \#$ if and only if $b = \#$. This guarantees that the active part of the CA cannot grow. Some states are accepting, let $A \subseteq S$ denote the set of accepting states.

Let $\Sigma \subset S$ be an alphabet, and let $w \in \Sigma^*$ be an input word. The corresponding initial configuration is c_w where the state $c_w(i)$ of cell i is the ith letter of w for $i = 1, 2, \ldots, |w|$, and the boundary symbol # for $i < 1$ and $i > |w|$. Word w is accepted by CA G iff there exists time t such that $G^t(c_w)$ has cell 1 in an accepting state. We say that w is accepted at time t. The language $L(G)$ recognized by G consists of all words over Σ that are accepted by G. The family of languages recognized by some CA (or OCA) will be denoted $\mathcal{L}(CA)$ (or $\mathcal{L}(OCA)$, respectively). It is not known whether $\mathcal{L}(CA)$ and $\mathcal{L}(OCA)$ are the same language family. What is known is that $\mathcal{L}(CA)$ is exactly the family of deterministic context-sensitive languages and that $\mathcal{L}(OCA)$ contains all context-free languages [47].

The language recognized by G in time $T(n)$ consists of all words w that are accepted at time $T(|w|)$ where $|w|$ is the length of the word w. If $T(n) = cn$ for some constant c then

the language is accepted in linear time, and if $T(n) = n - 1$ then the language is accepted in real time. Notice that $|w| - 1$ is the earliest time when the leftmost cell may possibly have received information about all letters of the input word w. Let us denote the families of languages accepted by CA in linear and real time by $\mathcal{L}(LCA)$ and $\mathcal{L}(RCA)$, respectively, and the analogous families for one-way CA by $\mathcal{L}(LOCA)$ and $\mathcal{L}(ROCA)$, respectively.

Inclusions

$$\mathcal{L}(ROCA) \subseteq \mathcal{L}(RCA) \subseteq \mathcal{L}(LCA)$$

are trivial. More interesting relations

$$\mathcal{L}(LCA) \subseteq \mathcal{L}(OCA)$$

and

$$\mathcal{L}(RCA) = \mathcal{L}(LOCA)$$

were proved in [36,12], respectively. In [12], it was also shown that language $\{a^{2^n} \mid n \geqslant 2\}$ can be recognized in real time by a CA but not in real time by any OCA. Fig. 9 summarizes the known inclusions.

Note that even the most restricted family $\mathcal{L}(ROCA)$ contains non-context-free languages, e.g. language $\{a^n b^n c^n \mid n \geqslant 1\}$ [25], while it does not contain all context-free languages [64].

Concerning closure properties it is known that the real-time language families $\mathcal{L}(RCA)$ and $\mathcal{L}(ROCA)$ are closed under the boolean operations [61]. It is also known that $\mathcal{L}(ROCA)$ and $\mathcal{L}(LCA)$ are closed under reversal [12,61], while $\mathcal{L}(ROCA)$ is not closed under concatenation [64]. The closure of $\mathcal{L}(RCA)$ under reversal is unknown, but it is known that the closure is true if and only if $\mathcal{L}(RCA) = \mathcal{L}(LCA)$. This is an intriguing open problem, already posed in 1972 [61].

Open problem 9 (*Smith [61]*). *Is* $\mathcal{L}(RCA) = \mathcal{L}(LCA)$? *equivalently*: *is* $\mathcal{L}(RCA)$ *closed under reversal*?

Note that, quite interestingly, it is not even know whether $\mathcal{L}(RCA)$ and $\mathcal{L}(CA)$ are different.

11. Conclusions

We have presented known results and open problems over a variety of research areas in CA theory. It is clear that we have omitted many exciting topics. Examples of omissions include the firing squad synchronization problem [52], results on fault tolerance [28] and quantum CA [71]. We have provided an overview of results on CA that we consider important in the field of computation theory, and several open problems—some of them quite old and difficult—were also presented.

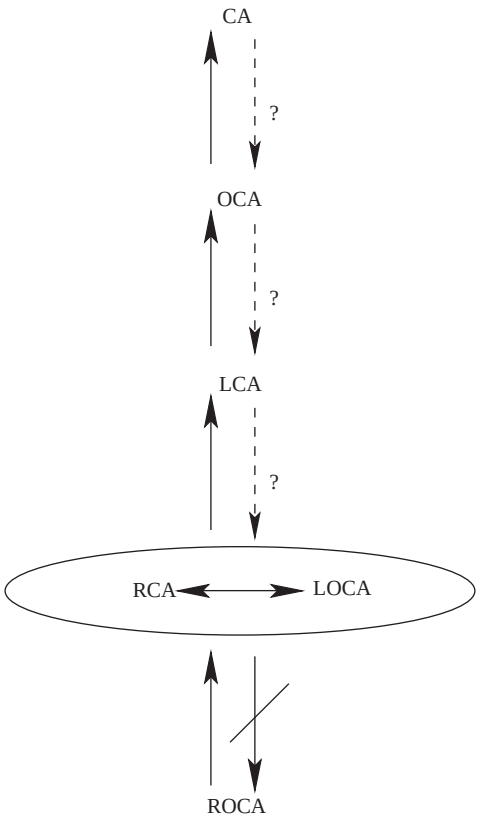


Fig. 9. Inclusions between unrestricted time, linear-time and real-time languages accepted by space-bounded CA and one-way CA. Question marks indicate unknown cases.

References

[1] J. Albert, K. Culik II, A simple universal cellular automaton and its one-way and totalistic version, Complex Systems 1 (1987) 1–16.
[2] S. Amoroso, Y. Patt, Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures, J. Comput. System Sci. 6 (1972) 448–464.
[3] H. Aso, N. Honda, Dynamical characteristics of linear cellular automata, J. Comput. System Sci. 30 (1985) 291–317.
[4] C. Bennett, Logical reversibility of computation, IBM J. Res. Develop. 6 (1973) 525–532.
[5] R. Berger, The undecidability of the Domino problem, Mem. Amer. Math. Soc. 66 (1966).
[6] E.R. Berlekamp, J.H. Conway, R.K. Guy, Winning Ways for Your Mathematical Plays II, Academic Press, New York, 1982.
[7] F. Blanchard, A. Maass, Dynamical properties of expansive one-sided cellular automata, Israel J. Math. 99 (1997) 149–174.
[8] N. Boccara, H. Fuks, Number conserving cellular automaton rules, Fund. Inform. 52 (2002) 1–13.
[9] T. Boykett, C. Moore, Conserved quantities in one-dimensional cellular automata, unpublished manuscript, 1998.

[10] A.W. Burks, Von Neumann's self-reproducing automata, in: A.W. Burks (Ed.), Essays on Cellular Automata, University of Illinois Press, Champaign, IL, 1970, pp. 3–64.
[11] G. Cattaneo, E. Formenti, G. Manzini, L. Margara, Ergodicity, transitivity, and regularity for linear cellular automata over $\mathbb{Z}_m$, Theoret. Comput. Sci. 233 (2000) 147–164.
[12] C. Choffrut, K. Culik II, On real-time cellular automata and trellis automata, Acta Inform. 21 (1984) 393–409.
[13] B. Chopart, M. Droz, Cellular Automata Modeling of Physical Systems, Cambridge University Press, Cambridge, 1998.
[14] E.F. Codd, Cellular Automata, Academic Press, New York, 1968.
[15] J.H. Conway, unpublished, 1970.
[16] K. Culik II, An aperiodic set of 13 Wang tiles, Discrete Math. 160 (1996) 245–251.
[17] K. Culik II, S. Yu, Undecidability of CA classification schemes, Complex Systems 2 (1988) 177–190.
[18] K. Culik II, L.P. Hurd, S. Yu, Formal languages and global cellular automaton behavior, Physica D 45 (1990) 396–403.
[19] K. Culik II, J. Pachl, S. Yu, On the limit sets of cellular automata, SIAM J. Comput. 18 (1989) 831–842.
[20] E. Czeizler, J. Kari, A tight linear bound on the neighborhood of inverse cellular automata, to appear.
[21] R.L. Devaney, An introduction to chaotic dynamical systems, Addison–Wesley, Reading, MA, 1989.
[22] B. Durand, Global properties of 2D cellular automata, in: E. Goles, S. Martinez (Eds.), Cellular Automata and Complex Systems, Kluwer, Dordrecht, 1998, .
[23] B. Durand, E. Formenti, G. Varouchas, On undecidability of equicontinuity classification for cellular automata, in: M. Morvan, E. Remila (Eds.), Discrete Mathematics and Theoretical Computer Science Proceedings AB, 2003, pp. 117–128.
[24] J. Durand-Lose, Representing reversible cellular automata with reversible block cellular automata, in: R. Cori, J. Mazoyer, M. Morvan, R. Mosery (Eds.), Discrete Models Combinatorics Computation and Geometry, Springer, Berlin, 2001, pp. 145–154.
[25] C. Dyer, One-way bounded cellular automata, Inform. Control 44 (1980) 261–281.
[26] M. Finelli, G. Manzini, L. Margara, Luapunov exponents vs expansivity and sensitivity in cellular automata, J. Complexity 14 (1998) 210–233.
[27] U. Frisch, B. Hasslacher, Y. Pomeau, Lattice-gas automata for the Navier–Stokes equation, Phys. Rev. Lett. 56 (1986) 1505–1508.
[28] P. Gacs, Reliable computation with cellular automata, J. Comput. Systems Sci. 32 (1986) 15–78.
[29] M. Gardner, Mathematical games, Sci. Amer. 223–225 (1970–1971).
[30] M. Garzon, Models of Massive Parallelism, Springer, Berlin, 1995.
[31] J. Hardy, Y. Pomeau, O. de Pazzis, Molecular dynamics of a classical lattice gas: transport properties and time correlation functions, Phys. Rev. A 13 (1976) 1949–1961.
[32] T. Hattori, S. Takesue, Additive conserved quantities in discrete-time lattice dynamical systems, Physica D 49 (1991) 295–322.
[33] G. Hedlund, Endomorphisms and automorphisms of shift dynamical systems, Math. Systems Theory 3 (1969) 320–375.
[34] L.P. Hurd, Formal language characterizations of cellular automaton limit sets, Complex Systems 1 (1987) 69–80.
[35] L.P. Hurd, J. Kari, K. Culik, The topological entropy of cellular automata is uncomputable, Ercodic Theory Dynamical Systems 12 (1992) 255–265.
[36] O. Ibarra, I. Jiang, Relating the power of cellular arrays to their closure properties, Theoret. Comput. Sci. 57 (1988) 225–238.
[37] M. Ito, N. Osato, M. Nasu, Linear Cellular Automata over $\mathbb{Z}_m$, J. Comput. System Sci. 27 (1983) 125–140.
[38] J. Kari, Reversibility of 2D cellular automata is undecidable, Physica D 45 (1990) 379–385.
[39] J. Kari, On the inverse neighborhoods of reversible cellular automata, in: G. Rozenberg, A. Salomaa (Eds.), Lindenmayer Systems, Impacts on Theoretical Computer Science, Computer Graphics, and Developmental Biology, Springer, Berlin, 1992, pp. 477–495.
[40] J. Kari, The nilpotency problem of one-dimensional cellular automata, SIAM J. Comput. 21 (1992) 571–586.
[41] J. Kari, Reversibility and surjectivity problems of cellular automata, J. Comput. System Sci. 48 (1994) 149–182.

[42] J. Kari, Rice's theorem for the limit sets of cellular automata, Theoret. Comput. Sci. 127 (1994) 229–254.
[43] J. Kari, Representation of reversible cellular automata with block permutations, Math. Systems Theory 29 (1996) 47–61.
[44] J. Kari, A small aperiodic set of Wang tiles, Discrete Math. 160 (1996) 259–264.
[45] J. Kari, On the circuit depth of structurally reversible cellular automata, Fund. Inform. 38 (1999) 93–107.
[46] J. Kari, Linear cellular automata with multiple state variables, in: Proc. STACS'2000, 17th Annu. Symp. on Theoretical Aspects of Computer Science, Lecture Notes in Computer Science, Vol. 1770, Springer, Berlin, 2000, pp. 110–121.
[47] T. Kasami, M. Fujii, Some results on capabilities of one-dimensional iterative logical networks, Electron. Comm. Japan 51-C (1968) 167–176.
[48] P. Kurka, Languages equicontinuity and attractors in cellular automata, Ergodic Theory Dynamical Systems 17 (1997) 417–433.
[49] G. Manzini, L. Margara, Invertible linear cellular automata over $\mathbb{Z}_m$: algorithmic and dynamical aspects, J. Comput. System Sci. 56 (1998) 60–67.
[50] G. Manzini, L. Margara, A complete and efficiently computable topological classification of D-dimensional linear cellular automata over $\mathbb{Z}_m$, Theoret. Comput. Sci. 221 (1999) 157–177.
[51] N. Margolus, Physics-like models of computation, Physica D 10 (1984) 81–95.
[52] J. Mazoyer, A six states minimal time solution to the firing squad synchronization problem, Theoret. Comput. Sci. 50 (1987) 183–238.
[53] E.F. Moore, Machine models of self-reproduction, Proc. Symp. in Applied Mathematics 14 (1962) 17–33.
[54] K. Morita, M. Harao, Computation Universality of one dimensional reversible injective cellular automata, IEICE Trans. E 72 (1989) 758–762.
[55] J. Myhill, The converse to Moore's Garden-of-Eden theorem, Proc. Amer. Math. Soc. 14 (1963) 685–686.
[56] N. Ollinger, The quest for small universal cellular automata, in: Proc. of ICALP 2002, 29th Internat. Colloq. on Automata, Languages and Programming, Lecture Notes in Computer Science, Vol. 2380, Springer, Berlin, 2002, pp. 318–329.
[57] D. Richardson, Tessellation with local transformations, J. Comput. System Sci. 6 (1972) 373–388.
[58] R.M. Robinson, Undecidability and nonperiodicity for tilings of the plane, Invent. Math. 12 (1971) 177–209.
[59] T. Sato, Decidability of some problems of linear cellular automata over finite commutative rings, Inform. Process. Lett. 46 (1993) 151–155.
[60] M.A. Shereshevsky, Expansiveness, entropy and polynomial growth for groups acting on subshifts by automorphisms, Indag. Math. 4 (1993) 203–210.
[61] A.R. Smith, Real-time language recognition by one-dimensional cellular automata, J. Comput. System Sci. 6 (1972) 233–253.
[62] K. Sutner, De Bruijn graphs and linear cellular automata, Complex Systems 5 (1991) 19–31.
[63] S. Takesue, Staggered invariants in cellular automata, Complex Systems 9 (1995) 149–168.
[64] V. Terrier, On real time one-way cellular array, Theoret. Comput. Sci. 141 (1995) 331–335.
[65] T. Toffoli, Computation and construction universality of reversible cellular automata, J. Comput. System Sci. 15 (1977) 213–231.
[66] T. Toffoli, N. Margolus, Cellular Automata Machines, MIT Press, Cambridge, MA, 1987.
[67] T. Toffoli, N. Margolus, Invertible cellular automata: a review, Physica D 45 (1990) 229–253.
[68] G. Vichniac, Simulating physics with cellular automata, Physica D 10 (1984) 96–115.
[69] J. vonNeumann, in: A.W. Burks (Ed.), Theory of Self-Reproducing Automata, University of Illinois Press, Champign, IL, 1966.
[70] H. Wang, Proving theorems by pattern recognition—II, Bell System Tech. J. 40 (1961) 1–42.
[71] J. Watrous, On one-dimensional quantum cellular automata, in: Proc. of the 36th Annu. Symp. on Foundations of Computer Science, IEEE, New York, 1995, pp. 528–537.
[72] T.A. Witten, L.M. Sander, Diffusion-limited aggregation, Phys. Rev. B 27 (1983) 5686–5697.
[73] S. Wolfram, Statistical mechanics of cellular automata, Rev. Mod. Phys. 55 (1983) 601–644.
[74] S. Wolfram, Universality and complexity in cellular automata, Physica 10 D (1984) 1–35.
[75] S. Wolfram (Ed.), Theory and Applications of Cellular Automata, World Scientific Press, Singapore, 1986.
[76] S. Wolfram, A New Kind of Science, Wolfram Media, 2002.