Statistical Complexity of Boolean Cellular Automata with Short-Term Reaction-Diffusion Memory on a Square Lattice

Zakarya Zarezadeh
Giovanni Costantini

Department of Electronic Engineering
University of Rome, "Tor Vergata"

Memory is a ubiquitous phenomenon in biological systems, in which the present system state is not entirely determined by the current conditions but also depends on the time evolutionary path of the system. Specifically, many phenomena related to memory are characterized by chemical memory reactions that may fire under particular system conditions. These conditional chemical reactions contradict the extant approaches for modeling chemical kinetics and have increasingly posed significant challenges to mathematical modeling and computer simulation. Along these lines, we can imagine a memory module contributing to cell therapy or the synthetic differentiation of certain cells in a certain fashion after experiencing a brief stimulus. We demonstrate that information processing properties of cellular automata (CAs) can be controlled by a signal composed of excitation pulses. We discuss how cellular memory can be incorporated into more complex systems like CAs to understand the controlling of information processing performed by a medium with the use of a pulse signal propagated from a number of control cells. In this paper, we also investigate the potential application of cellular computation for constructing pseudorandom number generators (PRNGs). Furthermore, the PRNG scheme based on CAs with reaction-diffusion memory is proposed for its capability of generating ultrahigh-quality random numbers. However, the quality bottleneck of a practical PRNG lies in the limited cycle of the generator. To close the gap between the pure randomness generation and the short period, we propose and implement a memory algorithm based on a reaction-diffusion process in a chemical system for Boolean CAs. This scheme is characterized by a tradeoff between, on one hand, the rate of generation of random bits and, on the other hand, the degree of randomness that the series can deliver. These successful applications of the memory modeling framework suggest that this innovative theory is an effective and powerful tool for studying memory processes and conditional chemical reactions in a wide range of complex biological systems. This result also opens a new perspective to apply CAs as a computational engine for the robust generation of pure random numbers, which has important applications in cryptography and other related areas.

Keywords: cellular automata; cryptography

1. Introduction

The computational power of cellular automata (CAs) has been emphasized since their origin: the self-reproducing two-dimensional von Neumann automaton is computation universal. Computational universality is obtained by simulating a Turing machine. To design an algorithm is to set up state transitions of a cellular automaton (CA) with a defined behavior on a set of initial configurations. This behavior may be of a computational nature or intrinsically connected with parallelism without any counterpart in any sequential model of computation. In cellular computation, understanding the computational and dynamical properties of biological neural networks is an issue of central importance. In this context, much interest has been focused on comparing the computational power of diverse theoretical models with those of abstract computing devices. Nowadays, the computational capabilities of neural models is known to be tightly related to the nature of the activation function of the neurons, to the nature of their synaptic connections, to the eventual presence of noise in the model, to the possibility for the networks to evolve over time, and to the computational paradigm performed by the networks. The first and seminal results in this direction were provided by McCulloch and Pitts, Kleene and Minsky [1–3], who proved that first-order Boolean recurrent neural networks were computationally equivalent to classical finite-state automata. From a general perspective, it was argued that the classical computational approach from Turing "no longer fully corresponds to the current notion of computing in modern systems" [4], especially when it refers to bio-inspired complex information processing systems [5]. Rather, in the brain (or in organic life in general), information is processed in an interactive way where previous experience must affect the perception of future inputs, and where older memories may themselves change with response to new inputs. In this paper, we pursue the study of the computational power of CAs; more precisely, as a first step, we provide a generalization to obtain the proposed network. As a second step, we show the expressive computational power of the proposed model by an application of it in cryptography and related areas of science. We also show that our automata-theoretical approach might bear new founding elements for the understanding of the complexity of real brain circuits. This paper is organized as follows: in Section 1, we introduced CA as an abstract computational system; in Sections 2, 3 and 4, we will discuss the multiple faces of the current interest in CAs, mathematical, computational, and scientific modeling; and in Sections 5 and 6, we will give a summary of the structure of the proposed model and describe the statistical and dynamical analysis of our model. We pose the problem of random number generation in the framework of CAs, and in

Section 7, we state our results and give a brief overview of future directions of this research.

2. Cellular Automata: Basic Definition

From a formal point of view, a CA is an infinite lattice of finite automata, called cells. The cells are located at the integer lattice points of the d-dimensional Euclidean space. In general, any Abelian group $\mathcal{G}$ in place of $\mathbb{Z}^d$ can be used. In particular, we may consider $\mathbb{Z} \big/ m^d$ a toroidal space, where $\mathbb{Z} / m$ is the additive group of integers modulo m. In $\mathbb{Z}^d$ we identify the cells by their coordinates. This means that the cells are addressed by the elements of $\mathbb{Z}^d$.

Definition 1. Let $\mathcal{S}$ be a finite set of states and $\mathcal{S} \neq \Phi$. A configuration of CAs is a function $c : \mathbb{Z}^d \longmapsto \mathcal{S}$. The set of all configurations is denoted by C.

At discrete time steps, the cells change their states synchronously. Simply, the next state of each cell depends on the current states of the neighboring cells according to an update rule. All the 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.

Definition 2. Let

$$\mathbb{N} = \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix}$$

be a vector of n elements of $\mathbb{Z}^d$. Then the neighbors of α cell at location $x \in \mathbb{Z}^d$ are the n cells at location $x + x_i$ for $i = 1, \ldots, n$. The local transformation rule (transition function) is a function $\mathcal{F} : \mathcal{S}^n \longmapsto \mathcal{S}$ where n is the size of neighborhood. State

$$\mathcal{F}\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \cdot \\ \cdot \\ \cdot \\ \alpha_n \end{pmatrix}$$

is the new state of the cell at time $t+1$ whose n neighbors were at state

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \cdot \\ \cdot \\ \cdot \\ \alpha_n \end{pmatrix}$$

at time t.

Definition 3. The local transition function defines a global function $\Phi : C \longmapsto C$ as follows,

$$\mathcal{G}(c)(x) :\rightarrow \begin{pmatrix} c(x+x_1) \\ c(x+x_2) \\ \cdot \\ \cdot \\ \cdot \\ c(x+x_n) \end{pmatrix}.$$

The CA evolves from a starting configuration $c^0(t=0)$, where the configuration c^{t+1} at time $(t+1)$ is determined by $c^{t+1} :\rightarrow \mathcal{G}(c^t)$. Thus, CAs are dynamical systems that are updated locally and are homogeneous and discrete in time and space. Often in literature, CAs are specified by a quadruple $\mathbb{A} = (d, \mathcal{S}, \mathbb{N}, \mathcal{F})$, where d is a positive integer, $\mathcal{S}$ is the set of states (finite), $\mathbb{N} \in (\mathbb{Z}^d)$ is the neighborhood vector, and $\mathcal{F} : \mathcal{S}^n \longmapsto \mathcal{S}$ is the local transformation rule. The elementary CA is 1(space) + 1(time) dimension, each cell of which takes only two states denoted by the integer values 0 and 1. The value of the cell i at time step t, x_i^t is updated by a definite rule depending on the states of the nearest two cells and itself. Namely, the value x_i^{t+1} is determined from x_{i-1}^t, x_{i+1}^t by a function $\Phi : \langle 0 \mid 1 \rangle \times \langle 0 \mid 1 \rangle \times \langle 0 \mid 1 \rangle \rightarrow \langle 0 \mid 1 \rangle$:

$$x_i^{t+1} = \Phi\begin{pmatrix} x_{i-1}^t \\ x_i^t \\ x_{i+1}^t \end{pmatrix}, \tag{1}$$

where $t \in \mathbb{Z}_{\geqslant 0}$, $i \in \mathbb{Z} / \mathbb{N}\mathbb{Z}$, and $\mathbb{N} \gg 1$ is the number of cells. There exist 2^{2^3} different rules, and each rule is referred to by a number n_Φ:

$$n_\Phi = \sum_{\mathcal{S}_1=0}^{1} \sum_{\mathcal{S}_2=0}^{1} \sum_{\mathcal{S}_3=0}^{1} \mathcal{F}\begin{pmatrix} \mathcal{S}_1 \\ \mathcal{S}_2 \\ \mathcal{S}_3 \end{pmatrix} 2^{4\mathcal{S}_1+2\mathcal{S}_2+\mathcal{S}_3}. \tag{2}$$

Example:

$$\text{if } \Phi\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \text{ then } n_\Phi = 110.$$

This rule also can be expressed as a bitwise operation:

$$\langle\langle 111 \longmapsto 0\rangle, \langle 110 \longmapsto 1\rangle, \langle 101 \longmapsto 1\rangle, \langle 100 \longmapsto 0\rangle, \langle 011 \longmapsto 1\rangle,$$
$$\langle 010 \longmapsto 1\rangle, \langle 001 \longmapsto 1\rangle, \langle 000 \longmapsto 0\rangle\rangle,$$

where a sequence of zeros and ones as the first element in the bracket indicates one of eight possible states of three adjacent cells, and the evolution at the next time step is indicated by the second element in the bracket. The disjunctive normal form of rule 110 can be expressed as

$$\begin{aligned} &(x_{i-1} \wedge x_i \wedge \neg\, x_{i+1}) \vee \\ &\quad (x_{i-1} \wedge \neg\, x_i \wedge x_{i+1}) \vee (\neg\, x_{i-1} \wedge x_i \wedge x_{i+1}) \vee \\ &\quad (\neg\, x_{i-1} \wedge x_i \wedge \neg\, x_{i+1}) \vee (\neg\, x_{i-1} \wedge \neg\, x_i \wedge x_{i+1}), \end{aligned} \tag{3}$$

with the Boolean variables x_i, the disjunction denoted by $\vee$, the conjunction by $\wedge$, and the negation by $\neg$. Understanding the global dynamics induced by the global map $\mathcal{F}$ is the long-standing challenge of CA theory. CAs with a finite number of $\mathbb{N}$ cells will eventually always become periodic (after at most $2^{\mathbb{N}}$ steps). Rule 110 is among the most complex elementary CA rules, as it has been proven that it is capable of universal computation in the Turing sense. We translate the DNA of CA rules to algebraic expressions, which will allow for a further analysis of CA dynamics. In the representation chosen here, the conjunction $\left(x_i \wedge x_j\right)$ is expressed by the algebraic multiplication $\left(x_i * x_j\right)$, the disjunction $\left(x_i \vee x_j\right)$ is expressed by the algebraic relation $\left(x_i + x_j\right) - \left(x_i * x_j\right)$, and the negation $\neg\, x_i$ is expressed by $\left(1 - x_i\right)$. All CA rules

$$x_i^{t+1} = \Phi\begin{pmatrix} x_{i-1}^t \\ x_i^t \\ x_{i+1}^t \end{pmatrix}$$

expressible as Boolean functions can be written in such a form, where the state of the cell i at time $t+1$ is given by an algebraic expression of the neighborhood states at time t.

Example: The algebraic expression that is discussed by example of the prototypic rules 110:

$$\Phi\left(x_i^{t+1}\right) = x_i^t + x_{i+1}^t - x_i^t * x_{i+1}^t - x_{i-1}^t * x_i^t * x_{i+1}^t. \tag{4}$$

3. Cellular Memory

Biological memory can be defined as a sustained cellular response to a transient stimulus. To understand this phenomenon, we must consider how the properties of different biological systems achieve memory of a stimulus, essentially permitting a cell to produce a lasting response. One way that cells accomplish this task is through transcriptional states, which involve populations of molecules regulating gene expression. If the transcriptional response is bistable, a chemical state becomes defined as on or off and, given certain parameters, this state can be inherited through DNA replication and cell division. In this way, a cell can produce a lasting memory of a biological response. In psychology, memory is an organism's ability to store, retain and recall information and experiences. In addition to the conventional function of the brain, memory has been used in systems biology recently to investigate the ability of small systems to store information. For example, cellular memory has been used to describe the ability of biological systems to maintain a sustained response to a transient stimulus as well as two or more discrete stable states. An extensive discussion on the applications of spatially distributed excitable chemical systems for information processing can be found in the literature [6, 7]. One of the possible translations of chemical dynamics into the language of informatics comes from the digital approach to computing and from binary logic [8, 9]. Two logical states can be easily identified in an excitable chemical system. The logical "true" state corresponds to concentrations of reagents that characterize excitation, whereas the logical "false" state is related to the unexcited state. A pulse of excitation traveling in a chemical medium may be regarded as a propagating bit of information. Within such an interpretation, no information is recorded or processed when the system remains in the stationary state. This property is plausible for those who try to find links between chemical information processing and information processing in biological systems, because the medium acts in a pretty obvious way—it does nothing if it is not excited by some external factors. If external excitations act on the system, the information about them spreads out in the medium in the form of diffusion-driven pulses of concentration. Information coded in pulses of concentration can be processed in areas of space where signals interact. It has been demonstrated that many operations on chemical signals can be easily performed in a nonhomogeneous chemical medium. Such a medium is usually built of active areas, where the system is excitable, and passive areas, where the system has a single, strongly attracting stationary state, so the excitations are rapidly dumped. A few simple rules describe the behavior of pulses in a typical excitable medium. An excitable system is characterized by a refractory time. It means that the system has to relax for a period of time before it can be re-excited. The time after which a subsequent excitation is successful depends on the strength of excitation. The form of this dependence leads to many interesting effects related to the transformation of signal frequency after crossing a barrier [10, 11]. The refractory stage of an excitable medium is responsible for the annihilation of colliding pulses. After head-on collision of two pulses in a channel, the excitation disappears because the medium behind the pulses is in a refractory state, so the excitation introduced by one of the colliding pulses does not excite the medium behind the other one. This property was used to construct the chemical information diode, which was one of the first signal processing devices analyzed in numerical simulations and realized in experiments [10–12]. One of the realizations of a chemical memory is based on an observation that a pulse of excitation can be stored on an excitable area; therefore, an excitable medium can be regarded as a memory cell capable of storing one bit of information. If there is a pulse, the state of memory corresponds to the logical 1; if there is no pulse, the state of memory corresponds to the logical 0. Cells must sense and dynamically respond to both internal and external signals. This often requires the synthesis or modification of transcription factors, which form an interaction network whose design determines the speed and sustainability of protein output in response to an environmental input. In this paper, we discuss how the information processing performed by a reaction-diffusion process can be utilized as a transformation operator in the CA world.

Cellular organization in biology has been an inspiration in several research fields, such as the description and definition of CAs [13–16], neural networks [17, 18], associative memory [19–22], neural computation [23–29], spiking systems [30] and others. A CA is a bio-inspired processing model for problem solving, initially proposed by von Neumann. This approach modularizes the processing by dividing the solution into synchronous cells that change their states at the same time in order to get the solution. The communication between them and the operations performed by each cell are crucial to achieve the correct solution.

4. Reaction-Diffusion Processes

Diffusive phenomena and reaction processes play an important role in many areas of physics, chemistry and biology and still constitute an active field of research. Systems in which reactive particles are brought into contact by a diffusion process and transform often give rise to very complex behaviors. Pattern formation is also a typical example of such a behavior in reaction-diffusion processes. In many reaction-diffusion problems, a particle-based model such as a lattice gas dynamic provides a useful approach and efficient numerical tools. For instance, processes such as aggregation, formation of a diffusion front, trapping of particles performing a random walk in some specific region of space, or the adsorption of diffusing particles on a substrate are important problems that are difficult to solve with the standard diffusion equation. A microscopic model based on CA dynamics is therefore of clear interest.

Reaction processes as well as growth mechanisms are most of the time nonlinear phenomena characterized by a threshold dynamic. While they are naturally implemented in terms of a point particle description, they may be very difficult to analyze theoretically and even numerically with standard techniques, due to the important role that fluctuations may play. In the simplest cases, fluctuations are responsible for symmetries breaking, which may produce interesting patterns. These systems depart from the behavior predicted by the classical approach based on differential equations for the densities. The reason is that they are fluctuation driven and that correlations can not be neglected. In other words, one has to deal with a full N-body problem, and the Boltzmann factorization assumption is not valid. For this kind of problem, a CA approach turns out to be a very successful one. CA particles can be equipped with diffusive and reactive properties in order to mimic real experiments and model several complex reaction-diffusion growth processes in the same spirit as a CA simulates a fluid flow. Chemical reactions such as $\mathbb{A} + \mathbb{B} \longmapsto \mathbb{C}$ are treated in an abstract way, as a particle transformation phenomenon rather than a real chemical interaction. Within the CA approach, there are two ways of modeling a spatially extended system with local reactive interactions

- The first one is to use a standard CA scheme: each cell is updated according to the state of its neighbors.
- The second way is to consider a lattice gas approach.

Here we continue the studies with the first kind of model, in which some reactive phenomena can be nicely described by a simple rule without going into the details of lattice gas automata such as the space partitioning paradigm.

5. Boolean Automata with Cellular Reaction-Diffusion Memory

In a classical way of modeling a spatially extended system with local reactive interactions due to a standard CA scheme, excitation of a cell takes three states: resting, excited and refractory. A resting cell becomes excited if the number of excited neighbors exceeds a certain threshold, an excited cell becomes refractory, and a refractory cell returns to its original resting state (Figure 1). The retained excitation model allows an excited cell to remain in the excited state if a number of excited neighbors remain in some threshold. The cell takes a refractory state otherwise (Figure 1). We classify different types of rules of excitation based on how the dynamics of excitable lattices develop after initial stimulation. A resting cell excites depending on the numbers of excited and also refractory neighbors.

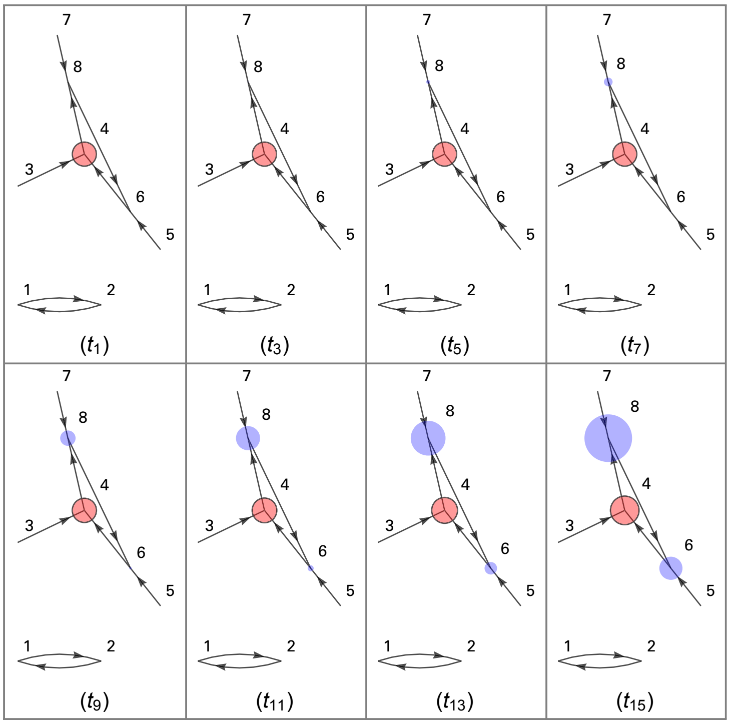

Figure 1. Schematic of spontaneous particle annihilation (at rate λ) and configurations for autocatalytic particle creation (at rate $\lambda / 4$) with respect to time t for our realization of a Boolean automata model on a $d = 2$ square lattice. k is the number of neighboring pairs of particles. Cells in automata proposed here undergo state transitions depending on the sums of excited and refractory cells in the neighborhood.

Cells in automata studied here undergo state transitions depending on sums of excited and refractory neighbors in the neighborhood of the cell. The interval-based local transitions rules exhibit rich dynamics of discrete reaction-diffusion and excitation patterns and intriguing phenomena in their spacetime dynamics. The nonlinear reaction-diffusion models involve spatial discrete systems where particles reside at the sites of a periodic lattice. Automaton particles spontaneously annihilate at a specified rate λ and are autocatalytically created, given the presence of nearby particles at rates depending on the local configuration that exhibit a rich behavior. These phenomena are analyzed utilizing the appropriate set of discrete reaction-diffusion equations (corresponding to lattice differential equations). In the first step of these studies, the nonlinear reaction-diffusion models involve spatially discrete systems where particles reside at the sites of a periodic lattice or grid. In the most general model, particles are added to (creation reaction), removed from (annihilation reaction) or shifted between (diffusive hopping) such sites. It is appropriate to note that this reaction model is equivalent to a spatial epidemic model where individuals are distributed on a periodic grid. Individuals can be either sick or healthy, and both the spontaneous rate of recovery of sick individuals and the rate of infection of healthy individuals by pairs of sick neighbors depend on the local configuration. Although not emphasized earlier, our discrete reaction model should be regarded as a stochastic Markov process, particle creation and annihilation occurring with a rate λ (probability distribution function) based on the weighted mean of particles on neighboring sites. A precise determination of behavior can be achieved via Monte Carlo simulations (although we caution that the analysis involves approximations). Our lattice realization of the model is a stochastic Markov process on a square lattice with periodic boundary conditions, with each site labeled by $i = (i_1, i_2, \ldots, i_N)$. It involves the following components:

- spontaneous creation of particles at unoccupied sites at rate λ
- annihilation of particles at empty sites at rate $k / k_{\max}$, where k is the number of pairs of particles on neighboring sites and $k_{\max}$ is the sum of possible values

Thus, the maximum number of pairs of particles satisfies $k_{\max} = 2d(d-1)$, corresponding to the case where all $2d$ neighboring sites are occupied. For some specific value of λ, for example $\lambda \leqslant 1$, the active state is more stable and displaces the absorbing inactive state, leading to perpetual propagation. While for $2 < \lambda \leqslant 3$, the inactive state is more stable and displaces the active state, leading to propagation. Or for example, a small localized perturbation with nonzero particle population of the zero state will not necessarily grow, but rather shrink. There is a critical size and profile at which growth and spreading occurs. The basic behavior of the probability distribution function for different values of λ is shown in Figure 2.

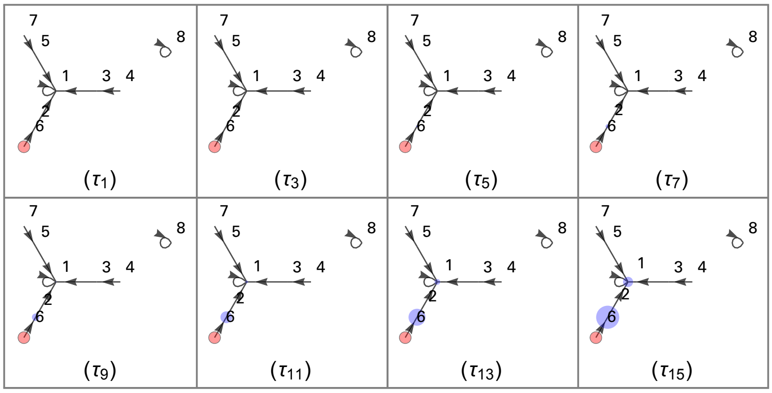

Figure 2. Time evolution of early development phase in the life of proposed automata network on periodic square grid of 3×3 cells with parameter $0.1 \leqslant \lambda \leqslant 0.3$. Time t flows from left to right and from upper to lower panel. Blue dot represents the excitatory cell, red dot the inhibitory cell and cross dots dead cells that differentiate into output of automata network.

Figure 2 shows the extent of artificial pattern propagation measured by a quantum stochastic walk (QSW) on directed graphs with $\mathbb{N} = 8$ nodes. The walks start at node i_1; red and blue circles represent the probability distributions of excitatory and inhibitory cells in order. To give a definition of our CA, we introduce an excitable automaton with reaction-diffusion cellular memory, where a resting cell is excited if a number of its excited neighbors belong to a dynamic interval

$$\theta = \begin{pmatrix} \theta_\varsigma \\ \theta_\xi \\ \theta_\Psi \end{pmatrix}.$$

The interval θ is called an excitation interval, which is dynamically updatable at every step t, depending on the state of the cell σ and the numbers of excited and refractory neighbors in the cell σ neighborhood. All cells update their states independently of each other with two different rules, which we will describe later. For two-dimensional CAs with a two-cell neighborhood ,we found that by tuning θ we can persuade the automaton to imitate almost all kinds of excitation dynamics. Let σ^t and σ^{t+1} be states of a cell σ at time steps t and $t+1$, and $\varsigma^t(x)$ and $\xi^t(x)$ be sums of excited and refractory neighbors in the cell σ neighborhood. The cell updates its state by the following relation:

$$\psi_{\sigma_i}^{t+1} = \sum_{i-1}^{i} \delta\left(\psi_{\sigma_i}^{t}, 1\right) \tag{5}$$

$$\psi_{\sigma_i}^{t} = \begin{cases} \psi_{\varsigma} = \delta\left(\psi_{\sigma_{i-1}}^{t}, 1\right) + \delta\left(\psi_{\sigma_i}^{t-1}, 1\right) \\ \psi_{\xi} = \delta\left(\psi_{\sigma_{i+1}}^{t}, 1\right) + \delta\left(\psi_{\sigma_i}^{t-1}, 1\right) \end{cases} \tag{6}$$

$$\begin{aligned} \phi_{\sigma_i}^{t+1} = & \sum_{i}^{i+1} \delta\left(\phi_{\sigma_i}^{t}, 2\right) + \sum_{i-1}^{i+1} \delta\left(\phi_{\sigma_i}^{t}, 2\right) \wedge \\ & \sum_{i-1}^{i} \delta\left(\phi_{\sigma_i}^{t}, 1\right) \wedge \sum_{i}^{i+1} \delta\left(\phi_{\sigma_i}^{t}, 1\right) + \sum_{i-1}^{i+1} \delta\left(\phi_{\sigma_i}^{t}, 0\right) \end{aligned} \tag{7}$$

$$\phi_{\sigma} = \begin{cases} \phi_{\varsigma} = \delta\left(\phi_{\sigma_{i-1}}^{t}, 1\right) + \delta\left(\psi_{\sigma_i}^{t-2}, 1\right) \\ \phi_{\xi} = \delta\left(\phi_{\sigma_{i+1}}^{t}, 1\right) + \delta\left(\psi_{\sigma_i}^{t-3}, 1\right) \end{cases} \tag{8}$$

where the sum runs over the two nearest neighbors of site σ_i and δ is the Kronecker delta function. Let us consider a two-dimensional CA in a square lattice, every cell of which takes two states: state 0 and state 1, and updates its state in discrete time. The next state of the cell in the next generation depends on its own state and its two neighbors. Let $\varsigma^t(x)$ and $\xi^t(x)$ be the cell σ neighbors in one of these possible states, at time step t. Then the cell-state transition function will be

$$\sigma^{t+1} = \theta\left(\psi^{\vec{t}}, \varsigma^{\vec{t}}, \xi^{\vec{t}}\right) \tag{9}$$

where $\vec{t}$ represents the states of cell σ and neighbors in the previous time steps (as a three-dimensional time vector). A cell in state α updates its state depending on transition function $\varsigma^{\vec{t}}$ and the number of neighbors in state β; a cell in state β updates its state depending on transition function $\xi^{\vec{t}}$ and the number of neighbors in state α. A natural way to compute excitation interval θ boundary values is

$$\theta_{\sigma}^{t+1} = \begin{cases} 2, & \psi_{\sigma}^{t} = 1 \wedge \phi_{\sigma}^{t} = 1 \\ 1, & \psi_{\sigma}^{t} = 0 \wedge \phi_{\sigma}^{t} = 1 \\ 3, & \psi_{\sigma}^{t} = 1 \wedge \phi_{\sigma}^{t} = 0 \\ 0, & \psi_{\sigma}^{t} = 0 \wedge \phi_{\sigma}^{t} = 0 \end{cases} \tag{10}$$

where the sums run over the four nearest neighbors of site σ_i and δ is the Kronecker delta function. The low boundary θ of excitation of cell σ is updated only if cell σ is excited. The boundary increases if cell x has more excited neighbors than refractory neighbors; the middle boundary θ decreases if the number of refractory neighbors of σ_i exceeds the number of excited neighbors. The boundary does not change if cell σ_i has the same number of excited neighbors as refractory neighbors. To exhibit a wide range of spacetime dynamics based on an interplay between propagating excitation patterns that modify the excitability of the automaton cells, multiple experiments are conducted on a cellular array of $n \times n$ cells with periodic boundary conditions. In a typical experiment, we start from an initial condition at time $t = 0$, with the given configuration (vector of size n with single active cell) of values ψ_i^t on the lattice, then the system starts to evolve as follows:

```
1. Define null matrix $k_{n\times n}$
2. Start at time $t = 0$
3. Define initial population and their positions on $k$, such that $k_{ij} \in \{0, 1\}$, $i, j = 1, 2, \ldots, N$
4. While $t <$ max transients:
    4.1. Run CA on initialize configuration $k$
    4.2. $t = t + 1$
5. $t = 0$
6. While $t <$ end:
    6.1. For each cell $i$ and $j$ in $k_{ij}$ do:
        6.1.1. Compute state of $\phi(\varsigma, \xi)$
        6.1.2. Compute state of $\psi$
        6.1.3. Compute weighted interval $\theta$
        6.1.4. Construct transition function $\mathcal{F}(\psi, \phi, \theta)$
        6.1.5. Update $\psi$, $\phi$, $\theta$
```

6. Numerical Simulations

6.1 Stability of Trajectories: The Maximum Lyapunov Exponent

Quantifying instability in physical systems and in mathematical models is a long-standing problem in nonlinear science, beginning with Lyapunov's pioneering work at the end of the nineteenth century. Lyapunov discovered that the basic quantities are exponential rates that, when positive, measure divergence from an unstable trajectory. In this paper, we elaborate on the well-known fact that instabilities often do not affect all components of a system to the same extent; more precisely, we study how fast defects may spread among these components, which we assume are spatially distributed. Let us consider a binary CA that is evolved from an initial state ξ_0 generating a trajectory ξ_1, $t = 0, 1, \ldots$. By analogy with continuous dynamical systems, the idea is to measure the effect of a small perturbation of ξ_0 on the evolution at later times. In his classic work, Wolfram [13–16] considered damage spreading, that is, growth of the set of affected sites in ξ_t when a few sites in ξ_0 are flipped. In one dimension, there are two directions of propagation; when the maximum extent of damage progresses linearly, the two slopes are called Lyapunov exponents, as they measure the exponential divergence in distance between the original and perturbed states in the appropriate metric. The Boolean derivative also is used as the analog for the Jacobian, which leads to the branching walk dynamics of defects that we now informally describe. It is possible to develop a formula very similar to the one used in continuous systems. The Boolean derivative is defined by

$$\frac{\partial \psi_n(s)}{\partial s_p} = \psi_n\left(\ldots, s_p, \ldots\right) \oplus \psi_n\left(\ldots, 1 - s_p, \ldots\right) \tag{11}$$

where $\oplus$ represents the sum modulo two (XOR). This quantity measures the sensitivity of the function $\psi_n$with respect to a change in s_p. The Jacobian matrix $\mathcal{J}(s)$ has components $\mathcal{J} = \partial \psi_n(s) / \partial s_p$. Recall that a trajectory ξ_t of the system is fixed. Assume a defect is present at a site y at time t. That defect looks into each of its neighborhood sites x to check whether flipping the state ξ_t at y would produce a different state at x than assigned by ξ_{t+1}; if so, the defect produces a successor at x. Each defect may produce more than one successor and acts independently of other defects. The exponential rate of accumulation of such defects is called the maximal Lyapunov exponent (MLE). There is some conceptual similarity between this object and the Lyapunov spectrum in multidimensional smooth dynamical systems, whereby the spectrum of the Jacobian accounts for perturbations in all directions in both the input and the output. In the discrete CA configuration space, there is essentially one way to make an infinitesimal perturbation in the input, but the effect can be quite different in different directions of the output. Moreover, we empirically observe that typically the direction with the maximal effect has the profile height that is close to the MLE. The Lyapunov exponent of a CA according to Bagnoli et al. [31] is computed by tracking all defects that emerge from the introduction of a single defect in the CA's initial configuration, that is, by flipping one of the cells' state, as follows:

$$\lambda = \lim_{t \to \infty} \frac{1}{t} \log\left(\frac{\xi_t}{\xi_0}\right) \tag{12}$$

where ξ_t denotes the total number of defects at the t step and a defect is defined as the smallest possible perturbation of the CA, such that it may be conceived as an object that flips that state of the cell in which it resides. Upon introducing such a defect to one of the CA's cells, it will propagate and affect increasingly more cells, because the dynamic of the CA is governed by its neighborhood configurations. If $\lambda < 0$, the perturbation vanishes exponentially with time, or grows exponentially if $\lambda > 0$. Therefore the system is chaotic (unstable dynamics extremely sensitive to noise or perturbations) while $\lambda > 0$; otherwise, it has stable dynamics insensitive to perturbations and noise ($\lambda < 0$) (Figure 3). Systems with $\lambda = 0$ are special, as their dynamics are sensitive to noise and perturbations without "overreacting" like chaotic systems. These systems at the "edge of chaos" adapt to fluctuations but remain close to their unperturbed dynamics.

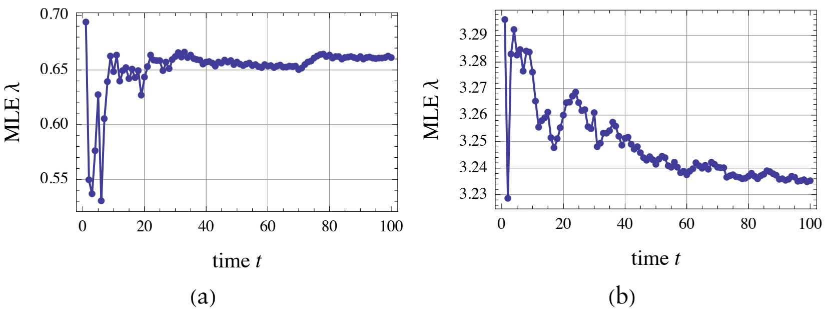

Figure 3. Maximum Lyapunov exponents λ versus time t. (a) CA rule 30. (b) Reaction-diffusion automata with memory that exhibit complex chaotic dynamics.

The quantity ξ_t should be computed by evolving both the initial configuration at $t = 0$ and its perturbed counterpart. Clearly, if we redefine ξ_t as a vector of ξ_t whose ith element ξ_t^i is the number of defects in cell x_i at a given step, we have a construct that incorporates both the multiplicity and position of defects. For comprehensiveness, it should be mentioned that Bagnoli et al. [31] do incorporate such a damage vector to define the Lyapunov exponent of elementary CAs, but they discard its information content, as their further analysis is based upon the sum of its elements. Besides, they call the quantity given by equation (12) the MLE of a CA, by analogy to continuous n-dimensional dynamical systems where the terms refer to the largest exponent in the Lyapunov spectrum that contains the perturbations and rates of separation, in all different directions.

6.2 Graph Locality of Quantum Walk and Constancy of the Speed of Propagation

Quantum computing promises to deliver faster computation based on the principles of quantum mechanics. Quantum walks also have been shown to be fruitful tools in analyzing the dynamic properties of quantum systems. Quantum walks are used extensively as algorithmic tools for quantum computation. Farhi and Gutmann [32] investigated the dynamics of continuous-time quantum walks on undirected graphs by introducing quantum systems whose Hamiltonians are based on the adjacency matrix of a graph. A graph is a symbolic representation of a network and of its connectivity. Of particular importance in computer science is the relationship between graphs, Markov chains and classical discrete random walks. Classical random walks on graphs have been crucial to the development of stochastic algorithms. In consequence, quantum walks on graphs have become an active area of research in quantum computation. Based on the theory of Markov chains, classical random walks (CRWs) study the evolution of the probability distribution of an abstract walker's position on a graph. In a CRW, a walker moves along the edges of a connected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ of $\mathcal{N}$ nodes, with a hopping probability assigned to each edge $e \in \mathcal{E}$ between nodes $i \in \mathcal{V}$ and $j \in \mathcal{V}$. For directed graphs, edges in $\mathcal{E}$ are ordered pairs of nodes. An edge between nodes i and j is called bi-directed if both (i, j) and (j, i) are in $\mathcal{E}$. Walks on undirected graphs can be modeled as walks on the corresponding directed graph with bidirectional edges. On any graph of vertices, a Markov chain can be defined. A Markov chain is a sequence of events that is governed by a stochastic process in which the results of a time step depend only on the results of the previous time step. Markov chains are described by a stochastic matrix $\mathcal{M}(n \times n, \mathbb{R})$, with $\sum_{j=1}^{n} m_{ij} = 1$ and entries m_{ij} representing the weight of the directed edge going from vertex i to j. These weights can be interpreted as a transition rate from site i to j. Repeatedly applying $\mathcal{M}$ to an n-dimensional stochastic vector τ with $\sum_{j=1}^{n} \tau_i = 1$ evolves an initial probability distribution through discrete time steps. The probability of being at vertex i changes according to

$$\frac{d\tau_i}{dt} = -\sum_j \mathcal{M}_{ij} \tau_j(t). \tag{13}$$

Markov chains on regular undirected graphs result in a stationary probability distribution τ_s that is independent of the initial state. The time it takes to reach the stable distribution is called the mixing time. Markov chains with equal probability of jumping from site i to any of the d sites adjacent to i are also known as random walks on a graph. In the quantum equivalent of random walks, a quantum walker walks between sites by changing its position state $|x\rangle \in \{|1\rangle, \ldots, |n\rangle\}$. The difference from classical walks is twofold: First, the various paths are realized in a superposition and thus interfere with one another, and a measurement collapses the paths into the current position. Second, the dynamics have to preserve the squared amplitude vector instead of the stochastic vector to preserve the total probability. This means that the evolution has to be unitary or, in the most general case of an open system, a completely positive trace-preserving map. In this paper, we study quantum stochastic walks (QSWs) [33] on a one-dimensional (1D) ring lattice of N nodes in which every node is connected to its nearest neighbors. This generalized regular network has broad applications in various coupled dynamical systems, including biological systems, neural networks, synchronization and small-world networks, and many other self-organizing systems. We analyze quantum walks on our proposed networks with periodic boundary conditions and we derive analytical expressions for the transition probabilities between two nodes of the networks. In a quantum walk, a state vector $|\psi, t\rangle$ undergoes a time evolution given by the solution of the Schrodinger equation:

$$\frac{d|\psi, t\rangle}{dt} = e^{-i\mathrm{H}t}|\psi, t = 0\rangle \tag{14}$$

where H is the Hamiltonian whose matrix elements come from the CRW generator matrix equation:

$$\langle i|\mathrm{H}|j\rangle = \mathcal{M}_{ij}, \tag{15}$$

the mass $m = 1$ and $\hbar = 1$ is assumed in the preceding equation. To get the exact solution of equation (14), all the eigenvalues and eigenvectors of the transfer operator and Hamiltonian are required. In the case of an undirected (or bidirected) graph, as studied by Farhi and Gutmann [32], the Hamiltonian H is set equal to the symmetric adjacency matrix A of the graph. For directed graphs, the adjacency matrix will not in general be Hermitian, and thus the time evolution will not be unitary. Therefore, this approach does not produce valid Hamiltonians for quantum walks. This paper follows the perspective of dynamical systems and complexity research and investigates a new approach to introduce quantum physics into CA networks by making use of the theory of quantum walks. To simulate a CA network's dissipative dynamic, we therefore need to focus on quantum walks that incorporate decoherence. To obtain the basic features of the dynamical property of the proposed model, a continuous QSW [33] on the lattice is implemented and analyzed. The genius of our model lies in the fact that operations on the different cell states and their neighbors as a local unit impose dynamics on the global network state. These global dynamics can be understood as a quantum walk on the states of the network, which can be defined as an evolution:

$$|\psi\rangle t_0 \rightarrow |\psi\rangle t_1 \rightarrow |\psi\rangle t_2 \rightarrow \ldots. \tag{16}$$

The vertices of the graph are given by binary strings denoting all possible different patterns. The connectivity thus depends on the updating protocol. If all cells are updated synchronously, each transition between different patterns is theoretically possible and the graph is fully connected. We concentrate on the more common case of updating a single cell at a time t. CAs as described before are defined by a mapping from the values of a cell and its neighbors to the next value of that cell. If the value of a cell and its neighborhood are matched to a state, the next state of that range can be defined if the values of the cells to the right and left of the range are also known. Assuming that the values in the cells immediately to the right and left of the neighborhood are not equally likely, we create a local Markov chain for a given range of a cell and its neighborhood.

The dynamic behavior of any CA is visualized and studied in terms of either its spacetime pattern or its basin-of-attraction field. The latter is essentially a graph, which may or may not consist of disjoint subgraphs, and is commonly referred to as the state transition diagram (STD) of the CA. In other words, the STD of a CA is essentially a directed graph $\mathcal{G}$ where each node represents one of the states of the CA and the edges signify transitions from one state to another. As discussed earlier, for an n-length CA, the complete STD contains all the $2n$ states. Each node in the graphs represents one of the 2^n possible configurations of the N sites. Possible sequences of site values are represented by possible paths through the graph. Each vertex on the graph represents one of the 2^n states of the CA and is joined to the vertex representing the state reached after one step in the next evolution. It should be noted that the STD of a CA rule is necessarily a directed graph, so the obvious way to represent an STD is to use an adjacency matrix.

As we just described, since in continuous quantum walks the structure of the graph needs to be undirected to ensure the hermiticity of the Hamiltonian, we used the QSW method. A graph $\mathcal{G}$ is defined as a set of vertices (or nodes) $\{1, \ldots, N\}$ together with an $N * N$-dimensional adjacency matrix A describing the connections (or edges) between them. If A is symmetric, so that $A_{ij} = A_{ji}$ for all i, j, $\mathcal{G}$ is said to be undirected. Otherwise it is a directed graph (or digraph).

We note that the diagonal entries A_{ij} represent self-loops, edges that start and finish at the same vertex; if $\mathcal{G}$ has no self-loops it is called simple. A (continuous-time) CRW on $\mathcal{G}$ may be defined as an N-component probability vector $\mathcal{P}(t)$ whose components sum to unity and represent the probability for the hypothetical walker being found at vertex i at time t. The initial condition typically places the walker on a particular vertex at time zero $\mathcal{P}(0) = \delta_{ij}$ for some j. The time evolution of $\mathcal{P}(t)$ is governed by a master equation:

$$\frac{d\mathcal{P}}{dt} = -\mathcal{M}\,.\,\mathcal{P}(t) \tag{17}$$

where generator matrix $\mathcal{M}$:

$$\mathcal{M}_{ij} = \begin{cases} -\gamma\, A_{ij}, & i \neq j \\ \gamma\, \text{outDeg}(j), & i = j \end{cases} \tag{18}$$

where $\gamma > 0$ determines the CRW transition rate between vertices. The off-diagonal elements of $\mathcal{M}_{ij}$ represent the individual probability flows $j \to i$ along each edge from vertex j, while the diagonal elements $\mathcal{M}_{jj}$ account for the total outflow from vertex j per unit time. The solution of equation (17) can be expressed as a matrix exponential:

$$\mathcal{P}(t) = e^{-\mathcal{M}\text{t}.\mathcal{P}(0)}. \tag{19}$$

A quantum walk (QW) may be defined in a natural way by analogy to the classic random walk (CRW). The graph structure is mapped onto a quantum mechanical Hilbert space in which the set of vertices forms an orthonormal basis $\{|1\rangle, \ldots, |N\rangle\}$. The probability vector $\mathcal{P}(t)$ from the CRW is replaced by a quantum state vector of probability amplitudes

$$|\psi(t)\rangle = \sum_{i=1}^{N} |i\rangle\, \langle i\, |\psi(t)\rangle. \tag{20}$$

The amplitudes $\langle i|\psi(t)\rangle$ represent a coherent superposition over all vertices, such that the probability associated with vertex i at time t is $|\langle i|\psi(t)\rangle^2$. It should be noted that, in order for the time evolution of the QW to be unitary (and hence probability-conserving), H must be Hermitian. From equation (15), it follows that M must be symmetric, and hence the graph $\mathcal{G}$ must be undirected for a QW. The solution to equation (14) can once again be expressed as a matrix exponential:

$$|\psi(t)\rangle = e^{-i\text{H}}|\psi(0)\rangle \tag{21}$$

with the initial condition $|\psi(0)\rangle = |q\rangle$ for some q. The QW just described is restricted to undirected graphs on which the Hamiltonian operator is Hermitian. It is also, by definition, limited to pure quantum states undergoing phase-coherent unitary evolution. By contrast, the CRW represents the opposite limit, in which scattering is nonunitary and evolution is purely phase incoherent. The QSW was proposed by Whitfield et al. [33] as a generalization that includes both QWs and CRWs as limiting special cases and allows random walks with a combination of coherent and incoherent dynamics. The incoherent part of the QSW can be used to incorporate scattering along directed edges and thus allows directed graphs to be treated. In place of the probability vector $\mathcal{P}(t)$ of the CRW or wavefunction $|\psi(t)\rangle$ of the QW, the QSW is framed in terms of the density matrix $\hat{\rho}$. In the general case, this operator describes a statistical ensemble of pure quantum states (mixed state):

$$\hat{\rho} = \sum_m \rho_m \mid |\psi(t)\rangle \langle \psi_m(t) \mid. \tag{22}$$

The weights $\rho_m \geqslant 0$ satisfy $\sum_m \rho_m = 1$ and represent the probabilities for the system to be in each of the quantum states $|\psi(t)\rangle$. This probability reflects a lack of knowledge of the true system state, which is distinct from the quantum mechanical uncertainty associated with the measurement of a single quantum state that happens to be a superposition of observable basis states. The special case of a known quantum state is recovered when only one of the ρ_m is nonzero, and is known as a pure state. We note several properties of $\hat{\rho}(t)$ that are relevant here:

- $\hat{\rho}(t)$ is Hamiltonian
- $\mathrm{tr}(\hat{\rho}(t)) = 1$
- $\mathrm{tr}(\hat{\rho}(t)^2) \leqslant 1$

with equality holding only for pure states; the expectation value of an operator $\hat{A}$ is given by $\langle \hat{A} \rangle = \mathrm{tr}(\hat{\rho}(t)\hat{A})$. For a QSW on graph $\mathcal{G}$, the natural representation for $\hat{\rho}(t)$ is an $\mathcal{N} \times \mathcal{N}$ matrix in the basis of vertex states $\{|1\rangle, \ldots, |N\rangle\}$, with elements $\rho_{ij} = \langle i|\hat{\rho}(t)|j\rangle$. As for the QW, the usual initial condition for a QSW will have the walker start in the state $|q\rangle$ at $t = 0$ so that $\hat{\rho}(0) = |q\rangle \langle q|$ or equivalently, $\rho_{ij} = \delta_{iq}\delta_{jq}$. For $t \geqslant 0$, the diagonal elements $\rho_{ij}(t)$ represent the probability density at vertex i (and are therefore referred to as populations), while the off-diagonal elements $\rho_{ij}(t)(i \neq j)$ describe the phase coherence between distinct vertices i and j (and are known as coherence). We can now define the QSW by the following master equation:

$$\frac{d\hat{\rho}}{dt} = -(1-\omega)\mathrm{i}[\hat{H}, \hat{\rho}(t)] + \omega \sum_{K=1}^{K} (\hat{\mathcal{L}}_k \hat{\rho}(t) \mathcal{L}_k^\dagger) - \frac{1}{2}(\mathcal{L}_k^\dagger \hat{\mathcal{L}}_k \hat{\rho}(t) + \hat{\rho}(t) \mathcal{L}_k^\dagger \hat{\mathcal{L}}_k) \tag{23}$$

where $\hat{H}$ is the Hamiltonian operator, describing coherent evolution; $\hat{\mathcal{L}}_k$ are the Lindblad operators, which describe phase-incoherent scattering; and $0 \leqslant \omega \leqslant 1$ is a weighting factor that interpolates between the coherent and incoherent terms. The first term on the right-hand side of equation (23), representing the phase-coherent component of the evolution, uses the same Hamiltonian $\hat{H}$ as in the QW. Indeed, when $\omega = 0$, equation (23) reduces to the Liouville–von Neumann equation, which is the density matrix equivalent of the Schrodinger equation (14). As for the QW, $\hat{H}$ must be Hermitian and hence based on an undirected graph. Unlike the QW case, however, in a QSW the graph $\mathcal{G}$ may be directed in general. We therefore define $\hat{H}$ in terms of a symmetrized version of $\mathcal{G}$, as follows:

$$H_{ij} = \begin{cases} -\gamma \max\left(A_{ij}, A_{ji}\right), & i \neq j \\ -\sum\limits_{n \neq j} H_{nj}, & i = j. \end{cases} \tag{24}$$

Example 1. Adjacency matrix (A), weighted adjacency matrix of undirected graph $\left(W^A\right)$ and related Hamiltonian matrix $(\hat{H})$ are computed numerically (symmetrized version):

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad W^A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix},$$

$$\hat{H} = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 2 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 0 & 2 \end{pmatrix}.$$

In this paper, we have decided to focus on Boolean automata networks whose evolution is governed by a specific updating rule. In this context, a network N is a set of nodes that interact over time. Each node has two possible states, called activity states. If we call $x(t) = x_i(t)_{i \in j} \in \Omega = \{0, 1\}^2$ the current configuration of the network

N at time t, the states of the nodes of this configuration are defined by:

$$\forall\, i \in N,\, x_i(t) = \begin{cases} 0, & \text{if } i \text{ is activated} \\ 1, & \text{otherwise.} \end{cases} \tag{25}$$

As mentioned earlier, the structure of our Boolean automata network N can be represented by a labeled directed graph called its interaction graph. In this graph, each arc (j, i) is labeled by an interaction weight $w_{ij} \in \mathbb{R}$. The sign of w_{ij} depends on the activating or inhibiting nature of the interaction that node j has on node i. If $w_{ij} > 0$ (resp. $w_{ij} < 0$), then node j is said to be an activator (resp. a repressor) of node i. If $w_{ij} = 0$, then node j does not act on node i (arc (j, i) does not exist in the interaction graph of the network). Let us write $|N|$ for the number of nodes of the network N and N_j to refer to the neighborhood of node i, that is, the set of nodes that are activators or repressors of node i. Then $j \in N_i \Longleftrightarrow W_{ij} \neq 0$. We define the interaction matrix $W_{N\times N}$ (also called the synaptic weights matrix in the context of neural networks) of the network. Its coefficient W_{ij} is the interaction weight that node j has on node i. In the interaction graph of the network, each node is also labeled by a value called its activation threshold. It represents the necessary quantity of interaction potential a node needs to become activated. We define the three-dimensional vector θ as the threshold vector, in which the coefficient $\theta_i \in \mathbb{R}$ gives the activation threshold of node i. Now we can describe the temporal evolution of our Boolean automata network. Informally, the new state of an arbitrary node i at time step $t+1$ depends on the sum of the interaction weights coming from its activated neighbors $\phi(\sigma)$ and $\psi(\sigma)$ at time step t. Formally, the local transition function is defined by:

$$\sigma^{t+1} = \theta\left(\vec{\psi^t}, \vec{\varsigma^t}, \vec{\xi^t}\right) \tag{26}$$

where δ is the Kronecker delta function, and θ is the interaction potential of node i. To describe the QSW method, here we have chosen to apply it to the analysis of a proposed CA network (Figure 4). The influence of reaction-diffusion memory will be explained by studying how its presence or absence acts on the asymptotic dynamical properties of the underlying network.

From the mathematical point of view, studies have shown that in different time steps this model can exhibit significantly different dynamical behaviors, according to their robustness against changes of their neighbor's nodes or cells.

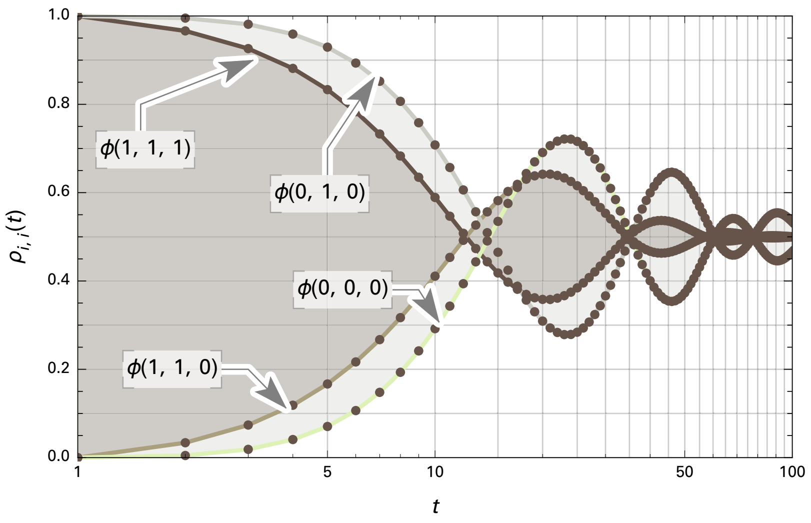

Figure 4. Estimated state transition probabilities with respect to time t by QSW on an eight-node graph of CAs with reaction-diffusion memory. Initial state $\langle 111\rangle\ \langle 110\rangle\ \langle 101\rangle\ \langle 100\rangle\ \langle 011\rangle\ \langle 010\rangle\ \langle 001\rangle\ \langle 000\rangle$, $\omega = 0.5$. The walker has a high probability of being in the desired output states $\langle 111\rangle\ \langle 010\rangle\ \langle 000\rangle\ \langle 110\rangle$ with probability $\mathcal{P} \approx 0.5$.

6.3 Statistical and Dynamical Complexities

Randomness in a dynamical system is related to the intrinsically unpredictable behavior of a state of the system. Therefore, we use a probabilistic approach to describe phenomena regardless of their conceptual differences, but there are situations where we want to talk about randomness itself, that is, characterize and/or quantify randomness. We therefore need a definition of randomness and a theoretical framework that allows us to develop and use these ideas. There have been some works trying to define randomness through different mathematical objects. For example, the concept of normal number due to Borel formalizes the notion of a random real number. One successful attempt that has become a mathematical theory is algorithmic information theory, also known as Kolmogorov complexity. It is based on the idea of patterns in a mathematical object. Recently the range of applications requiring random numbers was extended with the development of quantum cryptography and quantum information processing. An extensive discussion on the implementation and applications of random number generators can be found in the literature. One interesting way to design pseudorandom number generators (PRNGs) is connected to CA and chaos theory. In the past decade, CA-based PRNGs were studied extensively [13–16] and found to be superior over traditional approaches in areas ranging from random number generation [34–55] to cryptography [56–66]. The majority of research on CA-based PRNGs has been focused on the 1D CAs, also known as elementary CAs. Recently, there is a research trend such that increased complexity from hybrid CA cell configurations and increased CA dimensionality can lead to better performance. Researchers have tried to improve their results by designing new configurations with hybrid cell rules [36–39]. Wolfram in [34] first proposed 1D CAs as PRNGs. In particular, he extensively studied the bit sequences generated by rule 30 in his numbering scheme for 1D, $r = 1$ rules, where the rule number represents in decimal format the binary number encoding the rule table. Wolfram extensively studied the particular rule 30 CA, demonstrating its suitability as a high-performance randomizer that can be efficiently implemented in parallel. Indeed, this is one of the PRNGs that was shipped with the connection machine CM2 [67] and that is currently used in the Wolfram Mathematica software [68].

We would like to emphasize that any class 3 (chaotic) CA rule and rules that have large Lyapunov exponents and invariant entropies can be considered as a candidate random sequence generator. Autoplectic rules that produce complex patterns even from simple initial conditions are probably best. In addition, many rules that seem to produce chaotic overall patterns nevertheless yield sequences that show definite regularities, resulting, for example, in non-maximal temporal entropies. Chaotic systems are characterized by their high sensitivity to initial conditions and some properties like ergodicity, pseudorandom behavior and high complexity that make chaotic systems very attractive for implementing PRNGs. As one of the most important contributions in nonlinear science, classical chaos theory has been studied widely and applied in various contexts such as mathematics, physics, chemistry, computer science, biology and so on. Inspired by these reasons, we are motivated to search for novel chaotic systems. In this paper, we investigate CAs and propose a novel CA-based PRNG. By numerical simulations and performance comparisons in terms of quantifiers based on information theory, and various randomness tests, we found that the CA-based PRNG can be used to generate pure pseudorandom sequences. Careful mathematical analysis is required to have any confidence that a PRNG generates numbers that are sufficiently “random” to suit the intended use. All practical methods for obtaining random numbers are based on deterministic algorithms, which is why such numbers are more appropriately called pseudorandom, as distinguished from true random numbers resulting from some natural physical process. Random number generators must possess a number of properties if they are to be useful in lengthy stochastic simulations such as those used in computational physics. In practice, the output from many common PRNGs exhibits artifacts that cause it to fail statistical pattern-detection tests. These include, but are certainly not limited to

- lack of uniformity of distribution
- correlation of successive values
- short period (after a sufficient number of steps the generator comes back to some sequence of states that was already visited)

We would like to mention this is not a sufficient condition for randomness but allows us to discard some sequences as clearly not random. We will briefly describe the relatively intuitive statistical tests that will be applied to data samples taken from the random generator, which we consider to be sufficient in qualifying the device for its use in the experiment. In the following we will start by a tiny selection of possible statistical tests. However, in order to demonstrate the efficacy of a proposed random number generator, we will subject it to a battery of empirical and theoretical tests that we will describe later, and finally, by an extensive suite of statistical tests and the results of experiments, we have shown that our CA algorithm can be applied to the difficult problem of generating random numbers.

- Equi-distributional test: Provided that the sample dataset is sufficiently long, all possible n-bit blocks where n is the length of the block should appear with equal probability within the dataset. A direct way of determining the equi-distributional nature of a dataset is to evaluate the mean value of all n-bit blocks, which should be $(2^n - 1)/2$. This will give the same result for any symmetric distribution.
- Blocks of k zeros or ones: Another test for the randomness of a set of bits is counting blocks of consecutive zeros or ones. Each bit is equally likely to be a zero as a one; therefore, the probability of finding blocks of concatenated zeros or ones should be proportional to a 2^{-k}.

In contrast, we expect a random sequence to lead to equal probabilities for all these cases:

$$\begin{cases} \mathcal{P}(0) = \mathcal{P}(1) = \dfrac{1}{2} \\ \mathcal{P}(00) = \mathcal{P}(01) = \mathcal{P}(10) = \mathcal{P}(11) = \dfrac{1}{4} \\ \mathcal{P}(000) = \mathcal{P}(001) = \mathcal{P}(111), \ldots, = \dfrac{1}{8}. \end{cases} \tag{27}$$

We can generalize this restriction on the probabilities as:

$$\mathcal{P}(k \text{ bit sequence}) = \frac{1}{2^k}. \tag{28}$$

We would like to mention this is not a sufficient condition for randomness, but allows us to discard some sequences as clearly not random. Its advantage is that checking this condition is an algorithmic procedure that can be applied to any sequence. In this paper we did a related experimental setup, where we analyze the different block size of bits, starting from counting the number of zeros and ones in the bit stream extracted for testing (Figure 5). We have experimentally generated the test sequences; for a given n, there are 2^k different strings forming the set s_k. Therefore

$$s_1 = \{0, 1\} \tag{29}$$

$$s_2 = \{00\,, 01, 10, 11\} \tag{30}$$

$$s_3 = \{000, 001, 010, \ldots, 111\}, \tag{31}$$

where string $i \in s_m$ as $\mathcal{P}(i) = k(i) \big/ N$, $k(i)$ is the number of times that the string i is present in a sequence subdivided into segments of k symbols, and N is the total number of strings of length k in the sequence.

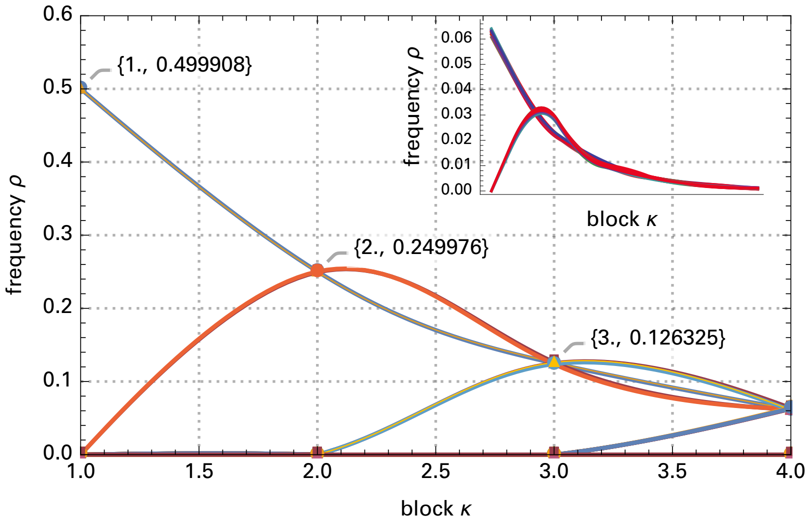

Figure 5. The occurrence of bits and blocks of zeros and ones of sizes from one to four as described previously, within the random bits sequences produced with the CA random number generator with reaction-diffusion memory.

In essence, statistical testing of random number generators is nothing but a particular kind of Monte Carlo simulation. Conversely, when testing a random number generator for suitability with respect to a particular Monte Carlo problem, running the simulation with a related but simplified model, that is, one where the distribution of the result can be attained theoretically, may serve as a test. Even if the distribution is not known, the results of the designated random number generator can still be compared to the ones produced by a few other "good" generators of quite different designs. Most statistical tests for random number generators utilize the concept of a -value. -values of single tests should not only be in the proper range (not too close to 0 or 1) but should also be uniformly distributed. If the final p-value of a test is really close to 0 or 1, the random number generator is said to fail the test. If the p-value is suspicious, the test is repeated and/or the sample size is increased, and often things will then clarify. Otherwise, the random number generator is said to have passed the test. In order to apply the statistical tests, we generated sequences of random bits with a length of 32 bits using this procedure: the CA of radius $r = 1$, on a square lattice of size $N = 8 \times 8$ without any post-processing (site or time spacing in order to de-correlate the sequences) is run for t time steps, thus generating $N / 2^t$ random temporal bit sequences of length L. Table 1 shows the results of applying the test battery Crush from the random number generator software package TestU01 and the Diehard battery of tests to our novel CA-based model. The results of the proposed model are validated, and we demonstrate that CAs can be used to rapidly produce purely random temporal bit sequences to an arbitrary precision. Furthermore, there is a notable advantage arising from the existence of a "tunable" algorithm for the generation of randomizers.

6.4 Cryptographic Properties

In this paper, we also investigate the potential application of cellular computation for constructing PRNGs. Further, the PRNG scheme based on CAs with reaction-diffusion memory is featured for its capability of generating ultrahigh-quality random numbers. However, the quality bottleneck of a practical PRNG lies in the limited cycle of the generator. To close the gap between the pure randomness generation and the short period, we propose and implement a memory algorithm based on the reaction-diffusion process in a chemical system for Boolean CAs. This scheme is characterized by a tradeoff between, on one hand, the rate of generation of random bits and, on the other hand, the degree of randomness that the series can deliver. This result opens a new perspective to apply CAs as a computational engine for the robust generation of pure random numbers, which has important applications in cryptography and other related areas.

Careful mathematical analysis is required to have any confidence that a PRNG generates numbers that are sufficiently random to suit the intended use. All practical methods for obtaining random numbers are based on deterministic algorithms, which is why such numbers are more appropriately called pseudorandom, as distinguished from true random numbers resulting from some natural physical process. Random number generators must possess a number of properties if they are to be useful in lengthy stochastic simulations such as those used in computational physics. In practice, the output from many common PRNGs exhibits artifacts that cause it to fail statistical pattern-detection tests. CA-based random number generators evolve a state vector of zeros and ones according to a deterministic rule. For a given CA, an element (or cell) at a given position in the new state vector is determined by certain neighboring cells of that cell in the old state vector. A subset of cells in the state vectors is then output as random bits from which the pseudorandom numbers are generated. In the last decade, CAs have been used to generate good random numbers. However, by an extensive suite of statistical tests and the results of experiments, we have shown that our CA algorithm can be applied to the difficult problem of generating random numbers. In order to demonstrate the performance of a proposed random number generator, we used the standard software packages for statistically evaluating the quality of random number sequences known as the Diehard battery test suite [69] and TestU01 [70].

Statistical Test Suite	Number of Statistics	Result
NIST	—	passed
Diehard	126	passed
SmallCrush	15	passed
Crush	144	passed
BigCrush	160	passed

Table 1. Summary result of the statistical test suite Diehard, NIST, testU01 (SmallCrush, Crush, BigCrush) test batteries applied to proposed CA PRNGs on grid size with periodic boundary condition.

7. Results and Discussion

This paper proposed the concept of memory reaction to describe conditional chemical reactions that occur in the path of memory events. The proposed model represents an innovative strategy to use a reduced model to describe nonlinear dynamics. Numerical simulations suggested that memory reactions for realizing species activation/inactivation play a major role in generating complex dynamics. Recalling previous results on the application of cellular automata (CAs), we enriched elementary CAs by introducing a reaction-diffusion memory term compared to the conventional cellular automaton (CA) paradigm that merely takes into account the last configuration. Apart from their potential applications, the dynamics of elementary rules are dramatically altered when endowing cells with memory of the previous time steps. CAs with memory in cells can be considered as a natural and promising extension of the basic paradigm that opens a new field of research in the selection of nontrivial rules of cell-state transitions and precise mechanics of relationships between chaotic and complex systems. Therefore, memory in elementary and other CA families offers a new approach for discovering complex dynamics based on gliders and nontrivial interactions between gliders. This paper also discusses the generation of random sequences by simple procedures that seem to capture many features of this phenomenon. The investigations described may not only suggest practical methods for random sequence generation, but also provide further understanding of the nature and origins of randomness in physical processes. Advantages arise from a reduced cross-correlation between the individual bit streams, and hence, an increased statistical independence in the set of out vectors. The simplicity, no significant limits on the degree of randomness and intrinsic parallelism of the proposed CA make possible efficient implementation on many kinds of computers, and very efficient hardware implementations could also be possible.

These successful applications suggested that the proposed theory is an effective tool to realize conditional chemical reactions in a wide range of complex biological systems. Time delay is a modeling technique to realize slow reactions or simplify multiple small step reactions. It is emphasized that the difference between the delayed reaction and the proposed memory reaction is substantial. First, the firing of delayed reactions depends on the competition with other reactions in the system. However, the occurrence of memory reactions is conditional to the path of memory events, though simultaneously the firing of memory reactions also depends on the competition with other reactions if it is within the memory time period. In addition, the key feature of delayed reaction is the time difference between the firing of a chemical reaction and the manifestation of its products. However, the products of a memory reaction are generated immediately after its firing. In this paper we also proposed the delayed memory reaction if the reaction is conditional to the path of memory events, as well as if there is a delay between the firing of the chemical reaction and the manifestation of its products. Thus, the memory and time delay are two distinct features of chemical reactions, though these two types of reactions are connected to a fixed length of time period.

References

[1] W. S. McCulloch and W. Pitts, “A Logical Calculus of the Ideas Immanent in Nervous Activity,” *The Bulletin of Mathematical Biophysics*, **5**(4), 1943 pp. 115–133. doi:10.1007/BF02478259.

[2] S. C. Kleene, *Representation of Events in Nerve Nets and Finite Automata*, Santa Monica, CA: RAND Corporation, 1951. www.rand.org/pubs/research_memoranda/RM704.html.

[3] M. L. Minsky, *Computation: Finite and Infinite Machines*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1967.

[4] J. van Leeuwen and J. Wiedermann, “Beyond the Turing Limit: Evolving Interactive Systems,” *SOFSEM 2001: Theory and Practice of Informatics*, (L. Pacholski and P. Ružička, eds.), Berlin, Heidelberg: Springer, 2001 pp. 90–109. doi:10.1007/3-540-45627-9_8.

[5] J. Wiedermann and J. van Leeuwen, “How We Think of Computing Today,” *Logic and Theory of Algorithms, CiE 2008* (A. Beckmann, C. Dimitracopoulos and B. Löwe, eds.), Heidelberg: Springer, 2008 pp. 579–593. doi:10.1007/978-3-540-69407-6_61.

[6] A. Adamatzky and B. de Lacy Costello, “Experimental Logical Gates in a Reaction-Diffusion Medium: The XOR Gate and Beyond,” *Physical Review E*, **66**(4), 2002 046112. doi:10.1103/PhysRevE.66.046112.

[7] N. Rambidi, “Roots and Promises of Chemical-Based Computing,” *BioSystems*, **64**(1–3), 2002 pp. 169–178. doi:10.1016/S0303-2647(01)00184-8.

[8] A. Tóth and K. Showalter, “Logic Gates in Excitable Media,” *The Journal of Chemical Physics*, **103**(6), 1995 pp. 2058–2066. doi:10.1063/1.469732.

[9] I. Motoike and K. Yoshikawa, “Information Operations with an Excitable Field,” *Physical Review E*, **59**(5), 1999 5354–5360. doi:10.1103/PhysRevE.59.5354.

[10] J. Sielewiesiuk and J. Górecki, “Complex Transformations of Chemical Signals Passing through a Passive Barrier,” *Physical Review E*, **66**(1), 2002 016212. doi:10.1103/PhysRevE.66.016212.

[11] J. Sielewiesiuk and J. Górecki, “Passive Barrier as a Transformer of ‘Chemical Signal’ Frequency,” *The Journal of Physical Chemistry A*, **106**(16), 2002 pp. 4068–4076. doi:10.1021/jp013844t.

[12] J. Gorecki and J. Gorecka, “Chemical Programming in Reaction-Diffusion Systems,” *Unconventional Computing 2005: From Cellular Automata to Wetware* (C. Teuscher and A. Adamatzky, eds.), Beckington, UK: Luniver Press, 2005. uncomp.uwe.ac.uk/free-books/uc-2005-free.pdf.

[13] S. Wolfram, *A New Kind of Science*, Champaign, IL: Wolfram Media, Inc., 2002.

[14] S. Wolfram (ed.), *Theory and Applications of Cellular Automata: Including Selected Papers, 1983–1986*, Singapore: World Scientific, 1986.

[15] S. Wolfram, "Statistical Mechanics of Cellular Automata," *Reviews of Modern Physics*, **55**(3), 1983 pp. 601–644. doi:10.1103/RevModPhys.55.601.

[16] S. Wolfram, "Universality and Complexity in Cellular Automata," *Physica D: Nonlinear Phenomena*, **10**(1–2), 1984 pp. 1–35. doi:10.1016/0167-2789(84)90245-8.

[17] F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," *Psychological Review*, **65**(6), 1958 pp. 386–408. doi:10.1037/h0042519.

[18] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, **323**(6088), 1986 pp. 533–536. doi:10.1038/323533a0.

[19] T. Kohonen, *Self-Organization and Associative Memory*, Berlin, Heidelberg: Springer-Verlag, 1989.

[20] G. Costantini, D. Casali and R. Perfetti, "Neural Associative Memory Storing Gray-Coded Gray-Scale Images," *IEEE Transactions on Neural Networks*, **14**(3), 2003 pp. 703–707. doi:10.1109/TNN.2003.810596.

[21] G. Costantini, D. Casali and R. Perfetti, "Associative Memory Design for 256 Gray-Level Images Using a Multilayer Neural Network," *IEEE Transactions on Neural Networks*, **17**(2), 2006 pp. 519–522. doi:10.1109/TNN.2005.863465.

[22] G. E. Hinton and J. A. Anderson (eds.), *Parallel Models of Associative Memory: Updated Edition*, New York: Psychology Press, 2014.

[23] G. Costantini, D. Casali and M. Todisco, "A Hardware-Implementable System for Retinal Vessel Segmentation," *Latest Trends on Computers, Vol. II* (N. Mastorakis, V. Mladenov and Z. Bojkovic, eds.), Cambridge, UK: WSEAS Press, 2010.

[24] G. Saggio, F. Giannini, M. Todisco and G. Costantini, "A Data Glove Based Sensor Interface to Expressively Control Musical Processes," in *4th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)*, Savelletri di Fasano, Italy, Piscataway, NJ: IEEE, 2011, pp. 192–195. doi:10.1109/IWASI.2011.6004715.

[25] J. J. Hopfield and D. W. Tank, "'Neural' Computation of Decisions in Optimization Problems," *Biological Cybernetics*, **52**(3), 1985 pp. 141–152. link.springer.com/article/10.1007/BF00339943.

[26] G. Costantini, G. Saggio and M. Todisco, "A Glove Based Adaptive Sensor Interface for Live Musical Performances," *First International Conference on Sensor Device Technologies and Applications*, Venice, Italy, July 18–25, 2010, Los Alamitos, CA: IEEE Computer Society, 2010 pp. 217–220. doi:10.1109/SENSORDEVICES.2010.47.

[27] G. Costantini, G. Saggio, L. Sbernini, N. Di Lorenzo, F. Di Paolo and D. Casali, "Surgical Skill Evaluation by Means of a Sensory Glove and a Neural Network," in *Proceedings of the International Conference on Neural Computation Theory and Applications - Vol. 1: NCTA*, Roma, 2014 (K. Madani and J. Filipe, eds.), SciTePres, 2014 pp. 105–110. doi:10.5220/0005030301050110.

[28] G. E. Hinton and T. J. Sejnowski (eds.), *Unsupervised Learning: Foundations of Neural Computation*, Cambridge, MA: MIT Press, 1999.

[29] G. Costantini, R. Perfetti and M. Todisco, "Recurrent Neural Network for Approximate Nonnegative Matrix Factorization," *Neurocomputing*, **138**, 2014 pp. 238–247. doi:10.1016/j.neucom.2014.02.007.

[30] W. Maass, "Networks of Spiking Neurons: The Third Generation of Neural Network Models," *Neural Networks*, **10**(9), 1997 pp. 1659–1671. doi:10.1016/S0893-6080(97)00011-7.

[31] F. Bagnoli, R. Rechtman and S. Ruffo, "Damage Spreading and Lyapunov Exponents in Cellular Automata," *Physics Letters A*, **172**(1–2), 1992 pp. 34–38. doi:10.1016/0375-9601(92)90185-O.

[32] E. Farhi and S. Gutmann, "Quantum Computation and Decision Trees," *Physical Review A*, **58**(2), 1998 pp. 915–928. doi:10.1103/PhysRevA.58.915.

[33] J. D. Whitfield, C. A. Rodríguez-Rosario and A. Aspuru-Guzik, "Quantum Stochastic Walks: A Generalization of Classical Random Walks and Quantum Walks," *Physical Review A*, **81**(2), 2010 022323. doi:10.1103/PhysRevA.81.022323.

[34] S. Wolfram, "Random Sequence Generation by Cellular Automata," *Advances in Applied Mathematics*, **7**(2), 1986 pp. 123–169. doi:10.1016/0196-8858(86)90028-X.

[35] Z. Zarezadeh, "Cellular Automaton-Based Pseudorandom Number Generator," *Complex Systems*, **26**(4), 2017 pp. 373–389. doi:10.25088/ComplexSystems.26.4.373.

[36] M. Tomassini, M. Sipper and M. Perrenoud, "On the Generation of High-Quality Random Numbers by Two-Dimensional Cellular Automata," *IEEE Transactions on Computers*, **49**(10), 2000 pp. 1146–1151. doi:10.1109/12.888056.

[37] M. Sipper and M. Tomassini, "Generating Parallel Random Number Generators by Cellular Programming," *International Journal of Modern Physics C*, **7**(2), 1996 pp. 181–190. doi:10.1142/S012918319600017X.

[38] M. Tomassini, M. Sipper, M. Zolla and M. Perrenoud, "Generating High-Quality Random Numbers in Parallel by Cellular Automata," *Future Generation Computer Systems*, **16**(2–3), 1999 pp. 291–305. doi:10.1016/S0167-739X(99)00053-9.

[39] M. Sipper and M. Tomassini, "Co-evolving Parallel Random Number Generators," in *International Conference on Parallel Problem Solving from Nature (PPSN 1996)* (H. M. Voigt, W. Ebeling, I. Rechenberg and H. P. Schwefel, eds.), Berlin, Heidelberg: Springer, 1996 pp. 950–959. doi:10.1007/3-540-61723-X_ 1058.

[40] S.-U. Guan and S. K. Tan, "Pseudorandom Number Generation with Self-Programmable Cellular Automata," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **23**(7), 2004 pp. 1095–1101. doi:10.1109/TCAD.2004.829808.

[41] P. Tsalides, T. A. York and A. Thanailakis, "Pseudorandom Number Generators for VLSI Systems Based on Linear Cellular Automata," *IEEE Proceedings E—Computers and Digital Techniques*, **138**(4), 1991 pp. 241–249. ieeexplore.ieee.org/document/81902.

[42] R. Alonso-Sanz and L. Bull, "Random Number Generation by Cellular Automata with Memory," *International Journal of Modern Physics C*, **19**(2), 2008 pp. 351–367. doi:10.1142/S012918310801211X.

[43] R. Alonso-Sanz and L. Bull, "Elementary Cellular Automata with Minimal Memory and Random Number Generation," *Complex Systems*, **18**(2), 2009 pp. 195–213.
wpmedia.wolfram.com/uploads/sites/13/2019/03/18-2-2.pdf.

[44] B. Shackleford, M. Tanaka, R. J. Carter and G. Snider, "FPGA Implementation of Neighborhood-of-Four Cellular Automata Random Number Generators," in *Proceedings of the 2002 ACM/SIGDA Tenth International Symposium on Field-Programmable Gate Arrays*, Monterey, CA, 2002, New York: ACM, 2002 pp. 106–112. doi:10.1145/503048.503064.

[45] B. Shackleford, M. Tanaka, R. J. Carter and G. Snider, "High-Performance Cellular Automata Random Number Generators for Embedded Probabilistic Computing Systems," in *Proceedings 2002 NASA/DoD Conference on Evolvable Hardware*, Alexandria, VA, 2002, (A. Stoica, J. Lohn, R. Katz, D. Keymeulen and R. S. Zebulum, eds.), Los Alamitos, CA: IEEE, 2002 pp. 191–200. doi:10.1109/EH.2002.1029885.

[46] S.-U. Guan and S. Zhang, "An Evolutionary Approach to the Design of Controllable Cellular Automata Structure for Random Number Generation," *IEEE Transactions on Evolutionary Computation*, **7**(1), 2003 pp. 23–36. doi:10.1109/TEVC.2002.806856.

[47] P. H. Bardell, "Analysis of Cellular Automata Used as Pseudorandom Pattern Generators," in *Proceedings International Test Conference 1990*, Washington, DC, 1990, Piscataway, NJ: IEEE, 1990 pp. 762–768. doi:10.1109/TEST.1990.114093.

[48] L. Kotoulas, D. Tsarouchis, G. C. Sirakoulis and I. Andreadis, "1-D Cellular Automaton for Pseudorandom Number Generation and Its Reconfigurable Hardware Implementation," in *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS 2006)*, Island of Kos, Greece, 2006, Piscataway, NJ: IEEE, 2006 p. 4. doi:10.1109/ISCAS.2006.1693661.

[49] A. Compagner and A. Hoogland, "Maximum-Length Sequences, Cellular Automata, and Random Numbers," *Journal of Computational Physics*, **71**(2), 1987 pp. 391–428. doi:10.1016/0021-9991(87)90037-4.

[50] R. Dogaru, "Hybrid Cellular Automata as Pseudo-random Number Generators with Binary Synchronization Property," in *International Symposium on Signals, Circuits and Systems (ISSCS 2009)*, Iasi, Romania, 2009, Piscataway, NJ: IEEE, 2009 pp. 1–4. doi:10.1109/ISSCS.2009.5206126.

[51] G. Cauwenberghs, "Delta-Sigma Cellular Automata for Analog VLSI Random Vector Generation," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, **46**(3), 1999 pp. 240–250. doi:10.1109/82.754858.

[52] J. M. Comer, J. C. Cerda, C. D. Martinez and D. H. Hoe, "Random Number Generators Using Cellular Automata Implemented on FPGAs," in *Proceedings of the 2012 44th Southeastern Symposium on System Theory (SSST)*, Jacksonville, FL, 2012, Piscataway, NJ: IEEE, 2012 pp. 67–72. doi:10.1109/SSST.2012.6195137.

[53] W. Meier and O. Staffelbach, "Analysis of Pseudo Random Sequences Generated by Cellular Automata," in *Advances in Cryptology (EUROCRYPT '91)* (D. W. Davies, ed.), Berlin, Heidelberg: Springer, 1991 pp. 186–199. doi:10.1007/3-540-46416-6_17.

[54] P. D. Hortensius, R. D. McLeod and H. C. Card, "Parallel Random Number Generation for VLSI Systems Using Cellular Automata," *IEEE Transactions on Computers*, **38**(10), 1989 pp. 1466–1473. doi:10.1109/12.35843.

[55] P. D. Hortensius, R. D. McLeod, W. Pries, D. M. Miller and H. C. Card, "Cellular Automata-Based Pseudorandom Number Generators for Built-In Self-Test," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **8**(8), 1989 pp. 842–859. doi:10.1109/43.31545.

[56] M. Tomassini and M. Perrenoud, "Stream Cyphers with One- and Two-Dimensional Cellular Automata," in *International Conference on Parallel Problem Solving from Nature (PPSN 2000)* (M. Schoenauer, et al., eds.), Berlin, Heidelberg: Springer, 2000 pp. 722–731. doi:10.1007/3-540-45356-3_ 71.

[57] S. Wolfram, "Cryptography with Cellular Automata," in *Conference on the Theory and Application of Cryptographic Techniques (CRYPTO '85)*, Berlin, Heidelberg: Springer, 1985 pp. 429–432. doi:10.1007/3-540-39799-X_32.

[58] M. Tomassini and M. Perrenoud, “Cryptography with Cellular Automata,” *Applied Soft Computing*, **1**(2), 2001 pp. 151–160. doi:10.1016/S1568-4946(01)00015-1.

[59] F. Seredynski, P. Bouvry and A. Y. Zomaya, “Cellular Automata Computations and Secret Key Cryptography,” *Parallel Computing*, **30**(5–6), 2004 pp. 753–766. doi:10.1016/j.parco.2003.12.014.

[60] A. Jaberi, R. Ayanzadeh and A. S. Z. Mousavi, “Two-Layer Cellular Automata Based Cryptography,” *Trends in Applied Sciences Research*, **7**(1), 2012 pp. 68–77. doi:10.3923/tasr.2012.68.77.

[61] S. Sen, C. Shaw, D. R. Chowdhuri, N. Ganguly and P. Pal Chaudhuri, “Cellular Automata Based Cryptosystem (CAC),” in *International Conference on Information and Communications Security (ICICS 2002)* (R. Deng, F. Bao, J. Zhou and S. Qing, eds.), Berlin, Heidelberg: Springer, 2002 pp. 303–314. doi:10.1007/3-540-36159-6_26.

[62] G. Á. Marañón, L. H. Encinas, A. H. Encinas, Á. M. del Rey and G. R. Sánchez, “Graphic Cryptography with Pseudorandom Bit Generators and Cellular Automata,” in *Knowledge-Based and Intelligent Information and Engineering Systems (KES 2003)* (V. Palade, R. J. Howlett and L. Jain, eds.), Berlin, Heidelberg: Springer, 2003 pp. 1207–1214. doi:10.1007/978-3-540-45224-9_163.

[63] P. Guam, “Cellular Automaton Public Key Cryptosystems,” *Complex Systems*, **1**(1), 1987 pp. 51–56. complex-systems.com/pdf/01-1-4.pdf.

[64] H. Gutowitz, “Cryptography with Dynamical Systems,” in *Cellular Automata and Cooperative Systems* (N. Boccara, E. Goles, S. Martinez and P. Picco, eds.), Dordrecht: Springer, 1993 pp. 237–274. doi:10.1007/978-94-011-1691-6_21.

[65] S. Nandi, B. K. Kar and P. Pal Chaudhuri, “Theory and Applications of Cellular Automata in Cryptography,” *IEEE Transactions on Computers*, **43**(12), 1994 pp. 1346–1357. doi:10.1109/12.338094.

[66] P. Pal Chaudhuri, D. R. Chowdhury, S. Nandi and S. Chattopadhyay, *Additive Cellular Automata: Theory and Applications*, Los Alamitos, CA: IEEE Computer Society Press, 1997.

[67] W. D. Hillis, *The Connection Machine*, Cambridge, MA: MIT Press, 1985.

[68] Mathematica, Release Version 11.3, Champaign: Wolfram Research, Inc., 2018.

[69] G. Marsaglia. “The Marsaglia Random Number CDROM Including the Diehard Battery of Tests of Randomness.” (Jul 5, 2019) ftpmirror.your.org/pub/misc/diehard.

[70] P. L’Ecuyer and R. Simard, “TestU01: A C Library for Empirical Testing of Random Number Generators,” *ACM Transactions on Mathematical Software*, **33**(4), 2007 Art. no. 22. doi:10.1145/1268776.1268777.