On the decomposition of stochastic cellular automata

Witold Bolt\textsuperscript{a,b,*}, Jan M. Baetens\textsuperscript{b}, Bernard De Baets\textsuperscript{b}

\textsuperscript{a}Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland
\textsuperscript{b}KERMIT, Department of Mathematical Modelling, Statistics and Bioinformatics, Ghent University, Coupure links 653, B-9000 Gent, Belgium

Abstract

In this paper we present two interesting properties of stochastic cellular automata that can be helpful in analyzing the dynamical behavior of such automata. The first property allows calculating cell-wise probability distributions over the state set of a stochastic cellular automaton, i.e. images that show the average state of each cell during the stochastic cellular automaton evolution. The second property shows that stochastic cellular automata are equivalent to so-called stochastic mixtures of deterministic cellular automata. By means of this property, any stochastic cellular automaton can be decomposed into a set of deterministic systems, each of which contributes to the behavior of the stochastic system.

Keywords: stochastic cellular automata, complexity analysis, continuous cellular automata, decomposition

1. Introduction

Cellular Automata (CAs) are often used for constructing models in a variety of fields of application including chemistry, biology, medicine, physics, ecology and the study of socioeconomic interactions. In many of these settings, Stochastic CAs (SCAs) are considered due to the stochastic nature of the phenomenon being modeled or the lack of understanding of the exact rules driving the phenomenon [1, 2]. Therefore, a better understanding of the dynamics of SCAs is crucial. Only a few methods for dealing with the analysis of models based on SCAs have been developed. In the case of many practical applications, sampling methods, relying on repetitive computer simulations are sufficient [3, 4, 5], especially in cases when the averaged behavior of the system is concerned. Techniques built on the mean–field theory can be used to study correlations and long-term behavior of SCAs [6]. The theory of Markov chains can be applied to provide analytical tools for analyzing the model’s behavior [7], although in

\*Corresponding author

Email address: witold.bolt@hope.art.pl (Witold Bolt)
practice, due to the theoretical and computational complexity of such methods, the application scope is limited.

This paper is devoted to providing effective analytical tools for the analysis of multi-state SCAs, which are based on deterministic CAs. Although the theoretical foundation of the presented results is already (at least partially) known in the literature \[7, 8, 9\], so far the applications are limited. Therefore, the main aim of this paper is to provide a complete, formal description of the discussed properties in a form that is suitable for applications and does not require a strong mathematical background, as well as to present examples that can motivate further applications in the domain of systems modeling.

The methods presented here are developed in the context of 1D SCAs on finite lattices, but can be easily generalized to the case of higher dimensions and infinite spaces.

Two main results are presented in this paper. The first one involves constructing images that show the cell-wise probability distribution over the state set, at any time step. The method is based on associating an SCA with a deterministic, Continuous CA (CCA). The second result shows the equivalence of SCAs and stochastic mixtures of deterministic CAs. Based on this finding, any SCA can be decomposed into a finite set of deterministic CAs, each of them contributing to the behavior of the stochastic system. An effective method for finding a decomposition is presented. It allows to uncover the deterministic component in the mixture with the highest impact on the behavior of the SCA.

This paper is organized as follows. We start with some preliminaries and definitions in Section 2. In Section 3, we introduce the concept of CCAs and the formalism enabling the analysis of multi-state CAs. Section 4 contains the definition of multi-state SCAs and holds the main results of this paper. The paper is concluded with Section 5 discussing the experimental results that illustrate both properties. A summary is presented in Section 6.

2. Preliminaries

Informally, a CA is a discrete dynamical system in which the space is subdivided into discrete elements, referred to as cells. At every consecutive, discrete time step, each cell is assigned one of \( N > 1 \) states using a deterministic rule, which depends only on the previous state of the considered cell, and the states of its neighboring cells \[10\].

Formally speaking, let the state set \( S \) be a finite set of \( N > 1 \) elements. Elements of the set \( \mathcal{C} = \{ c_i \mid i = 1, \ldots, M \} \) will be referred to as cells. Every cell \( c_i \) is assigned a state \( s(c_i, t) \in S \) at each time step \( t \in \mathbb{N} \), according to a local, deterministic rule. The vector \( s(\cdot, t) \in S^M \) will be referred to as the configuration, and as the initial configuration when \( t = 0 \). The sequence \( (s(\cdot, 0), s(\cdot, 1), \ldots) \) will be referred to as the space-time diagram of the CA. For technical reasons, we impose periodic boundary conditions, but our results do not depend on this assumption.
The mapping \( A : \mathcal{S}^M \to \mathcal{S}^M \) satisfying:
\[
s(\cdot, t + 1) = A(s(\cdot, t)),
\]
(for every \( t \in \mathbb{N} \)) will be referred to as a global CA rule or simply a CA, if there exists a radius \( r \in \mathbb{N} \) and a function \( f : \mathcal{S}^{2r+1} \to \mathcal{S} \) satisfying:
\[
s(c, t + 1) = f(s(c_{-r}, t), \ldots, s(c_{r}, t)),
\]
for every \( i \) and \( t \in \mathbb{N} \). Such a function \( f \) will be referred to as a local rule. Note that a local rule uniquely defines the global rule, while for a given global rule, multiple local rules exist. Additionally, it is assumed that \( r \ll M \). The sequence \( (c_{i-r}, \ldots, c_{i+r}) \) will be referred to as the neighborhood of cell \( c_i \) and \( R = 2r + 1 \) will denote the neighborhood size. For the sake of simplicity, \( s(c_{i-r}, \ldots, c_{i+r}, t) \) will denote the state of the neighborhood of cell \( c_i \) at time step \( t \), and will be referred to as the neighborhood configuration at time step \( t \).

3. Continuous CAs

There exist multiple ways of extending the definition of CAs to cover infinite state sets. Examples of such approaches include Coupled Map Lattices (CMLs) \[11\] and so-called fuzzy CAs \[12, 13, 14\]. In this section, we present Continuous CAs (CCAs), which can be seen as a generalization of the ideas presented in \[12\]. Our formalism is based on a polynomial representation of discrete CA rules. We start with formulating the continuous counterparts of binary CAs. After that we present a generalization to cover multi-state CAs.

3.1. Binary CAs

Binary CAs are widely studied \[15,16\], because they allow to evolve complex patterns and exhibit complex behavior despite their intrinsic simplicity. The state set of such a CA \( A \) is \( \mathcal{S} = \{0, 1\} \) and its \( f : \mathcal{S}^R \to \mathcal{S} \) is uniquely defined by the following system of equalities:
\[
f(s_1, \ldots, s_R) = l_i,
\]
for all possible binary sequences \( (s_1, \ldots, s_R) \), where \( i \) is defined as \( i = 1 + \sum_{j=0}^{R-1} s_{R-j} 2^j \) such that \( i \in \{1, 2, \ldots, 2^R\} \). Following Eq. (2), variables \( s_i \) correspond to the states constituting the neighborhood configuration. The binary vector \( (l_i)_{i=1}^{2^R} \) will be referred to as the lookup table (LUT) of the local rule \( f \). Following \[12\], we know that the function \( f \) can be expressed as the following polynomial:
\[
f(s_1, \ldots, s_R) = \sum_{i=1}^{2^R} l_i \left( \prod_{m=1}^{R} n(s_m, i) \right),
\]
where \( n(s_m, i) \) is defined as:

\[
\begin{align*}
n(s_m, i) &= \begin{cases} 
s_m & \text{if } \text{ind}(i)[m] = 2, \\
1 - s_m & \text{if } \text{ind}(i)[m] = 1. 
\end{cases}
\end{align*}
\]

The function \( \text{ind} \) is related to the binary representation of integers. In [12] a simpler formulation using the function \( \text{bin} \) is used. The construction presented here, although a bit more complicated in the binary setting, allows for an easier generalization to the multi-state case. It is defined in such a way that \( \text{ind}(i)[m] \) is the \( m \)-th digit, incremented by one, read from left to right, of the binary representation of the integer \( i - 1 \), padded with ones on the left, so that it always has length \( R \). Consequently, it holds that:

\[
i = 1 + \sum_{m=1}^{R} (\text{ind}(i)[R - (m - 1)] - 1) 2^{m-1}.
\] (5)

The values of \( \text{ind}(i) \) for \( R = 3 \) and \( i \in \{1, \ldots, 8\} \) are shown below:

\[
\begin{align*}
\text{ind}(1) &= (1, 1, 1), & \text{ind}(2) &= (1, 1, 2), & \text{ind}(3) &= (1, 2, 1), & \text{ind}(4) &= (1, 2, 2), \\
\text{ind}(5) &= (2, 1, 1), & \text{ind}(6) &= (2, 1, 2), & \text{ind}(7) &= (2, 2, 1), & \text{ind}(8) &= (2, 2, 2).
\end{align*}
\]

**Example 1 (Elementary CAs).** Binary, 1D CAs, with neighborhood radius \( r = 1 \), are commonly referred to as Elementary CAs (ECAs) [15]. There are 256 such ECAs. Treating the LUT entries \( l_i \) as digits of an integer written in base of 2, we can enumerate the local rules of ECAs. By convention, the binary vectors are read in the reverse order, i.e. \( (l_i)_{i=8}^1 \) is interpreted as \( (l_8, l_7, \ldots, l_1) \). For example, given LUT \( l = (0, 1, 1, 0, 1, 0, 0, 1) \) of ECA 150, and defining the Boolean complement as \( \bar{s} = 1 - s \), its local rule can be written, according to Eq. (4) as:

\[
f_{150}(s_1, s_2, s_3) = \bar{s}_1 \bar{s}_2 s_3 + s_1 s_2 \bar{s}_3 + s_1 \bar{s}_2 s_3 + s_1 s_2 s_3.
\]

Using this notation, a CCA can be defined analogously to a binary CA, with two notable differences. Firstly, the state set of a CCA is the unit interval, i.e. \( S = [0, 1] \), and, secondly, the local rule \( f: [0, 1]^R \rightarrow [0, 1] \) is given by Eq. (4) with coefficients \( l_i \in [0, 1] \). We will refer to such a sequence \( (l_i)_{i=1}^R \) as a generalized LUT. It is easy to check that this definition of a CCA is formally correct. Indeed, the values of the function \( f \) in Eq. (4) are guaranteed to belong to the unit interval if \( l_i \in [0, 1] \) for all \( i \) and \( s_m \in [0, 1] \) for all \( m \in \{1, \ldots, R\} \). Note that this construction is directly related to the one presented in [12], where fuzzy CAs are constructed as polynomials representing fuzzified logical functions. Following the same line of reasoning, in [17] an alternative polynomial representation for the local rules of binary CAs is presented, which is consistent with a logical representation of the local rules.

In order to introduce the formalism that is needed in the multi-state setting, we present a slightly modified way of representing binary CAs compared to the
one obtained through Eq. (4). Let us assume that the state set is given by $S_2 = \{(1,0), (0,1)\} \subset \mathbb{R}^2$. Then the local rule is a function $f: S_2^R \to S_2$ and can be represented as a vector function $f = (f_1, f_2)$, where $f_j: S_2^R \to \{0,1\}$. Note that any $s = (s_1, s_2) \in S_2$ satisfies $s_2 = 1 - s_1$. Similarly, $f_2 = 1 - f_1$, so that the local rule $f$ can be defined using $f_2$ only. We define $f_2$ with a formula similar to Eq. (4):

$$f_2(s_1, \ldots, s_R) = \sum_{i=1}^{2^R} l_{i,2} \left( \prod_{m=1}^{R} n(s_{m,2}, i) \right),$$

(6)

where for every $i = 1, \ldots, 2^R$ it holds that $l_i = (l_{i,1}, l_{i,2}) \in S_2$ and for every $m \in \{1, \ldots, R\}$, $s_m = (s_{m,1}, s_{m,2}) \in S_2$. Since $s_{m,1} = 1 - s_{m,2}$, we can rewrite the formula for $n$ as:

$$n(s_{m,2}, i) = \begin{cases} 
    s_{m,2}, & \text{if } \text{ind}(i)[m] = 2, \\
    s_{m,1}, & \text{if } \text{ind}(i)[m] = 1,
\end{cases}$$

(7)

which can simplified to:

$$n(s_{m,2}, i) = s_{m, \text{ind}(i)[m]},$$

and thus $f_2$ becomes:

$$f_2(s_1, \ldots, s_R) = \sum_{i=1}^{2^R} l_{i,2} \left( \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} \right).$$

(8)

Having defined the function $f_2$, we can prove the following fact, which gives a direct formula for $f_1$.

**Lemma 1.** Function $f_1$ can be expressed as:

$$f_1(s_1, \ldots, s_R) = \sum_{i=1}^{2^R} l_{i,1} \left( \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} \right).$$

(9)

Due to the above, we can write a general expression for $f_j$, $j = 1, 2$, as:

$$f_j(s_1, \ldots, s_R) = \sum_{i=1}^{2^R} l_{i,j} \left( \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} \right).$$

(10)

Resorting to this vector notation, a CCA can be defined with a state set:

$$S_c = \{(x_1, x_2) \in [0,1]^2 \mid x_1 + x_2 = 1\} \subset \mathbb{R}^2,$$

(11)

and a local rule $f: S_c^R \to S_c$ given by Eq. (10) with $(l_{j})_{j=1}^{2^R} \in S_c^{2^R}$.
Example 2. The local rule $f_{150}$ of ECA 150 presented in Example 1 can be rewritten in the context of the state set $S_2$, as $f: S_2^3 \rightarrow S_2$, $f = (f_1, f_2)$. Note that for $s_i \in S_2$ it holds that $s_i = (s_{i,1}, s_{i,2})$ for $i = 1, 2, 3$. The local rule $f$ can be written as:

\[
\begin{align*}
    f_1(s_1, s_2, s_3) &= s_{1,2} s_{2,3} + s_{1,2} s_{2,1} s_{3,2} + s_{1,1} s_{2,2} s_{3,1} + s_{1,1} s_{2,1} s_{3,1} \\
    f_2(s_1, s_2, s_3) &= s_{1,2} s_{2,3} + s_{1,2} s_{2,1} s_{3,1} + s_{1,1} s_{2,2} s_{3,1} + s_{1,1} s_{2,1} s_{3,2}.
\end{align*}
\]

The representation given by Eq. (10) is equivalent to that given by Eq. (4), but allows for an easier generalization to multi-state CAs. Therefore, we will use it throughout the remainder of this paper.

3.2. Multi-state CAs

We start with generalizing the state set $S_2$ defined in Section 3.1 to the case of $N$ states. Let $S_N \subset \mathbb{R}^N$ be the set of all base vectors of the $N$-dimensional Euclidean space, i.e.

\[
S_N = \left\{ e \in \{0, 1\}^N \mid \|e\| = 1 \right\}.
\]  

Let the $i$-th state in a multi-state CA be represented by the base vector $e_i$ which has zeros everywhere, except at position $i$, where there is a one. For instance, if $N = 3$, then $S_3 = \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$. For the sake of simplicity, in the remainder of this paper, we will write $S$ instead of $S_N$, assuming that $N$ is fixed.

We now generalize the function $\text{ind}: \{1, \ldots, N^R\} \rightarrow \{1, \ldots, N^R\}$, defined previously for the binary case, to enumerate each of the $N^R$ neighborhood configurations. Let $i \in \{1, \ldots, N^R\}$ be the index identifying a neighborhood configuration. We assume that such a configuration is formed of the following states:

\[
(e_{\text{ind}(i)[1]}, e_{\text{ind}(i)[2]}, \ldots, e_{\text{ind}(i)[R]}).
\]  

The ordering of cells in neighborhoods can be chosen arbitrarily, but we assume that $\text{ind}$ satisfies the following equation:

\[
i = 1 + \sum_{m=1}^{R} \left( \text{ind}(i)[R - (m - 1)] - 1 \right) N^{m-1}.
\]  

The direct formula for calculating $\text{ind}$ is given by:

\[
\text{ind}(i)[m] = \left\lfloor \frac{(i - 1) \mod N^{R-(m-1)}}{N^{R-m}} \right\rfloor + 1.
\]  

Essentially, $\text{ind}(i)$ yields the positional representation of $i - 1$ in base $N$, padded with zeros on the left, so that it always has length $R$, where each of the
digits is incremented by one. Hence, one finds easily:

\[ \text{ind}(1) = (1, \ldots, 1, 1) \]
\[ \text{ind}(2) = (1, \ldots, 1, 2) \]
\[ \vdots \]
\[ \text{ind}(N) = (1, \ldots, 1, N) \]
\[ \text{ind}(N + 1) = (1, \ldots, 2, 1) \]
\[ \vdots \]
\[ \text{ind}(N^R) = (N, \ldots, N, N) \]

Any local rule \( f \) of an \( N \)-state, deterministic CA can be uniquely defined by writing down its outputs for all of the possible neighborhood configurations. Formally, let us consider the system of equations:

\[ f(e_{\text{ind}(i)[1]}, \ldots, e_{\text{ind}(i)[R]}) = l_i, \quad (16) \]

where \( i \in \{1, \ldots, N^R\} \) and \( l_i \in S \). Similarly to the binary case, \( L = (l_i)^{N^R}_{i=1} \in S^{N^R} \) will be called the LUT of a multi-state CA. Basically, \( L \) is a matrix of \( N^R \) columns and \( N \) rows, containing only 0s and 1s, in such a way that every column contains exactly one non-zero entry. Each of the matrices \( L \in S^{N^R} \) uniquely defines an \( N \)-state CA in terms of its local rule.

In order to define a CCA in the context of multiple states, we need to formally define its state set. Let \( S_c \) be defined as:

\[ S_c = \left\{ (x_1, \ldots, x_N) \in [0, 1]^N \mid \sum_{i=1}^N x_i = 1 \right\} \subset \mathbb{R}^N, \quad (17) \]

which is consistent with the definition for \( N = 2 \) given by Eq. (11). Note that any \( s \in S_c \) satisfies \( s_i = 1 - \sum_{j \neq i} s_j \). The set \( S_c \) is commonly referred to as a standard \((N - 1)\)-simplex [18].

**Example 3.** Let \( N = 3 \). The set \( S_c \) is a 2D triangle placed in the 3D Euclidean space, with vertices: \((1, 0, 0), (0, 1, 0)\) and \((0, 0, 1)\). \( \square \)

In the case of a CCA, the local rule is a vector function \( f : S_c^R \rightarrow S_c \), given by \( f = (f_1, \ldots, f_N) \), where for \( j \in \{1, \ldots, N\} \) it holds that \( f_j : S_c^R \rightarrow [0, 1] \) and:

\[ f_j(s_1, \ldots, s_R) = \sum_{i=1}^{N^R} P_{ij} \left( \prod_{m=1}^R s_{m, \text{ind}(i)[m]} \right), \quad (18) \]

where \( P \in S_c^{N^R} \). Note that \( s_m \) is an element of \( S_c \subset [0, 1]^N \) for \( m = 1, \ldots, R \), such that \( s_{m,q} \in [0, 1] \) for \( q = 1, \ldots, N \). We will refer to \( P \) as the generalized LUT of a CCA.
We will show that the definition given by Eq. (18) is formally correct, meaning that $f$ defined with components $f_j$ satisfies $f: \mathcal{S}_c^R \to \mathcal{S}_c$, which is equivalent to showing that $f_j(s_1, \ldots, s_R) \in [0,1]$ and $\sum_{j=1}^{N} f_j(s_1, \ldots, s_R) = 1$ for any $(s_1, \ldots, s_R) \in \mathcal{S}_c^R$.

Lemma 2. For any $(s_1, \ldots, s_R) \in \mathcal{S}_c^R$, it holds that

$$\sum_{i=1}^{N} R \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} = 1. \quad (19)$$

Proof. Let

$$LHS = \sum_{i=1}^{N} R \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]}.$$

Note that for every $j \in \{1, \ldots, N\}^R$, there exists exactly one $i \in \{1, \ldots, N\}$ such that $j = \text{ind}(i)$. Due to this, it holds that:

$$LHS = \sum_{j \in \{1, \ldots, N\}^R} R \prod_{m=1}^{R} s_{m,j_m}.$$

By regrouping the elements of the sum $S$ we may write:

$$LHS = \sum_{j \in \{1, \ldots, N\}^R} R \prod_{m=1}^{R} s_{m,j_m} \left( \sum_{i=1}^{N} s_{R,i} \right).$$

Since $s_R \in \mathcal{S}_c$, we know that $\sum_{i=1}^{N} s_{R,i} = 1$, and thus:

$$LHS = \sum_{j \in \{1, \ldots, N\}^R} R \prod_{m=1}^{R-1} s_{m,j_m}.$$

By repeating this regrouping procedure $R - 1$ times, we finally get:

$$LHS = \sum_{j \in \{1, \ldots, N\}^R} s_{1,j} = 1. \quad \square$$

Proposition 1. For any $j \in \{1, \ldots, N\}$, and any $s_1, \ldots, s_R \in \mathcal{S}_c$, it holds that:

$$f_j(s_1, \ldots, s_R) \in [0,1]. \quad (20)$$

Proof. Firstly, $f_j(s_1, \ldots, s_R) \geq 0$ is fulfilled since all of the elements in the sum in Eq. (18) are non-negative. Therefore, it suffices to show that $f_j(s_1, \ldots, s_R) \leq 1$. Since $P_{ij} \leq 1$, we can write:

$$f_j(s_1, \ldots, s_R) = \sum_{i=1}^{N} P_{ij} \left( \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} \right) \leq \sum_{i=1}^{N} \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} = 1. \quad \square$$
Lemma 3. For any \(s_1, \ldots, s_R \in S_c\), it holds that:

\[
1 - f_j(s_1, \ldots, s_R) = \sum_{k \neq j} f_k(s_1, \ldots, s_R). \tag{21}
\]

Proof. The claim is proven by the following chain of equalities:

\[
1 - f_j(s_1, \ldots, s_R) = 1 - \sum_{i=1}^{N^R} P_{ij} \left( \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} \right)
= 1 - \sum_{i=1}^{N^R} \left( 1 - \sum_{k \neq j} P_{ik} \right) \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]}
= 1 - \sum_{i=1}^{N^R} \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} + \sum_{i=1}^{N^R} \sum_{k \neq j} P_{ik} \left( \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} \right)
= \sum_{k \neq j} \sum_{i=1}^{N^R} P_{ik} \left( \prod_{m=1}^{R} s_{m, \text{ind}(i)[m]} \right) = \sum_{k \neq j} f_k(s_1, \ldots, s_R). \quad \square
\]

Proposition 2. For any \(s_1, \ldots, s_R \in S_c\) it holds that:

\[
\sum_{j=1}^{N} f_j(s_1, \ldots, s_R) = 1.
\]

Proof. This proposition is a direct consequence of Lemma 3. \quad \square

We have shown the correctness of the CCA definition. We conclude this section with the observation that the local rule \(f: S^R \to S\) of any \(N\)-state, deterministic, 1D CA can always be written in the form of Eq. (18), where \(s_1, \ldots, s_R \in S, f_j: S^R \to \{0, 1\}\) and \(\sum_{j=1}^{N} f_j(s_1, \ldots, s_R) = 1\). This shows that CCAs are a generalization of \(N\)-state, deterministic CAs.

4. Properties of SCAs

In this section, we provide a general definition of an \(N\)-state SCA, which is formulated using the notation introduced in Section 3 for multi-state CAs and CCAs. Subsequently, we show the nature of the relation between SCAs and CCAs. Finally, we present a method for decomposing an SCA into a set of deterministic CAs.

4.1. General construction of multi-state SCAs

We define SCAs, the stochastic counterpart of multi-state CAs. We assume that the state set \(S\) is given by Eq. (12), and contains all the standard base vectors \(e_i\). In such a setting, the states are assigned according to a probability
distribution, which depends on the neighborhood configuration at the previous
time step. Therefore, SCAs are not defined through a local rule, but rather by
a set of conditional expressions:

\[ P(s(c_i, t + 1) = e_j \mid s(c_{i-r}, \ldots, c_{i+r}, t) = (e_{\text{ind}(k)[1]}, \ldots, e_{\text{ind}(k)[R]})) = p_{kj}, \quad (22) \]

where \( j \in \{1, \ldots, N\} \) and \( k \in \{1, \ldots, N^R\} \). The conditional probabilities \( p_{kj} \)
in Eq. (22) must not depend on the choice of \( i \) and \( t \). Note that for every \( k \) it holds that \( \sum_{j=1}^{N} p_{kj} = 1 \), and for any \( k, j \) obviously \( p_{kj} \in [0, 1] \), such that the vector \( p_k = (p_{k1}, \ldots, p_{kN}) \in \mathcal{S}_c \) for all \( k \). Furthermore, the matrix \( P = (p_{kj}) \)
can be considered an element of \( \mathcal{S}_c^{N^R} \). Consequently, every element in \( \mathcal{S}_c^{N^R} \)
uniquely defines the probability distribution of some local rule. The matrix
\( P \) will be referred to as the probabilistic lookup table (pLUT). Such matrices
are commonly known as stochastic matrices [19] and are typically used in the
analysis of Markov chains.

Any SCA is uniquely defined by a pLUT belonging to \( \mathcal{S}_c^{N^R} \). Since we know
from Section 3.2 that every element of \( \mathcal{S}_c^{N^R} \) uniquely defines a CCA, we elaborate
on the relationship between an SCA defined by a pLUT \( P \in \mathcal{S}_c^{N^R} \) and a CCA
defined by a generalized LUT \( P \).

4.2 Relationship between CCAs and SCAs

In contrast to deterministic CAs, only the initial state \( s(\cdot, 0) \) is known with
certainty in the case of SCAs, while for \( s(\cdot, t) \), \( t > 0 \), only a probability distribution
can be calculated. Let \( \pi(c_i, t \mid I_0) \) denote the probability distribution
over the set of states \( \mathcal{S} \) for the \( i \)-th cell at time step \( t \), given that the evolution
started from the initial condition \( I_0 \), i.e. \( s(\cdot, 0) = I_0 \). Whenever it will not bring
confusion, we will omit \( I_0 \) and write \( \pi(c_i, t) \). Since there are \( N \) states, we need
to specify \( N \) probabilities summing up to 1, to define \( \pi(c_i, t) \). For this reason,
let us represent \( \pi(c_i, t) \) as a vector of probabilities, i.e. \( \pi(c_i, t) \in \mathcal{S}_c \), such that:

\[ \pi(c_i, t \mid I_0)[k] = P\left( s(c_i, t) = e_j \mid I_0 = I_0 \right), \quad (23) \]

where \( j = 1, \ldots, N \).

When using SCAs in practical problem settings, one might be interested in
computing the values of \( \pi(c_i, t) \) for any \( t \). Normally, this is achieved by means of
sampling methods [3, 4, 5], i.e. multiple space-time diagrams are generated from
the same initial condition and probability distributions are estimated, based on
the frequencies of reaching any of the states in a given cell. Unfortunately, such
an approach has serious drawbacks, especially in terms of the computational
burden it brings along. Yet, one can calculate \( \pi(c_i, t) \) much faster by relying on
CCAs.

**Proposition 3.** Let \( P \in \mathcal{S}_c^{N^R} \) be the pLUT of an SCA, and let \( f: \mathcal{S}_c^R \to \mathcal{S}_c \) be
a function defined according to Eq. (18) using \( P \) as the generalized LUT, then:

\[ \pi(c_i, t + 1) = f(\pi(c_{i-r}, t), \ldots, \pi(c_{i+r}, t)), \quad (24) \]

for any \( i \) and \( t \).
Proof. Let $k \in \{1, \ldots, N^R\}$ and $j \in \{1, \ldots, N\}$, then $p_{kj}$ is the probability of reaching state $e_j$ if the neighborhood configuration was known to be $(e_{\text{ind}(k)[1]}, \ldots, e_{\text{ind}(k)[R]})$, which will be referred to as the $k$-th neighborhood. Assuming that $\pi(\cdot, t)$ is known, the probability of cells $(c_{i-r}, \ldots, c_{i+r})$ being in the $k$-th neighborhood, can be calculated with the following formula:

$$P\left(s(c_i, t+1) = (e_{\text{ind}(k)[1]}, \ldots, e_{\text{ind}(k)[R]})\right) = \prod_{m=1}^{R} \pi(c_{i-r+m-1}, t)[\text{ind}(k)[m]]. \quad (25)$$

Due to the Total Probability Theorem, the probability of reaching state $e_j$ in at time step $t+1$ is expressed as:

$$P\left(s(c_i, t+1) = e_j\right) = \sum_{k=1}^{N^R} P\left(s(c_i, t+1) = e_j \mid s(c_i, \ldots, c_{i+r}, t) = (e_{\text{ind}(k)[1]}, \ldots, e_{\text{ind}(k)[R]})\right) \times P\left(s(c_i, \ldots, c_{i+r}, t) = (e_{\text{ind}(k)[1]}, \ldots, e_{\text{ind}(k)[R]})\right). \quad (26)$$

Substituting Eqs. (22) and (25) in Eq. (26), we get:

$$P\left(s(c_i, t+1) = e_j\right) = \sum_{k=1}^{N^R} p_{kj} \prod_{m=1}^{R} \pi(c_{i-r+m-1}, t)[\text{ind}(k)[m]]. \quad (27)$$

Taking into account the definition of the function $f$ in Eq. (18), we can rewrite Eq. (27) as:

$$P\left(s(c_i, t+1) = e_j\right) = f_j(\pi(c_{i-r}, t), \ldots, \pi(c_{i+r}, t)).$$

Since $\pi(c_i, t+1)[j] = P(s(c_i, t+1) = e_j)$ the following is proven:

$$\pi(c_i, t+1) = f(\pi(c_{i-r}, t), \ldots, \pi(c_{i+r}, t)). \quad \square$$

As shown above, the construction of CCAs is directly connected to the Total Probability Theorem, and thus it enables us to better understand the evolution of cell-wise probability distributions of SCAs. It is important to highlight that the evolution of those distributions is deterministic in SCAs, although the dynamical system itself is stochastic. The finding above agrees with [14], where a similar result in the case of binary CAs was shown. Yet, it was confined to study the asymptotic behavior of deterministic CAs. Interestingly, in [20] SCAs are defined as deterministic mappings of probability measures. Although the definition presented there is much more abstract, and suits mostly theoretical applications, it reassembles the main ideas of CCAs.
4.3. Decomposition of SCAs

In the previous subsection, we have shown that CCAs are directly related to SCAs, since the evolution of a CCA is equivalent to the evolution of the cell-wise probability distributions of an SCA. Here, we uncover the relationship between SCAs and deterministic, \(N\)-state CAs, and discuss the possibility of decomposing an SCA into deterministic CAs. Let us start with an introductory example motivating our general construction.

**Example 4 (\(\alpha\)-asynchronous CAs).** Classically, states in deterministic CAs are updated synchronously, i.e. a new state is assigned to all cells simultaneously at every time step according to the local rule. Yet, different approaches of breaking the synchronicity of CAs have been proposed [21]. Interestingly, the choice of the update scheme, which defines the order or timing of cell state updates, has very important repercussions on the dynamical properties of CAs [22]. Here we focus on so-called \(\alpha\)-asynchronous CAs (\(\alpha\)-ACAs) [23].

Any \(\alpha\)-ACA is defined by its deterministic counterpart \(A\) and a probability \(\alpha\), called the synchrony rate, which controls whether or not its cells are updated. More precisely, \(\alpha\) is the probability of applying the local rule \(f\) of \(A\), i.e.:

\[
s(c_i, t + 1) = \begin{cases} 
  f(s(c_{i-r}, \ldots, c_{i+r}, t)) & \text{, with probability } \alpha, \\
  s(c_i, t) & \text{, otherwise.}
\end{cases}
\]

Note that if \(\alpha = 0\), such a system remains at its initial configuration, whereas the system is equivalent to CA \(A\) if \(\alpha = 1\).

The essential property of the construction presented above is that an \(\alpha\)-ACA can also be seen as a synchronous, stochastic system in which the local rule of \(A\) is selected with probability \(\alpha\), while the identity rule is selected with probability \(1 - \alpha\). Hence, we may say that the rule of a CA \(A\) is stochastically mixed with the identity rule.

The idea behind \(\alpha\)-ACA can be easily extended to the mixing of \(q \geq 2\) deterministic rules. Consider synchronous, deterministic CAs \(A_1, A_2, \ldots, A_q\) and probabilities \(\alpha_1, \alpha_2, \ldots, \alpha_q\) that satisfy \(\sum_{i=1}^{q} \alpha_i = 1\). Then, we define an SCA \(\tilde{A}\) in which the state of every cell at every consecutive time step is obtained by evaluating one of the local rules, corresponding to one of the \(A_i\). The rule is chosen randomly and independently for every cell at each time step with the selection probability for CA \(A_i\) being \(\alpha_i\). Such systems, referred to as stochastic mixtures of deterministic CAs in the remainder of this paper, have very interesting properties that are currently also investigated by others [24].

For technical reasons, we assume that the local rules of \(A_1, \ldots, A_q\) are defined with a common radius \(r \geq 0\). Since each local rule with radius \(r' < r\) can be represented equivalently as a local rule with radius \(r\), we are not loosing generality with this assumption.

The first observation is a direct consequence of the definition presented above.
Fact 1. Assume that $r \geq 0$ is the radius of the neighborhood of the local rules of automata $A_1, \ldots, A_q$. Let $R = 2r + 1$ and suppose that $L_i \in S_{NR}$ is the LUT of CA $A_i$ for $i = 1, \ldots, q$. Then the pLUT $P \in S_{eNR}$ of the stochastic mixture $\tilde{A}$ satisfies:

$$P = \sum_{i=1}^{q} \alpha_i L_i.$$  \hspace{1cm} (28)

Linear combinations of the form $\sum_{i=1}^{q} \alpha_i L_i$, where $\sum_{i=1}^{q} \alpha_i = 1$ and $\alpha_i \in [0, 1]$ are commonly referred as convex combinations, which allows us to rephrase Fact 1 as: the pLUT of a stochastic mixture of deterministic CAs is the corresponding convex combination of the LUTs of the mixed CAs.

Having defined the concept of a stochastic mixture, we are interested in characterizing this class of SCAs. Interestingly, we are able to show that in general there are no characteristics that distinguish stochastic mixtures from other SCAs, since any SCA can be represented as a stochastic mixture. This fact is not surprising if we envisage a stochastic mixture of deterministic CAs as a convex combination of LUTs of deterministic CAs. We can rely on the classical Krein–Milman theorem from convex set theory [25]. It states that a compact convex set is the convex hull of its extreme points. In our context, the set $S_{eNR}$ containing the LUTs of deterministic CAs is the set of extreme points of $S_{eNR}$, which is convex and compact. The convex hull of $S_{eNR}$ is the set of all possible convex combinations of elements from $S_{eNR}$, which is precisely the set of all stochastic mixtures of deterministic CAs. Due to this we can state the following theorem.

Theorem 1. Any SCA can be represented as a stochastic mixture of a finite number of deterministic CAs.

The decomposition of a stochastic mixture into a convex combination described above is not unique, therefore many methods for constructing the decomposition can be formulated. In the proof of Proposition 4 we show one possible algorithm of which it will be shown in Proposition 5 that it uncovers the most influential component of the stochastic mixture.

Proposition 4. Let $P \in S_{eNR}$. Then there exists a $n^*$, such that for $i = 1, 2, \ldots, n^*$, we can define coefficients $\alpha_i \in [0, 1]$, $\sum_{i=1}^{n^*} \alpha_i = 1$ and matrices $L^i \in S_{NR}$, that satisfy:

$$P = \sum_{i=1}^{n^*} \alpha_i L^i.$$  \hspace{1cm} (29)

Proof. We start by defining auxiliary matrices $P^m$, for $m \geq 0$. Let:

$$P^m = \begin{cases} P, & \text{if } m = 0, \\ P^{m-1} - \alpha_m L^m, & \text{if } m > 0. \end{cases}$$  \hspace{1cm} (30)
where, for $m > 0$:  
\[ \alpha_m = \min_i \max_j P_{ij}^{m-1}, \]  
(31)

and $L^m = (L_{ij}^m)$ such that:
\[ L_{ij}^m = \begin{cases} 1, & \text{for } j = \min \left\{ j \mid P_{ij}^{m-1} \leq P_{il}^{m-1}, \text{for any } l \in \{1, \ldots, N\} \right\}, \\ 0, & \text{otherwise}. \end{cases} \]  
(32)

Note that, for any $m$, $i$, it holds that $L_{ij}^m = 1$ for only one value of $j$. Consequently $L^m \in S^{N \times R}$.

Firstly, note that $\alpha_1$ is one of the elements of $P^0 = P$, so $\alpha_1 \in [0, 1]$. Additionally, $\alpha_1 > 0$, since if $\alpha_1 = 0$, then for some $i$ we would have $\sum_j P_{ij} = 0$, which contradicts the fact that $\sum_j P_{ij} = 1$ for any $i$.

Secondly, note that for any $i$, $j$, it holds that $P_{ij}^0 \geq 0$. We will show next that the assumption $P_{ij}^{m-1} \geq 0$ leads to $P_{ij}^m \geq 0$. Indeed $P_{ij}^m \in \{P_{ij}^{m-1}, P_{ij}^{m-1} - \alpha_m\}$. If $P_{ij}^m = P_{ij}^{m-1}$, then obviously $P_{ij}^m \geq 0$, while if $P_{ij}^m = P_{ij}^{m-1} - \alpha_m$ it means that $P_{ij}^{m-1} \geq P_{il}^{m-1}$ for any $l$, and $P_{ij}^{m-1} = \max_l P_{il}^{m-1}$. On the other hand, since $\alpha_m = \min_l \max_l P_{il}^{m-1}$, it means that $P_{ij}^{m-1} \geq \alpha_m$. Since it was assumed that $P_{ij}^{m-1} \geq 0$, we see that $P_{ij}^m \geq 0$ as well.

Thirdly, following the argument above, we can easily show that for every $i$, $j$, $m$, it holds that $P_{ij}^m \leq P_{ij}^{m-1}$. This means that for every $i$, $\max_j P_{ij}^m \leq \max_j P_{ij}^{m-1}$, and thus $\alpha_{m+1} \leq \alpha_m$. Since $\alpha_1 \in [0, 1]$, and $\alpha_m \geq 0$ for any $m$, we see that $\alpha_m \in [0, 1]$.

Note that for each $i$ it holds that $\sum_j P_{ij}^m = \left( \sum_j P_{ij}^{m-1} \right) - \alpha_m$. Expanding this recursively, gives us $\sum_j P_{ij}^m = (\sum_j P_{ij}^0) - \sum_{l=1}^m \alpha_l$. From the definition of a stochastic matrix, we know that $\sum_j P_{ij}^0 = 1$, such that $\sum_j P_{ij}^m = 1 - \sum_{l=1}^m \alpha_l$, which means that the column sums in matrix $P^m$ are equal. Since for all $m$ we have shown that $P_{ij}^m \geq 0$, we know that for each $m$ it holds $1 - \sum_{l=1}^m \alpha_l \geq 0$, therefore $\sum_{l=1}^m \alpha_l \leq 1$.

We further note that $\alpha_m = 0$ for some $m$, if and only if, $P^{m-1}$ contains one zero column. Since we have shown that the column sums in $P^m$ are equal for any $m$, we know that $\alpha_m = 0$ if and only if, $P^{m-1} = 0$. Therefore, $\alpha_m > 0$ if and only if $P^{m-1} \neq 0$, which implies that in each of the columns of $P^{m-1}$ there is at least one non-zero entry.

Let $z(m) = \# \{(i, j) \mid P_{ij}^m = 0\}$ denote the number of zeros in matrix $P^m$. Note that if $P^m \neq 0$, then $z(m) < z(m + 1)$, which follows from the fact that if $P^m \neq 0$, there exists $P_{ij}^m = \alpha_{m+1} > 0$, and thus for at least one couple $(i, j)$ we know that $P_{ij}^{m+1} = P_{ij}^m - \alpha_{m+1} = 0$. Since $z(m)$ cannot be greater than the total number of elements in $P^m$, it is not possible that $z(m) < z(m + 1)$ for all $m$. As such, we know that there exist $n^*$ for which $P^{n^*} = 0$ and $P^{n^*-1} \neq 0$, and thus $\sum_{l=1}^{n^*} \alpha_l = 1$. Using the definition of $P^{n^*}$ multiple times, we get:
\[ 0 = P^{n^*} = P^{n^*-1} - \alpha_{n^*} L^{n^*} = \ldots = P^0 - \alpha_1 L^1 - \ldots - \alpha_{n^*} L^{n^*} = P - \sum_{i=1}^{n^*} \alpha_i L^i, \]
such that \( P = \sum_{i=1}^{n^*} \alpha_i L^i \).

To illustrate the construction presented in the proof of Proposition 4, every step of the algorithm is visualized in the following example.

**Example 5.** Although Proposition 4 deals with stochastic matrices with dimensions corresponding to pLUTs, it is obvious that the same method can be used to decompose a stochastic matrix of arbitrary size. For simplicity, we apply the method to an exemplary \( 4 \times 3 \) stochastic matrix \( P \) given by:

\[
P = \begin{bmatrix}
0.6 & 0 & 0.2 & 0.4 \\
0.3 & 1 & 0.3 & 0.4 \\
0.1 & 0 & 0.5 & 0.2
\end{bmatrix}.
\]

According to Eqs. (30) and (32), we build matrices \( P^0 \) and \( L^1 \):

\[
P^0 = P = \begin{bmatrix}
0.6 & 0 & 0.2 & 0.4 \\
0.3 & 1 & 0.3 & 0.4 \\
0.1 & 0 & 0.5 & 0.2
\end{bmatrix}, \quad L^1 = \begin{bmatrix}
1 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}.
\]

Note that we highlighted the selected maximal elements in each of the columns using boldfaced font. The boldfaced entries in \( P^0 \) correspond to the positions of one in matrices \( L^1 \). Note that in the last column of \( P^0 \), the choice for the maximal element is not unique. Yet, as given by Eq. (32), we pick the first of the entries from top to bottom. Picking the other possibility would affect the subsequent matrices \( L^m \) and result in a different decomposition, but the coefficients \( \alpha_m \) would not be changed.

Following Eq. (31), we find that \( \alpha_1 = 0.4 \). We proceed, and calculate \( P^1 \) and \( L^2 \):

\[
P^1 = P^0 - \alpha_1 L^1 = \begin{bmatrix}
0.2 & 0 & 0.2 & 0 \\
0.3 & 0.6 & 0.3 & 0.4 \\
0.1 & 0 & 0.1 & 0.2
\end{bmatrix}, \quad L^2 = \begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0
\end{bmatrix},
\]

and from this we see that \( \alpha_2 = 0.3 \). We continue the procedure:

\[
P^2 = P^1 - \alpha_2 L^2 = \begin{bmatrix}
0.2 & 0 & 0.2 & 0 \\
0 & 0.3 & 0 & 0.1 \\
0.1 & 0 & 0.1 & 0.2
\end{bmatrix}, \quad L^3 = \begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix},
\]

and \( \alpha_3 = 0.2 \). We proceed one more step and get:

\[
P^3 = P^2 - \alpha_3 L^3 = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0.1 & 0 & 0.1 \\
0.1 & 0 & 0.1 & 0
\end{bmatrix}, \quad L^4 = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
1 & 0 & 1 & 0
\end{bmatrix},
\]

and \( \alpha_4 = 0.1 \). This is the final step, since: \( P^4 = P^3 - \alpha_4 L^4 = 0 \). Therefore,
the decomposition can be written as:

\[
P = 0.4 \times \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} + 0.3 \times \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} + 0.2 \times \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + 0.1 \times \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}.
\]

As mentioned earlier, the construction presented in the proof of Proposition 4 is one of many possibilities of decomposing a pLUT, but in the following proposition we will show that it enables us to capture the element of the mixture with the highest possible coefficient.

**Proposition 5.** Let \( P \in S_{\mathbb{N}^R}^{N_R}, p = 1, \ldots, p_{\text{max}}, q = 1, \ldots, q_{\text{max}}, \) and let \( \alpha_p, \beta_q \in [0, 1], L^p, M^q \in S_{\mathbb{N}^R}^{N_R} \) be such that \( \sum_p \alpha_p = \sum_q \beta_q = 1 \) and \( P = \sum_p \alpha_p L^p = \sum_q \beta_q M^q \). Let \( \alpha_p, L^p \) be defined as in the proof of Proposition 4.

Then

\[
\max_q \beta_q \leq \alpha_1 = \max_p \alpha_p.
\]

**Proof.** Let \( i, j \) be such that \( P_{ij} = \alpha_1 \). Then for all \( k \) it holds that \( P_{ik} \leq \alpha_1 \). Let \( q \in \{1, \ldots, q_{\text{max}}\} \) and let \( l \) be such that \( M_q^l = 1 \). Since \( P = \sum_q \beta_q M^q \), we know that \( \beta_q \leq P_{il} \), and thus \( \beta_q \leq P_{il} \leq \alpha_1 \). Therefore, for any \( q \in \{1, \ldots, q_{\text{max}}\} \) it holds that \( \beta_q \leq \alpha_1 \) and thus \( \max_q \beta_q \leq \alpha_1 \).

The informal meaning of the above proposition is that the presented approach to decompose an SCA uncovers the deterministic rule that has the highest probability of being executed, and therefore is likely to have the highest influence on the behavior of the system. The exact relations between the dynamical behavior of SCAs and the dynamical characteristics of the components of stochastic mixtures are still under investigation.

### 5. Experiments

#### 5.1. Analysis of \( \alpha \)-asynchronous ECAs

Through this experiment, we have analyzed the behavior of \( \alpha \)-asynchronous ECAs. For a given rule, the cell-wise distance between the space-time diagram of the deterministic rule, and the space–time diagram of a CCA representation of an \( \alpha \)-ACA variant for \( \alpha \in [0, 1] \), was measured. More formally, if \( A \) is an ECA rule, and \( A_{\alpha} \) the CCA representation of the \( \alpha \)-asynchronous variant of \( A \) for \( \alpha \in [0, 1] \), we measured the distance \( D(\alpha) \) between space-time diagrams defined as:

\[
D(\alpha) = \frac{1}{MT} \sum_{t=1}^{T} \|A^t(I_0) - A_{\alpha}^t(I_0)\|,
\]

for a random initial condition \( I_0 \in \{0, 1\}^M \), where \( M > 0 \) denotes the number of cells and \( T \) the number of time steps, and \( A^t(I_0) \) denotes the result of applying
the global rule $A$, $t$ times to input $I_0$. In this experiment, we choose $M = T = 69$.

Unsurprisingly, it turned out that most pronounced discrepancies between the parallel evolutions are observed when $\alpha$ approaches 1. For that reason, we restrict the further discussion to $\alpha \in [0, 1]$. We classified the ECAs according to the behavior of the function $D$ (the contents of the classes is shown in Table 1):

- **Class I**: $D(\alpha)$ is almost 0, for all $\alpha \in [0.9, 1]$ (Figure 1a).
- **Class II**: $D(\alpha)$ smoothly decreases towards 0, as $\alpha$ increases (Figure 1b).
- **Class IIIa**: there is a very sudden drop of $D(\alpha)$ as $\alpha$ approaches 1 (Figure 1c).
- **Class IIIb**: there is a very sudden drop of $D(\alpha)$ as $\alpha$ approaches 1, and the behavior is not monotonic (Figure 1d).

The interpretation of the classes is as follows. Rules belonging to Class I, are resistant to $\alpha$–asynchronicity, which means that introducing the asynchronicity aspect does not affect the behavior, or that its impact is negligible. Rules belonging to Class II are affected by the asynchronicity, but the impact on the final behavior is proportional to the synchrony rate $\alpha$, i.e. the behavior steadily
Table 1: Classes of $\alpha$–Asynchronous ECAs.

| Class     | Elements                                                                 |
|-----------|--------------------------------------------------------------------------|
| Class I   | 0, 8, 12, 32, 40, 64, 68, 72, 76, 77, 93, 96, 128, 132, 136, 140, 160, 168, 192, 196, 200, 205–207, 220, 221, 224, 232, 233, 235–239, 249–255 |
| Class II  | 1–5, 7, 10, 13, 15–17, 19, 21, 23, 24, 29, 31, 34, 36, 42, 44, 48, 50, 51, 55, 56, 63, 66, 69, 71, 79, 80, 85, 87, 92, 95, 100, 104, 108, 112, 119, 127, 130, 138, 141, 144, 152, 162, 164, 170–172, 174–176, 178, 179, 186–191, 197, 201–203, 208, 216–219, 222, 223, 226, 228, 230, 231, 234, 240–248 |
| Class IIIa| 6, 9, 18, 20, 22, 25–28, 30, 33, 35, 37–39, 41, 45, 46, 49, 52–54, 57–62, 65, 67, 70, 73–75, 78, 82, 83, 86, 88–91, 94, 97–99, 101–103, 105–107, 109–111, 114–116, 118, 120–126, 129, 131, 133–135, 137, 139, 145–151, 153–159, 161, 163, 165–167, 169, 173, 177, 180–185, 193, 195, 198, 199, 204, 209–211, 214, 215, 225, 227, 229 |
| Class IIIb| 11, 14, 43, 47, 81, 84, 113, 117, 142, 143, 212, 213 |

approaches the deterministic case, as $\alpha$ approaches one. Rules belonging to Classes IIIa and IIIb are sensitive to $\alpha$–asynchronicity, meaning that the behavior of the system changes drastically as soon as $\alpha$ is smaller than 1. Class IIIb can be distinguished from Class IIIa, by the noisy behavior of $D(\alpha)$ for $\alpha$ close to one. The cause of the differences is not yet uncovered.

5.2. Stochastic density classifiers

We consider the SCA defined by the local rule $C_3$ introduced in [26, 27]. This rule is defined by the pLUT shown in Table 2. Note that this pLUT uses the standard notation for its state set $S = \{0, 1\}$, and its entries constitute the probabilities of reaching state 1. It was shown both analytically and experimentally, that this rule is a stochastic solution for the classical density classification problem (DCP) [20] with arbitrary precision, i.e. by varying the parameter $\eta$ we can achieve any probability $p < 1$ of successful classification. The DCP was introduced in [28, 29].

| \[(1, 1, 1)\] | \[(1, 1, 0)\] | \[(1, 0, 1)\] | \[(1, 0, 0)\] | \[(0, 1, 1)\] | \[(0, 1, 0)\] | \[(0, 0, 1)\] | \[(0, 0, 0)\] |
|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| 1 | $\eta$ | $1$ | $1 - \eta$ | $1$ | $0$ | $0$ | $0$ |

Table 2: pLUT of local rule $C_3$ [26, 27].

Using the CCA representation, we can visualize the average behavior of the $C_3$. In Figure 2 the first 80 rows of the averaged space-time diagram (time goes from top to bottom) obtained using the CCA representation of $C_3$ with $\eta = 0.1$ are shown, together with three typical space-time diagrams obtained by direct evaluation of the SCA rule $C_3$. All images were obtained using the
same initial condition involving 29 cells out of which 16 were black (state 1). As can be inferred from Figure 2, two samples lead to behavior that is similar to the one displayed by the average space-time diagram, while the one depicted in Figure 2d behaves differently, and leads to a wrong classification. Since the probability of obtaining a correct classification for the initial condition at stake was estimated to be 90.1% on the basis of 10000 repetitions, the impact of the erroneous cases on the averaged image is hardly visible.

One of the most interesting questions within the analysis of DCP solutions relates to the expected time of convergence towards the outcome of the classification and how this relates to the number of cells. Figure 3 depicts the average convergence time calculated over an ensemble of 5000 random initial conditions for different numbers of cells and values of \( \eta \), both for the CCA and SCA representation. The initial conditions were generated as follows. For every initial condition independently, a number \( p \in [0, 1] \) was selected randomly, and then the initial states were selected with \( p \) being the probability of selecting state 0. Such a selection procedure assures that each of the possible densities has the same probability in the ensemble of initial conditions, and thus we can evaluate the classifier across a diverse set of densities.

In the case of the CCA representation, we cannot expect reaching a truly homogeneous, global state. Therefore we evolved it until the maximum, absolute difference between the states of any two cells was lower than 0.001. Then, we verified whether the average of the states was closer to 1 or 0. In the case of the SCA, for each of the 5000 initial conditions, 100 simulations were performed, and the results were averaged.

Figure 2: Space time diagrams of \( C_3 \) with \( \eta = 0.1 \).
Figure 3: Expected convergence time of rule $C_3$ for different values of $\eta$.

The results for small grid sizes (smaller than 200) are similar for both representations and agree with the findings presented in [26, 27]. Yet, the computing time when using the CCA representation is significantly lower than in the case of SCAs. This indicates that the CCA representation of SCAs proposed in this paper might enable substantial reduction of the required computing time. Moreover, using the CCA representation, we are able to get more insight into the behavior of the dynamical system by plotting the evolution of the density over time. Figure 4 shows the results of such an experiment. For the sake of clarity, these results are based on an ensemble of 400 initial conditions (69 cells) out of which 200 had a density greater than a half (green), while the other 200 had a density lower than the half (red). The ensemble of initial configurations was the same for all values of $\eta$. Note that the range of the horizontal axis differs across plots. From Figure 4, we can infer that the time and quality of classification increases with decreasing $\eta$. Due to computational costs involved in evolving and averaging SCAs directly, drawing and analyzing such plots has become possible only due to the introduction of CCAs.

5.3. Totalistic SCAs

Consider the class of 1D, binary SCAs with a unit neighborhood radius ($r = 1$), that satisfy the following conditions:

\[
\begin{align*}
P(s(c_i, t + 1) = e_2 \mid s(c_{i-1}, c_i, c_{i+1}, t) = (e_2, e_1, e_1)) &= \\
P(s(c_i, t + 1) = e_2 \mid s(c_{i-1}, c_i, c_{i+1}, t) = (e_1, e_2, e_1)) &= \\
P(s(c_i, t + 1) = e_2 \mid s(c_{i-1}, c_i, c_{i+1}, t) = (e_1, e_1, e_2)) &= p_1, \\
P(s(c_i, t + 1) = e_2 \mid s(c_{i-1}, c_i, c_{i+1}, t) = (e_1, e_2, e_2)) &= \\
P(s(c_i, t + 1) = e_2 \mid s(c_{i-1}, c_i, c_{i+1}, t) = (e_2, e_1, e_2)) &= \\
P(s(c_i, t + 1) = e_2 \mid s(c_{i-1}, c_i, c_{i+1}, t) = (e_2, e_2, e_1)) &= p_2,
\end{align*}
\]
Figure 4: Density evolution over time for the CCA representation of SCA $C_3$ for 200 initial conditions with density above half (green) and 200 others with density below half (red) and different values $\eta$. 

(a) $\eta = 0.1$
(b) $\eta = 0.05$
(c) $\eta = 0.01$
(d) $\eta = 0.005$
(e) $\eta = 0.001$
where \( e_1 \) and \( e_2 \) are the base vectors of the Euclidean space \( \mathbb{R}^2 \). Such SCAs are commonly referred to as totalistic SCAs. We consider a subclass of such SCAs, satisfying:

\[
\begin{align*}
P(s(c_i, t + 1) = e_2 | s(c_{i-1}, c_i, c_{i+1}, t) = (e_2, e_2, e_2)) &= 1, \\
P(s(c_i, t + 1) = e_1 | s(c_{i-1}, c_i, c_{i+1}, t) = (e_1, e_1, e_1)) &= 1.
\end{align*}
\]

In this paper, we will refer to this subclass as totalistic SCAs. By applying the decomposition algorithm given by Proposition 4, we can prove the following fact showing their structure.

**Fact 2.** Any totalistic SCA can be written as a stochastic mixture of the following ECAs: 128, 150, 232, 254, where up to three of those rules are applied with non-zero probability.

We can determine the regions in the \((p_1, p_2)\)-plane where a given deterministic CA has the highest probability of application, i.e. where it is dominant. These regions are given by:

- ECA 128 is dominant, if \( p_1 < 0.5 \) and \( p_2 < 0.5 \),
- ECA 150 is dominant, if \( p_1 < 0.5 \) and \( p_2 > 0.5 \),
- ECA 232 is dominant, if \( p_1 > 0.5 \) and \( p_2 < 0.5 \),
- ECA 254 is dominant, if \( p_1 > 0.5 \) and \( p_2 > 0.5 \).

Note that if \( p_1 = 0.5 \) or \( p_2 = 0.5 \), there is no unique, dominant rule, therefore we have omitted those cases in the description above.

Let \( \alpha_{128} \), \( \alpha_{150} \), \( \alpha_{232} \), \( \alpha_{254} \) denote the probabilities of applying ECAs 128, 150, 232, 254, respectively. Figure 5 depicts the dependence of \( \alpha_{150} \) and \( \alpha_{232} \) on \( p_1 \) and \( p_2 \). We have omitted the remaining two images for \( \alpha_{128} \) and \( \alpha_{254} \), since they do not yield additional information.

The dynamical characteristics of the four ECAs that compose the totalistic SCAs differ significantly. Rules 128 and 254 are simple – they belong to Class I according to Wolfram’s classification [16], and their maximum Lyapunov exponents (MLEs) [30] equals \(-\infty\), which means that any changes to their initial configuration do not influence their long-term behavior. Rule 232 belongs to complexity Class II, and its MLE is positive, but very close to zero. In contrast, rule 150 is a Class III rule, and its MLE is the highest among all ECAs, which highlights the fact that this rule is sensitive to the smallest perturbation of its initial configuration. Since rules 150 and 232 are relatively more complex than rules 128 and 254, we might expect a distinct behavior of the stochastic mixture in regions where the former are dominant.

In order to verify this we set up an experiment involving a random initial condition of \( M = 49 \) cells which was evolved 100 times for \( T = 49 \) time steps. Such a procedure, for the same initial condition, was repeated for multiple different choices of \((p_1, p_2)\) using a \(100 \times 100\) homogenous grid, resulting in a
Figure 5: Dependence of the application probabilities: $\alpha_{150}$ of ECA 150 and $\alpha_{232}$ of ECA 232 in the totalistic SCA, on the values of parameters $p_1$, $p_2$.

set of space-time diagrams. The set of such space-time diagrams is denoted as $\mathcal{I}(p_1, p_2)$. Let $\Delta(p_1, p_2) = \{\text{dist}(I, J) \mid I \neq J; I, J \in \mathcal{I}(p_1, p_2)\}$ denote the set of all pair-wise Hamming distances between space-time diagrams, and $\Delta_T(p_1, p_2) = \{\text{dist}(I[T], J[T]) \mid I \neq J; I, J \in \mathcal{I}(p_1, p_2)\}$ denote the set of all pair-wise Hamming distances between the final configurations in the space-time diagrams. The results for the minimum, mean and maximum of $\Delta(p_1, p_2)$ and $\Delta_T(p_1, p_2)$ are shown in Figure 6.

As can be seen from the charts, the distance is especially high in those regions where the application probability $\alpha_{150}$ is high, i.e. where $p_1 > 0.5$ and $p_2 < 0.5$, and near the line $p_2 = -p_1$ in the quarter $p_1 < 0.5, p_2 > 0.5$. The first region is the one where ECA 150 is dominant, while ECA 232 is dominant in the second one, but with ECA 150 having a substantially high application probability as well. Although the Hamming distance is just a simple indication of complexity, we already see a strong influence of the ECA 150 component.

The findings presented above suggest that analyzing the elements of a stochastic mixture unveil information on the dynamics of an SCA. Further research in this direction is undertaken to understand the influence of different dynamical classes of deterministic rules in the stochastic mixtures.

6. Summary

In this paper, two basic, yet important properties of SCAs were discussed. We have shown, both theoretically and practically that SCAs can be effectively analyzed in the context of deterministic CAs and CCAs. The stochastic mixture representation of SCAs in some cases allows to understand the underlying dynamics, while the CCA representation allows to quickly uncover the averaged behavior of the system. Further research is undertaken to further expand the application scope of the presented methods.
Figure 6: Impact of the probabilities $p_1$ and $p_2$ on the distances of space-time diagrams $\Delta(p_1, p_2)$ and the final configurations after $T = 49$ time steps $\Delta_T(p_1, p_2)$ of the totalistic SCA, obtained from the same initial condition.
Acknowledgments

Witold Bolt is supported by the Foundation for Polish Science under International PhD Projects in Intelligent Computing. Project financed from The European Union within the Innovative Economy Operational Program 2007-2013 and European Regional Development Fund.

The authors gratefully acknowledge the financial support of the Research Foundation Flanders (FWO project 3G.0838.12.N).

Bibliography

References

[1] N. Ganguly, B. K. Sikdar, A. Deutsch, G. Canright, P. P. Chaudhuri, A survey on cellular automata, Tech. rep. (2003).

[2] D. Das, A survey on cellular automata and its applications, in: Global Trends in Computing and Communication Systems, Springer, 2012, pp. 753–762.

[3] M. Precharattana, W. Triampo, Effects of initial concentration and severity of infected cells on stochastic cellular automaton model dynamics for hiv infection, in: G. Sirakoulis, S. Bandini (Eds.), Cellular Automata, Vol. 7495 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 454–463.

[4] P. Van der Weeën, J. M. Baetens, J. Verwaeren, X. Van Doorslaer, P. Heynderickx, J. Dewulf, B. De Baets, Modeling the photocatalytic degradation of moxifloxacin by means of a stochastic cellular automaton, Chemical Engineering Journal 188 (2012) 181–190.

[5] P. Van der Weeën, N. De Clercq, J. M. Baetens, C. Delbaere, K. Dewettinck, B. De Baets, A discrete stochastic model for oil migration in chocolate-coated confectionery, Journal of Food Engineering 119 (2013) 602–610.

[6] P. Balister, B. Bollobás, R. Kozma, Large deviations for mean field models of probabilistic cellular automata, Random Structures & Algorithms 29 (2006) 399–415.

[7] J. Mairesse, I. Marcovici, Around probabilistic cellular automata, Theoretical Computer Science 559 (2014) 42–72.

[8] J. Lebowitz, C. Maes, E. Speer, Statistical mechanics of probabilistic cellular automata, Journal of Statistical Physics 59 (1990) 117–170.

[9] M. Schüle, T. Ott, R. Stoop, Computing with probabilistic cellular automata, in: C. Alippi, M. Polycarpou, C. Panayiotou, G. Ellinas (Eds.), Artificial Neural Networks – ICANN 2009, Vol. 5769 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 525–533.
[10] J. von Neumann, The general and logical theory of automata, in: L. A. Jeffres (Ed.), Cerebral Mechanisms in Behaviour. The Hixon Symposium, John Wiley & Sons, Pasadena, CA, USA, 1951, pp. 1–41.

[11] K. Kaneko, Theory and Applications of Coupled Map Lattices, Wiley, New York, NY, USA, 1993.

[12] P. Flocchini, F. Geurts, A. Mingarelli, N. Santoro, Convergence and aperiodicity in fuzzy cellular automata: Revisiting rule 90, Physica D: Nonlinear Phenomena 142 (2000) 20–28.

[13] H. Betel, P. Flocchini, On the asymptotic behavior of fuzzy cellular automata, Electronic Notes in Theoretical Computer Science 252 (2009) 23–40.

[14] H. Betel, P. Flocchini, On the relationship between boolean and fuzzy cellular automata, Electronic Notes in Theoretical Computer Science 252 (2009) 5–21.

[15] S. Wolfram, Statistical mechanics of cellular automata, Reviews of Modern Physics 55 (1983) 601–644.

[16] S. Wolfram, Universality and complexity in cellular automata, Physica D: Nonlinear Phenomena 10 (1984) 1–35.

[17] M. Schüle, T. Ott, R. Stoop, Global dynamics of finite cellular automata, in: V. Kůrková, R. Neruda, J. Koutník (Eds.), Artificial Neural Networks - ICANN 2008, Vol. 5163 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 71–78.

[18] W. Rudin, Principles of Mathematical Analysis, 3rd Edition, McGraw–Hill Book Co., New York, 1976, International Series in Pure and Applied Mathematics.

[19] G. Latouche, V. Ramaswami, Introduction to Matrix Analytic Methods in Stochastic Modeling, Society for Industrial and Applied Mathematics, 1999.

[20] A. Bušić, J. Mairese, I. Marcovici, Probabilistic cellular automata, invariant measures, and perfect sampling, Advances in Applied Probability 45 (2013) 960–980.

[21] B. Schönfisch, A. de Roos, Synchronous and asynchronous updating in cellular automata, BioSystems 51 (1999) 123–143.

[22] J. M. Baetens, P. Van der Weeën, B. De Baets, Effect of asynchronous updating on the stability of cellular automata, Chaos, Solitons & Fractals 45 (2012) 383–394.

[23] N. Fatès, M. Morvan, An experimental study of robustness to asynchronism for elementary cellular automata, Complex Systems 16 (2005) 1–27.
[24] J. Mendonça, M. de Oliveira, An extinction-survival-type phase transition in the probabilistic cellular automaton $p182 - q200$, Journal of Physics A: Mathematical and Theoretical 44 (15) (2011) Article ID 155001, 10 p.

[25] M. Krein, D. Milman, On extreme points of regular convex sets, Studia Mathematica 9 (1940) 133–138.

[26] N. Fatès, Stochastic cellular automata solutions to the density classification problem - when randomness helps computing, Theory of Computing Systems 53 (2013) 223–242.

[27] N. Fatès, A guided tour of asynchronous cellular automata, in: J. Kari, M. Kutrib, A. Malcher (Eds.), Cellular Automata and Discrete Complex Systems, Vol. 8155 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, pp. 15–30.

[28] P. Gács, G. L. Kurdyumov, L. A. Levin, One-dimensional uniform arrays that wash out finite islands, Problemy Peredachi Informatsii 14 (3) (1978) 92–96.

[29] N. H. Packard, Adaptation toward the edge of chaos, University of Illinois at Urbana-Champaign, Center for Complex Systems Research, Champaign, IL, USA, 1988.

[30] F. Bagnoli, R. Rechtman, S. Ruffo, Damage spreading and lyapunov exponents in cellular automata, Physics Letters A 172 (1992) 34–38.