Identification of Cellular Automata Based on Incomplete Observations With Bounded Time Gaps

Witold Bołt, Jan M. Baetens, and Bernard De Baets

***Abstract*—In this paper, the problem of identifying the cellular automata (CAs) is considered. We frame and solve this problem in the context of incomplete observations, i.e., prerecorded, incomplete configurations of the system at certain, and unknown time stamps. We consider 1-D, deterministic, two-state CAs only. An identification method based on a genetic algorithm with individuals of variable length is proposed. The experimental results show that the proposed method is highly effective. In addition, connections between the dynamical properties of CAs (Lyapunov exponents and behavioral classes) and the performance of the identification algorithm are established and analyzed.**

***Index Terms*—Cellular automata (CAs), genetic algorithms (GAs), nonlinear dynamical systems, system identification.**

I. Introduction

CELLULAR automata (CAs) constitute an attractive and effective modeling paradigm for a variety of problems [1]. In order to use CAs for a practical modeling task, one needs to understand the mechanisms underlying the phenomenon at stake, and translate them into a CA rule. Additionally, the state space and neighborhood structure need to be pinned down beforehand. This limits the use of CAs, since there are problems for which manually designing a local rule is hard. Moreover, in some cases only the initial and final states of the systems are known (e.g., [2]–[4]). Research on automated CA identification is motivated by such problems. Various methods have been studied so far in this field: genetic algorithms (GAs) [5]–[8]; genetic programming [9]–[11]; gene expression programming [12]; other evolutionary algorithms [13]; ant colony algorithms [14]; machine learning approaches [15]; as well as direct construction algorithms [16]–[19]. Within these methods, two main groups can be identified. First, there are methods for solving specific, global problems. An example of such a problem is the density classification problem in which only the initial condition and the desired outcome are known [20]–[23]. Second, there are methods that exploit the entire time series of configurations. Typically, it is assumed that all configurations are known (i.e., observed completely). Only limited efforts have been devoted to solve the identification problem in the absence of complete information [5].

The main goal of the research presented in this paper is to design a method capable of automated CA identification in the case of incomplete information. The incompleteness of the information, within the scope of this paper, is of twofold nature. First, we consider the *spatial incompleteness*, which relates to missing states of specific cells at one or more time stamps. Second, we consider the *temporal incompleteness*, which relates to missing CA configurations at certain time stamps. In contrast to the spatial incompleteness, where the location and therefore the amount of missing information is known, these parameters are not known in the case of temporal incompleteness, i.e., a certain, unknown number of time stamps are missing in between every consecutive pair of observed time stamps. Such a setting has not yet been discussed in the literature. In this paper, a detailed definition of this identification problem is presented. Moreover, an effective solution algorithm based on a GA is presented.

It is important to understand the motivation behind the problem definition discussed in this paper. Essentially, spatial incompleteness can be related to malfunctioning measuring equipment or limited capabilities of observing real-world phenomena. Such scenarios are very likely in many practical applications [24]. Temporal incompleteness, on the other hand, can be understood in two ways. First, similar to spatial incompleteness, it can be related to limited observation capabilities, i.e., the frequency of capturing the time stamps might be limited and unstable. Second, there is an issue of synchronization. The clock of the phenomenon in question, and the one of the prospective CA-based models, are not necessarily in synchrony. Due to this, the number of CA time stamps in between two consecutive observed time stamps might be unknown and might be changing dynamically. Consequently, although the problem is discussed in a theoretical setting of 1-D, two-state CAs, the nature of the tackled problems relates directly to the one of the practical modeling.

This paper is organized as follows. Section II introduces the basic definitions. In Section III, the CA identification problem is formally defined. Further, in Section IV, this problem is then reformulated as an optimization task. Section V presents the algorithm for solving the identification problem.

Manuscript received April 3, 2018; revised June 29, 2018 and September 25, 2018; accepted October 3, 2018. Date of publication October 29, 2018; date of current version January 21, 2020. This work was supported by the Research Foundation Flanders (FWO) under Project 3G.0838.12.N. This paper was recommended by Associate Editor L. Rutkowski. *(Corresponding author: Witold Bołt.)*

W. Bołt is with the Systems Research Institute, Polish Academy of Sciences, 01-447 Warsaw, Poland, and also with KERMIT, Department of Data Analysis and Mathematical Modelling, Ghent University, 9000 Ghent, Belgium (e-mail: witold.bolt@hope.art.pl).

J. M. Baetens and B. De Baets are with KERMIT, Department of Data Analysis and Mathematical Modelling, Ghent University, 9000 Ghent, Belgium.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2018.2875266

TABLE I
LUT OF LOCAL RULE $n = (\ell_8, \ell_7, \ell_6, \ell_5, \ell_4, \ell_3, \ell_2, \ell_1)_2$

111	110	101	100	011	010	001	000
ℓ_8	ℓ_7	ℓ_6	ℓ_5	ℓ_4	ℓ_3	ℓ_2	ℓ_1

The experimental results are presented in detail in Section VI, while Section VII summarizes the main results.

II. Preliminaries

We concentrate on 1-D, deterministic, binary CAs with a finite number of cells, defined by a local rule with a symmetric neighborhood. Yet, the problem definition and the identification algorithm can be extended to higher dimensions and more complex state sets. Below, we give a definition of the CAs that are considered in this paper. Let $f_A\colon \{0,1\}^{2r+1} \to \{0,1\}$ be a function, where $r \in \mathbb{N}$, then, for any integer N, we define the N-cell global CA rule $A_N\colon \{0,1\}^N \to \{0,1\}^N$ as

$$A_N(s_1, \ldots, s_i, \ldots, s_N) = \left(s'_1, \ldots, s'_i, \ldots, s'_N\right) \tag{1}$$

where $s'_i = f_A(s_{i-r}, \ldots, s_{i+r})$ and the periodic boundary conditions are assumed, i.e., $s_{i+N} = s_i$ for any $i \in \mathbb{Z}$. Such function f_A will be referred to as a *local rule*, while the integer r will be referred to as the *neighborhood radius* or simply *radius*. Any local rule can be uniquely defined by a lookup table (LUT), listing all the possible arguments and mapping them to the corresponding function values. The ordering of the arguments in an LUT is assumed to be lexicographic, thus only the second row needs to be stored. Therefore, an LUT will be represented as a binary vector of length 2^{2r+1}. The general form of an LUT describing a local rule with unit radius ($r = 1$) is shown in Table I. Note that LUTs can be used to enumerate local rules, as the coefficients ℓ_i can be treated as digits in the binary representation of an integer n, i.e., the number of a local rule is given by $n = \sum_{i=1}^{8} \ell_i 2^{i-1} = (\ell_8, \ell_7, \ldots, \ell_1)_2$. Clearly, this extends to larger radii.

The set of all binary sequences of finite length will be denoted by $\{0,1\}^*$, i.e., $\{0,1\}^* = \bigcup_{M\in\mathbb{N}_0}\{0,1\}^M$. The function $A\colon \{0,1\}^* \to \{0,1\}^*$, satisfying $A(X) = A_M(X)$ if $X \in \{0,1\}^M$, where each of the global rules A_M is defined by the same local rule, will be referred to as a *generalized global rule of a CA*. Since such functions will be frequently used throughout this paper, for the sake of simplicity, we will refer to them as *global rules* or *rules*. In this paper, a CA is identified with its global rule. Therefore, a CA is technically a function, and by referring to a CA we refer to its global rule and vice versa. Note that a given local rule f_A uniquely defines a global rule A, but the opposite is not true when the radius is not fixed. For a given rule A, there may exist many different local rules defining it.

The set $\mathcal{A}_r$, where $r \in \mathbb{N}$ is the radius, denotes the set of all CAs that can be expressed using local rules with radius r. All of the CAs in $\mathcal{A}_1$ are referred to as elementary CAs (ECAs). This class is one of the most commonly studied classes of CAs [25]. Fact 1 shows two important properties of the sets $\mathcal{A}_r$.

Fact 1: For any $r \geq 0$, $\mathcal{A}_r \subset \mathcal{A}_{r+1}$, and $|\mathcal{A}_r| = 2^{2^{2r+1}}$.

Let A be a CA, $X \in \{0,1\}^M$ for some M and $T \in \mathbb{N}^+$. The finite sequence of vectors given by:

$$\left(X, A(X), A^2(X), \ldots, A^{T-1}(X)\right)$$

where A^t denotes the t-th application of rule A, will be referred to as a *space-time diagram* covering T time stamps. Each of its elements will be referred to as a *configuration* of the CA A, while the first element will be referred to as the *initial configuration*. For any $t = 0, 1, \ldots, T-1$ and $m = 1, \ldots, M$, $A^t(X)[m]$ denotes the state of the mth cell in the t-th row of the space-time diagram.

Example 1: We consider the four ECAs defined by local rules 40, 56, 150, and 110. Fig. 1 depicts their corresponding space-time diagrams. All of them start from the same, random initial configuration containing 69 cells. By convention, the space-time diagrams are visualized as bitmaps in which every row corresponds to a configuration at a specific time stamp. Hence, the first row in the image is the initial configuration. Furthermore, state 1 is visualized as a black pixel, while a white pixel corresponds to state 0.

These four ECAs exemplify the behavior that can be found among 1-D two-state CAs. More precisely, Wolfram conjectures that there are four behavioral classes [25]. After a few time stamps, ECA 40 evolves to a homogeneous configuration (class I), ECA 56 reaches a periodic configuration (class II), ECA 150 behaves chaotically (class III), while ECA 110 displays the complex behavior (class IV).

III. Problem Statement

In this section, we introduce the identification problem. Our formulation is based on the concept of an incomplete observation of a space-time diagram, i.e., it contains only incomplete information on the states of the underlying CA.

Let $I \in \{0, 1, ?\}^{N\times M}$. We interpret I as an array containing symbols from the set $\{0, 1, ?\}$. The role of the symbol "?" is to indicate that information on the actual state of a cell is lacking. Additionally, let the first row $I[1]$ be completely observed, i.e., $I[1] \in \{0,1\}^M$. We will refer to such an array I as an *observation*. An observation $I \in \{0,1\}^{N\times M}$, i.e., an observation that does not contain the symbol "?," will be called *spatially complete*.

Note that the assumption of a completely observed first row is crucial for the construction of the presented algorithm and cannot be relaxed easily. Yet, in many practical applications there might be natural means of controlling the initial state, for example, by repeating the laboratory experiments multiple times, thus this assumption can be easily met.

Example 2: We will visualize the observations as space-time diagrams where the cells of the configurations are colored white if their state is 0 and black when it is 1, while gray is used to color cells whose state is unknown. Following this convention, Fig. 2(a) shows a complete observation, in this case, a space-time diagram of ECA 57. In Fig. 2(b), some of the states in the complete space-time diagram of ECA 57 are changed to "?," resulting in a spatially incomplete observation. Finally, Fig. 2(c) depicts an observation, where some of the states are changed to "?" and some configurations are

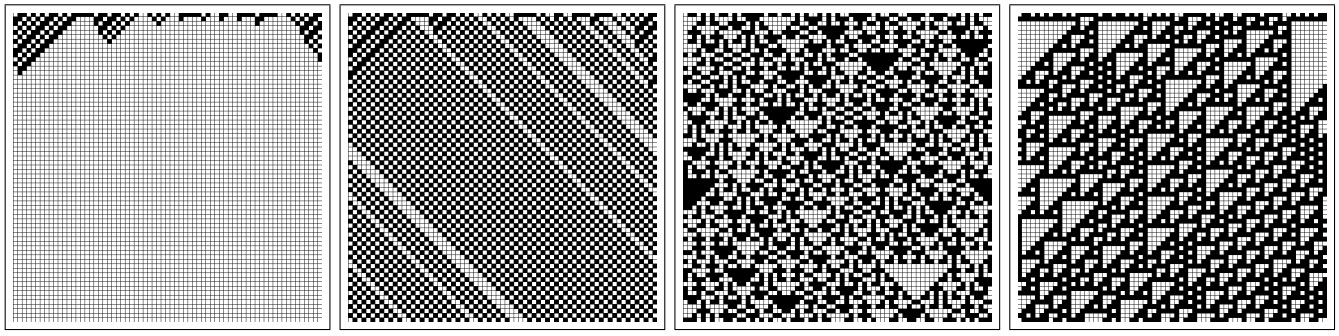


Fig. 1. Space-time diagrams of ECAs, containing 69 cells and 69 time stamps each, illustrating the behavioral classes that can be found among ECAs. (a) ECA 40 (class I). (b) ECA 56 (class II). (c) ECA 150 (class III). (d) ECA 110 (class IV).

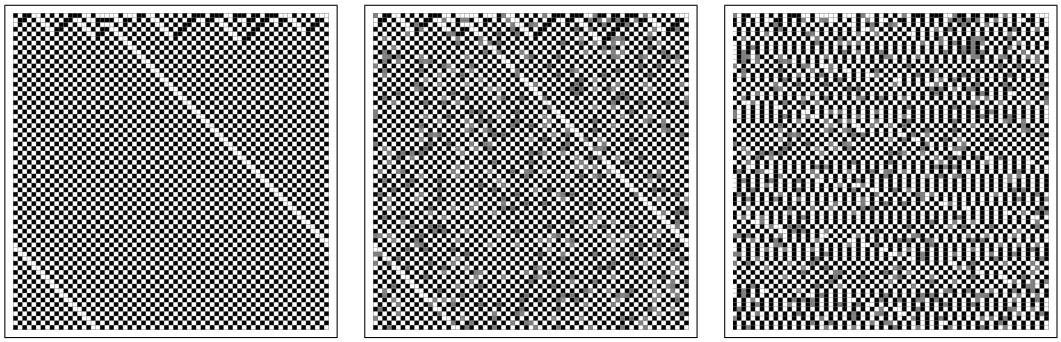


Fig. 2. Observations of ECA 57 in (a) complete, (b) spatially incomplete, and (c) spatially and temporally incomplete form.

omitted from the complete space-time diagram, resulting in a temporally incomplete observation.

Let I be an observation. The number of completely observed states is defined as $C(I) = \#\{I[n, m] \neq ?\}$. In our setting, it holds that $C(I) \geq M$, due to the assumption that $I[1] \in \{0, 1\}^M$. With a given observation I, we associate the set $\mathrm{com}(I)$ of all of the spatially complete observations I' satisfying $I'[n, m] = I[n, m]$ for all n, m such that $I[n, m] \neq ?$.

We will say that a CA A *fits an observation* I if there exists an $I' \in \mathrm{com}(I)$ and an increasing sequence of positive natural numbers $\sigma = (\sigma_i)_{i=1}^{N-1}$ such that for any $n \in \{1, 2, \ldots, N-1\}$

$$A^{\sigma_n}\big(I'[1]\big) = I'[n+1]. \tag{2}$$

The meaning of a CA A fitting an observation I, is that the rows of I correspond to configurations of A at some time stamps, which are given by σ. The first row corresponds to the initial configuration. Entries with the symbol "?" correspond to unknown states. Examining I', which is easy to obtain once A and σ are found, allows to uncover these unknown states.

Proposition 1: A CA A fits an observation I if and only if there exist an $I' \in \mathrm{com}(I)$ and a sequence of natural numbers $\gamma = (\gamma_i)_{i=1}^{N-1}$ such that for any $n \in \{1, 2, \ldots, N-1\}$

$$A^{1+\gamma_n}(I'[n]) = I'[n+1]. \tag{3}$$

The sequence $\sigma = (\sigma_i)_{i=1}^{N-1}$ in (2) corresponds to the time stamps in the CA evolution (which are assigned to the rows of the observation), while the sequence $\gamma = (\gamma_i)_{i=1}^{N-1}$ in Proposition 1 refers to the time gaps, i.e., the number of missing time stamps between two consecutive rows in the observed diagram. Obviously, $\sigma_n = \sum_{i=1}^{n}(1 + \gamma_i)$.

Example 3: Consider the following space-time diagram D. It shows the evolution of ECA 150 from a single black cell. For the sake of readability, rows have been labeled with the corresponding time stamps.

0	0	0	0	1	0	0	0	0	$t = 0$
0	0	0	1	1	1	0	0	0	$t = 1$
0	0	1	0	1	0	1	0	0	$t = 2$
0	1	1	0	1	0	1	1	0	$t = 3$
1	0	0	0	1	0	0	0	1	$t = 4$

Let the observation I be given by

0	0	0	0	1	0	0	0	0	$t = 0$
0	1	1	?	?	0	1	1	0	$t = ?$
1	?	0	0	?	0	0	0	?	$t = ?$

We see that the space-time diagram D and observation I share the same initial configuration, i.e., $D[1] = I[1]$. Now, let $I' \in \mathrm{com}(I)$ be given by

0	0	0	0	1	0	0	0	0	$t = 0$
0	1	1	0	1	0	1	1	0	$t = ?$
1	0	0	0	1	0	0	0	1	$t = ?$

It holds that $D[4] = I'[2]$ and $D[5] = I'[3]$. We know that $A(D[4]) = D[5]$, since D is the space-time diagram of a CA A, and so $A(I'[2]) = I'[3]$. Additionally, it holds that $A^3(D[1]) = D[4] = I'[2]$. Since $I'[1] = D[1]$, it holds that

$A^3(I'[1]) = I'[2]$. According to Proposition 1, this means that CA A fits the observation I.

In practice, we want to use multiple observations to complete the identification task. Therefore, we will consider a finite set of observations $\mathcal{I}$. We will say that rule A fits the observation set $\mathcal{I}$ if it fits all of the observations contained in this set. We will write $C(\mathcal{I})$ to express the number of observed states in all of the observations belonging to $\mathcal{I}$, i.e., $C(\mathcal{I}) = \sum_{I\in\mathcal{I}} C(I)$. Moreover, we will use $M_{\mathcal{I}}$ to denote the total number of columns in the observations belonging to this observation set, i.e., $M_{\mathcal{I}} = \sum_{I\in\mathcal{I}} M_I$, where M_I is the number of columns in observation I. We will denote the number of rows N_I in an observation $I \in \mathcal{I}$ by N_I. For an observation set $\mathcal{I}$, the set $\mathcal{R}(\mathcal{I})$ denotes the set of all CAs fitting the observation set $\mathcal{I}$. With the above definitions, we can restate the identification problem as finding all (or some) of the elements of the set $\mathcal{R}(\mathcal{I})$. In this paper, we will restrict this formulation of the identification problem to finding at least one of the elements of the set $\mathcal{R}(\mathcal{I})$.

The following fact will be used in the design of the identification algorithm in order to reduce the computational burden by considering only subsets of the observation set for the calculation of the error, as such enabling a fast method for estimating the values of the fitness function (this approach is described in detail in Section V-B). Informally, this fact can be understood by considering the set $\mathcal{I}$ as the set of conditions that a rule needs to meet. Having fewer conditions, it becomes more likely to find solutions. Yet, no solutions are omitted by relaxing the conditions.

Fact 2: Let $\mathcal{I}$ and $\mathcal{I}'$ be the observation sets such that $\mathcal{I}' \subseteq \mathcal{I}$. Then $\mathcal{R}(\mathcal{I}) \subseteq \mathcal{R}(\mathcal{I}')$.

We consider only finite observation sets, therefore, we know that for every observation set $\mathcal{I}$ there exists a $\Gamma > 0$ which is a common upper bound for all of the time gaps in all observations belonging to $\mathcal{I}$. In other words, if a solution to the identification problem exists, and γ^I is the sequence of time gaps of observation $I \in \mathcal{I}$, then $0 \le \gamma_n^I < \Gamma$, for every $I \in \mathcal{I}$ and $n = 1, \ldots, N_I - 1$. In the remainder, it is assumed that the upper bound Γ is known. Hence, we consider the case of *bounded time gaps*.

IV. Formulating the Identification Problem as Global Optimization Problem

In this section, we will reformulate the CA identification problem as a global optimization problem. Let $a, b \in \{0, 1, ?\}^M$. We define the disagreement between the vectors a and b as

$$\mathrm{dis}(a, b) = \sum_{i:a_i,b_i\in\{0,1\}} |a_i - b_i|. \tag{4}$$

If there is no i such that $a_i \neq ?$ and $b_i \neq ?$, then $\mathrm{dis}(a, b) = 0$. Therefore, $\mathrm{dis}(a, b) = 0$ does not imply $a = b$. Obviously, $\mathrm{dis}(a, b) = \mathrm{dis}(b, a)$. Note that dis is not a distance function as the triangle inequality is not fulfilled.

Let $\gamma = (\gamma_i)_{i=1}^{N-1}$ be a sequence of natural numbers, and let A be a CA rule. The observation $\bar{I}_\gamma^A$ defined as

$$\bar{I}_\gamma^A[n, m] = \begin{cases} I[n, m], & \text{if } I[n, m] \neq ? \\ A^{1+\gamma_{n-1}}\left(\bar{I}_\gamma^A[n-1]\right)[m], & \text{otherwise} \end{cases}$$

will be referred to as the A-completion of observation I with time gaps γ. Note that any observation I satisfies $I[1] = \bar{I}_\gamma^A[1]$ for any A and γ. Moreover, $\bar{I}_\gamma^A \in \mathrm{com}(I)$.

Intuitively, the meaning of $\bar{I}_\gamma^A$ is as follows. If I is an observation, then the spatially complete observation $\bar{I}_\gamma^A$ agrees with I on all positions occupied by values 0 or 1 in I. The missing state values, denoted by "?" in I, are found by evaluating A, taking into account the time gaps defined by the sequence γ.

Assuming that for every $I \in \mathcal{I}$ the sequence of natural numbers $\gamma^I = (\gamma_i^I)_{i=1}^{N_I-1}$ represents the time gaps in I and $\gamma = (\gamma^I)_{I\in\mathcal{I}}$, the error $\widetilde{E}_{\mathcal{I}}$ is defined as

$$\widetilde{E}_{\mathcal{I}}(A, \gamma) = \sum_{I\in\mathcal{I}} \sum_{n=1}^{N_I-1} \mathrm{dis}\left(A^{1+\gamma_n^I}\left(\bar{I}_{\gamma^I}^A[n]\right), \bar{I}_{\gamma^I}^A[n+1]\right). \tag{5}$$

The identification problem can be expressed mathematically as the minimization of the error $\widetilde{E}$.

As we only consider the case where an upper bound for the time gaps is known, we can define the error measure $\widetilde{E}_{\mathcal{I}}$ independently of the selection of the time gaps γ as

$$\widetilde{E}_{\mathcal{I}}(A) = \min_{\gamma} \widetilde{E}_{\mathcal{I}}(A, \gamma) \tag{6}$$

where the minimum only covers γ satisfying $0 \le \gamma_i^I < \Gamma$, $I \in \mathcal{I}$, $i = 1, \ldots, N_I - 1$.

Note that the minimum in (6) always exists, since there is only a finite, yet huge, number of possibilities for γ. Additionally, note that for a spatially complete observation I, the choice of γ_i^I is independent of the choice of γ_j^I for any $i \neq j$, and for observations I and J, the choice of γ^I is independent of the choice of γ^J. Consequently, to find the value of $\widetilde{E}_{\mathcal{I}}$ in the case of a spatially complete observation set, we need to examine at most $\sum_{I\in\mathcal{I}} \Gamma\,(N_I - 1)$ sequences of time gaps.

In the case of spatially incomplete observations, the choice of $\gamma_{n_i}^I$ is not independent of the choice of $\gamma_{n_j}^I$ for $n_i < n_j$. This is due to the definition of A-completion of an observation, where the choice of the missing state values in the n_1th row may influence the choices in the following rows. Thus, in order to find the exact value of the error measure, we should examine all of the Γ^{N_I-1} possibilities, which implies a substantial computational burden. For that reason, instead of an exact value of the error, we compute an estimate by treating the time stamps as in the spatially complete case, i.e., we ignore the dependencies between the rows and try to select values of γ by analyzing the consecutive pairs of rows in the observation. The only difference, as compared to the spatially complete case, is that if for a given n, several values for γ_n^I lead to the same, minimal value of the error, one of those candidates is selected randomly. In such an approach, we might overestimate the error, i.e., the calculated value can never be lower than the actual error. Additionally, since the procedure

is stochastic, for a given CA that is a solution to the identification problem, recalculating the approximate error measure multiple times and taking the minimum of all of the obtained results increases the probability of finding the exact value.

This approach of estimating the error turned out be sufficient for cases where the number of completely observed states $C(\mathcal{I})$ is relatively high, meaning that the spatial incompleteness is limited. When $C(\mathcal{I})$ gets significantly low compared to the total number of entries in observations, the complexity of the problem largely depends on the structure and origin of the observations. In order to improve the performance in such cases, we implement a filtering technique that eliminates rows of observations for which the number of completely observed states is smaller than or equal to a selected threshold $\tau^* \geq 0$. If the nth row is eliminated, the maximal time gap allowed in between the $(n-1)$th and $(n+1)$th row becomes $2\,\Gamma$.

V. Evolutionary Algorithm

In this section, we describe the solution of the identification problem, for which an evolutionary algorithm based on a GA [26] will be used. GAs constitute a well-known optimization method used in many domains of application, including earlier research on CAs. For example, GAs have been used in the construction of CA-based random number generators [27], for finding CAs capable of image reconstruction [28] and for retrieving CA-based population models of competing individuals [29].

For the sake of reproducibility, we formally define the GA by clearly giving the individuals' representation, structure of the population, a fitness function, the genetic operators, namely, the selection procedure for reproduction, and the cross-over and mutation operators, and finally the halting conditions.

A. Individuals and Populations

Here, the individuals that make up the population are CAs, encoded through the LUT of their local rule, which is possible since the LUT of any CA $A \in \mathcal{A}_r$ can be represented as a bit-string of length 2^{2r+1} (see Section II).

The population is a collection of local rules with different radii between r_* and r^*, since the radius of the desired solution might not be known in practice and one of the goals is to detect it. Note that as a consequence of Fact 1, we might opt to consider populations of local rules with radius r^* only, but by allowing diverse radii, populations can evolve quicker and produce simpler solutions, i.e., local rules with smaller radii.

The number of CAs in a population is denoted by $P > 0$. The symbol $\mathcal{P}^n$, where $n = 1, 2, \ldots$, denotes the population of the nth generation of the GA. Population $\mathcal{P}^1$ is the initial population, and is constructed by randomly selecting P bit-strings. Let $\mathcal{P}^1 = \{L_1^1, \ldots, L_P^1\}$, where L_i^1 is the ith individual in the initial population, then $|L_i^1| = 2^{2r_i+1}$ for some $r_i \in \{r_*, \ldots, r^*\}$. This radius r_i can change as the GA evolves. Populations $\mathcal{P}^n$ for $n > 1$ are evolved by applying the genetic operators described in the remainder of this section. Individuals belonging to the nth population are denoted by L_i^n, for $i = 1, \ldots, P$.

B. Fitness Function

The fitness function is directly related to the error measure $\widetilde{E}_{\mathcal{I}}$ defined by (6). Let $L \in \{0,1\}^{2^{2r+1}}$ be the LUT of some local rule that defines a CA A. Then $\mathrm{fit}_{\mathcal{I}}(L)$ denotes the fitness of A, and is defined as

$$\mathrm{fit}_{\mathcal{I}}(L) = C(\mathcal{I}) - M_{\mathcal{I}} - \widetilde{E}_{\mathcal{I}}(A). \tag{7}$$

The fitness function takes integer values from 0 up to $C(\mathcal{I}) - M_{\mathcal{I}}$ which is the total number of completely observed states excluding the states in all of the initial configurations. Therefore, there are only finitely many values of the fitness function. The goal of the GA is to maximize fitness, and a CA leading to a maximal fitness is a solution of the identification problem.

Initial experiments have shown that the fitness function defined by (7) is effective. Yet, the computing time for finding its values is unacceptable when large observation sets are considered, since the cost of computing an exact value is linear in the size of the observation set. An approximation algorithm is used to overcome this issue for a large observation set. During the evolution of the nth population, we estimate the value of $\mathrm{fit}_{\mathcal{I}}$ by using $\mathrm{fit}_{\mathcal{I}_n}$, where $\mathcal{I}_n \subset \mathcal{I}$ is a nonempty subset. Such an approach is justified by Fact 2, which assures that the solution set is not being reduced. The set $\mathcal{I}_1$ is a subset of $0 < s \leq |\mathcal{I}|$ randomly selected observations of $\mathcal{I}$, while the set $\mathcal{I}_{n+1}$, for $n \geq 1$, is built from $\mathcal{I}_n$ by replacing one of its observations denoted by I_r, with a new selection from $\mathcal{I} \backslash \mathcal{I}_n$ denoted by I_a. There are two scenarios for selecting I_r and I_a, depending on the contents of $\mathcal{I}_n$, which are presented below.

Let $B(n)$ yield the fittest individual from the nth population, that is,

$$B(n) = \arg\max_{L \in \mathcal{P}^n} \mathrm{fit}_{\mathcal{I}_n}(L). \tag{8}$$

If multiple choices of $B(n)$ are possible, we pick one at random. As described in more detail in Section V-H, for such an individual, the fitness $\mathrm{fit}_{\mathcal{I}}$ over the entire set $\mathcal{I}$ is recalculated verifying whether the halting condition is satisfied at every iteration of the GA. Consequently, we have access to the values $\widetilde{E}_I(B(n))$ for any $I \in \mathcal{I}$, and thus we can easily find $I^* \in \mathcal{I}$ such that $\widetilde{E}_{I^*}(B(n)) \geq \widetilde{E}_I(B(n))$ for any $I \in \mathcal{I}$.

If $I^* \in \mathcal{I}_n$, then I_r is selected randomly from $\mathcal{I}_n$ in such a way that $I_r \neq I^*$ and I_a is also selected randomly from $\mathcal{I} \backslash \mathcal{I}_n$. On the other hand, if $I^* \notin \mathcal{I}_n$, then $I_a = I^*$ and I_r is selected randomly from $\mathcal{I}_n$. Formally, the goal of this procedure is to assure that the observation resulting in the highest error I^* is included in $\mathcal{I}_{n+1}$ and that exactly one observation is replaced at every generation, i.e., $|\mathcal{I}_n \cap \mathcal{I}_{n+1}| = s - 1$.

Let $F(n)$ denote the highest fitness observed among the individuals that had the highest values of fitness estimation during the GA evolution up to the nth generation. Let $n \in \mathbb{N}$, then $F(n)$ is defined as $F(n) = \max_{i \in \{1,\ldots,n\}} \mathrm{fit}_{\mathcal{I}}(B(i))$.

Obviously $F(n) \leq F(n+1)$. In addition to this maximal fitness F, we also define the age of this value as the number of GA iterations during which the maximal fitness did not change. Formally, a value $F(n)$ has an age $a(n) > 0$ if and only if it holds that

$$F(n - a(n) - 1) < F(n - a(n)) = \cdots = F(n-1) = F(n).$$

Algorithm 1: Up-Scaling of the LUT

Input: LUT given by variable L.
Output: Up-scaled LUT stored in variable $L^{\uparrow}$.

```
for i = 1, ..., 2^{2r+1} do
    L↑[2i] ← L[i];
    L↑[2i − 1] ← L[i];
    L↑[2i − 1 + 2^{2r+2}] ← L[i];
    L↑[2i + 2^{2r+2}] ← L[i];
```

Algorithm 2: Down-Scaling of the LUT

Input: LUT given by variable L.
Output: Down-scaled LUT stored in variable $L_{\downarrow}$.

```
Initialize C[i] ← 0 for i = 1, ..., 2^{2r−1};
for i = 0, ..., 2^{2r+1} − 1 do
    j ← 1 + (i/2 mod 2^{2r−1});
    C[j] ← C[j] + L[i + 1];
for i = 1, ..., 2^{2r−1} do
    if C[i] > 2 then
        L↓[i] ← 1
    else
        L↓[i] ← 0
```

C. Selection

We use a standard, random selection method. The probability of selecting a given individual is proportional to its fitness. This is feasible, since the fitness function introduced in Section V-B is bounded. The selection process is repeated with replacement, so that each of the individuals can be selected multiple times.

D. LUT Rescaling

Before describing the cross-over operator, we introduce a rescaling operation in order to be able to define cross-over of LUTs with different radii. Given Fact 1, each CA defined by a local rule with neighborhood radius r, can also be defined by a local rule with neighborhood radius $r+1$. A simple method for "converting" the LUT of a given local rule f, with radius r, to the LUT of its corresponding local rule $f^{\uparrow}$, with radius $r+1$, follows from the fact $f^{\uparrow}(s_1, \ldots, s_{2(r+1)+1}) = f(s_2, \ldots, s_{2r+2})$, for all $s_1, \ldots, s_{2(r+1)+1} \in \{0, 1\}$. Note that both f and $f^{\uparrow}$ define the same CA. The operation of increasing the radius by one will be referred to as up-scaling by one. Let $L \in \{0,1\}^{2^{2r+1}}$ be the LUT of f, then the LUT of $f^{\uparrow}$, denoted as $L^{\uparrow} \in \{0,1\}^{2^{2(r+1)+1}}$, can be constructed from L using Algorithm 1.

Similarly to the up-scaling operation, we define the inverse operation. Let f be a local rule with neighborhood radius r, then the down-scaled local rule $f_{\downarrow}$, with radius $r-1$, is defined as

$$f_{\downarrow}(s_1, \ldots, s_{2r-1}) = \left\lfloor \frac{1}{4} \sum_{a=0}^{1} \sum_{b=0}^{1} f(a, s_1, \ldots, s_{2r-1}, b) \right\rceil$$

where $\lfloor \cdot \rceil$ indicates the rounding to the nearest integer (we assume $[0.5] = 0$). The method of computing the LUT of the down-scaled rule is presented in Algorithm 2. Let $L \in \{0,1\}^{2^{2r+1}}$ be the LUT of some local rule f, then this algorithm constructs the LUT $L_{\downarrow} \in \{0,1\}^{2^{2r-1}}$ corresponding to the local rule $f_{\downarrow}$. It is easy to see that, in general, f and $f_{\downarrow}$ define different CAs. Informally, $f_{\downarrow}$ should be understood as the closest approximation of f with a smaller radius.

Using the up-scaling and down-scaling operations, we can define a general rescaling operation from radius r to r', by applying up-scaling (when $r' > r$) or down-scaling (when $r' < r$) operations multiple times.

E. Cross-Over

To produce offspring, two parents are selected according to the selection procedure outlined in Section V-C. If the radii of the selected parents are equal, we use uniform cross-over, i.e., the result is a vector L_c with values that are randomly selected from the parents L_1 and L_2 so that $\mathbb{P}(L_c[i] = L_1[i]) = \mathbb{P}(L_c[i] = L_2[i]) = 0.5$. If $r_1 \neq r_2$, then the LUTs are rescaled before applying crossover. Assuming that $r_1 < r_2$, the radius r of the resulting rule is selected randomly from the set $\{r_1, r_1+1, \ldots, r_2\}$, after which the parents are rescaled to the radius r and the cross-over operator is applied.

F. Mutation Operator

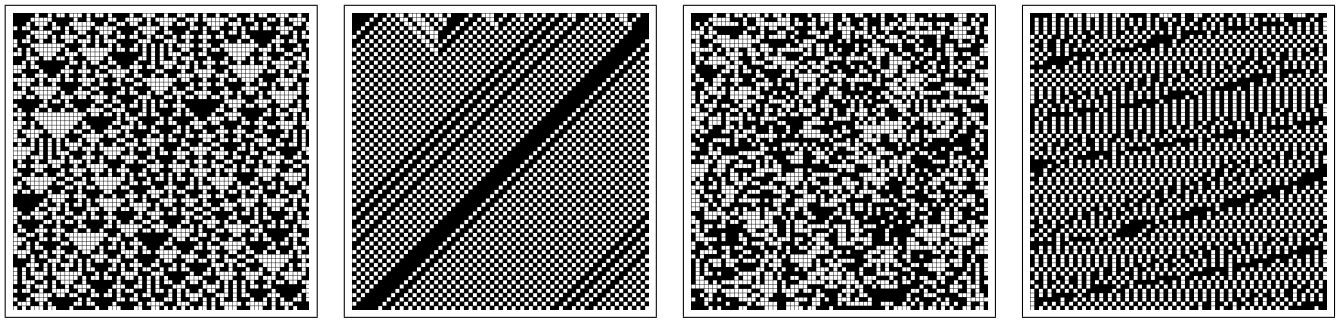
Finally, the offspring individuals are mutated. Here, three types of mutations are used: 1) bit flipping; 2) decrease of radius; and 3) increase of radius. The latter two mutations simply rely on the down-scaling or up-scaling operations outlined in Algorithms 1 and 2, and introduce the diversity in the length of the individuals, while bit flipping randomly flips selected bits in the LUT of the individual.

Mutations are applied in a well-defined order. First, with probability p_u, up-scaling is applied. After that, we flip randomly selected bits. The bit-flip mutation is applied bit-by-bit independently. The probability p_f of mutating a bit is defined as follows:

$$p_f(n) = e^{-\alpha (R - a(n))} \tag{9}$$

for some $\alpha > 0$ and $R \in \mathbb{N}$. The role of the parameter R is described in more detail in Section V-G. For now, we may assume that $a(n)$ will never be higher than R and thus, $p_f(n) \leq 1$. This formula is motivated by the fact that when the age $a(n)$ is low, the probability of mutation should also be relatively low in order to give the algorithm a chance to fine-tune the solution. If this fine-tuning fails, the age $a(n)$ increases and the population tends to freeze near a local optimum. In order to escape from it, we then apply mutation with a higher probability. Such a procedure is partially effective due to the elite survival procedure introduced in Section V-G. Finally, after applying bit-flipping, we apply down-scaling with probability p_d.

This order of application is chosen because the number of possibilities to be affected by bit-flipping increases by first applying up-scaling. Besides, performing the down-scaling at the end of the sequence of mutation operators allows to evolve simpler rules.



(a) (b) (c) (d)

Fig. 3. Space-time diagrams of (a) ECA 150 and (b) ECA 184 containing 69 cells and 69 time stamps each, and (c) and (d) corresponding temporally incomplete observations with random time gaps of at most ten time steps.

G. Elite Survival and Population Reinitiation

After evolving a new population, an elite survival procedure is applied, which has shown to be a prerequisite to reach the convergence. The procedure is implemented by selecting the $P_E \ll P$ fittest individuals from the previous population and letting them replace randomly selected individuals in the newly evolved one.

Including this elite survival procedure dramatically increases the performance of the algorithm, though there are cases where such an approach causes the population to progress toward a local optimum. Our experiments showed that the best way to overcome this is to apply a reinitiation procedure. If for a given n it holds that the age $a(n) = R$, then we reinitiate the algorithm by replacing the current population with a new, randomly selected set of CAs. In other words, we do not allow a situation where the age of the best individual evolved so far is higher than R. Such an approach might seem to contradict the evolutionary nature of the algorithm, but the use of a dynamic mutation probability $p_f(n)$ and an elitist survival procedure relates it to the evolution of multiple separated genetic islands [30], out of which the one with the fittest individual is selected after a predefined number of iterations.

H. Halting Conditions

The GA evolves until a CA that fits the observation set is discovered or a predefined number of GA iterations passes.

As mentioned in Section V-B, the fitness $\mathrm{fit}_{\mathcal{I}}$ is approximated by $\mathrm{fit}_{\mathcal{I}'}$ for some $\mathcal{I}' \subset \mathcal{I}$, which is effective for selection, but cannot be used to verify whether the halting condition has been met. Therefore, for the individual A with the highest value of $\mathrm{fit}_{\mathcal{I}'}(A)$, we also calculate $\mathrm{fit}_{\mathcal{I}}(A)$. The algorithm stops as a CA A is found such that $\widetilde{E}_{\mathcal{I}}(A) = 0$. The knowledge of $\mathrm{fit}_{\mathcal{I}}(A)$ at every iteration is exploited to build the subsets $\mathcal{I}_n \subset \mathcal{I}$ used for estimating $\mathrm{fit}_{\mathcal{I}}$ (see Section V-B).

I. Examples: Identification of ECAs 150 and 184

Having described the GA for solving the identification problem, we now present an example to illustrate its behavior. An in-depth study of its performance and limitations can be found in Section VI.

TABLE II
VALUES OF THE DESIGN PARAMETERS USED FOR THE IDENTIFICATION OF ECAS 150 AND 184

parameter	value	description
r_*	2	minimal rule radius
r^*	5	maximal rule radius
P	192	number of individuals in population
P_E	24	elite size
Γ	10	upper bound for the time gaps
S	69	number of rows / columns in each observation
K	64	number of observations
s	8	number of samples for fitness approximation
α	0.01	parameter for the dynamic mutation probability
p_d	0.01	down-scale mutation probability
p_u	0.01	up-scale mutation probability
R	250	maximal age of fittest individual for re-initiation
G	9×10^5	maximum number of GA iterations per run

For this purpose, a GA with design parameters as listed in Table II was used. These values were chosen on the basis of preliminary experiments, while the choice of r_* was motivated by the fact that observation sets resulting from ECAs will be used. If $r_* = 1$ would be used, the GA would be able to rapidly explore the entire space of ECAs (consisting of only 256 candidates), which makes that more interesting characteristics of the GA might be missed.

Using these parameters two experiments were conducted, further referred to as R1 and R2. In R1, the considered observation set consisted of observations of ECA 150, while ECA 184 was used in R2. The choice of these two ECAs was motivated by the fact that ECA 150 shows a strong sensitive dependence on the initial configuration, while ECA 184 (known as the traffic rule) results in an orderly behavior [31]. These two types of behavior can be seen in the space-time diagrams in Fig. 3(a) and (b). In this figure, also the corresponding temporally incomplete observations with random time gaps of at most ten time steps are shown [Fig. 3(c) and (d)]. Such temporally incomplete observations with random time gaps, starting from 64 different initial configurations, were used by the GA.

In both experiments, the GA quickly found a solution. In the case of experiment R1, it took 26 iterations, while 23 iterations were needed in experiment R2. In Fig. 4, the maximum, average, and minimum fitness of the population is shown over time for both the experiments. Note that for the sake of readability the fitness values were normalized to the interval [0, 100].

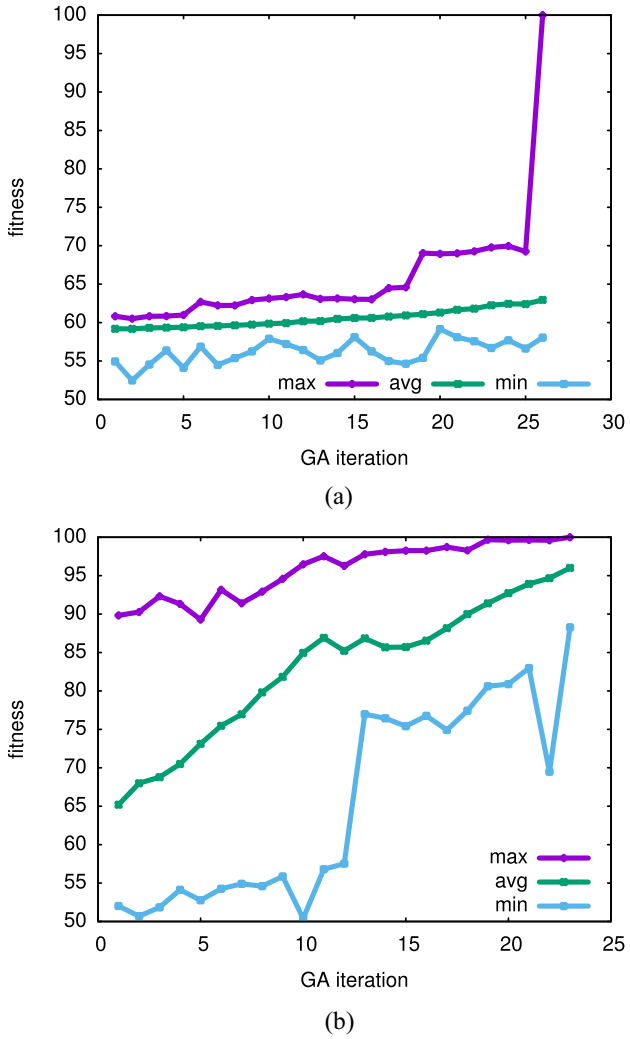


Fig. 4. Maximum, average, and minimum fitness of the population versus the GA iteration in experiments (a) R1 and (b) R2.

First, let us note that the plotted values correspond to the fitness approximation calculated over a subset of the observation set as described in Section V-B. This explains why the maximum is not strictly increasing, though the algorithm uses an elite survival procedure.

Moreover, as can be seen in the plots, in both experiments the average fitness (almost) constantly increased as the populations evolved. In R1, the growth was stable but slow, while in the case of R2 we see a more rapid increase, especially during the first 10 iterations. In the case of R1, the final solution was found within one significant "jump" of the maximal fitness. This can be attributed to the fact that the behavior of CAs that are very close to the best solution is completely different from the one of ECA 150. In contrast, in R2, we see a rather gentle increase of the maximum fitness toward 100, which can be explained by the fact that there are many CAs resulting in checkerboard-like patterns similar to the ones evolved by ECA 184.

In Fig. 5, we see the space-time diagram of some CAs that were discovered by the GA during the evolution as "current best" candidates for R1, while Fig. 6 displays these for R2.

Interestingly, in both experiments, relatively early in the evolution of the GA was able to concentrate on some of the crucial patterns in the space-time diagrams of the unknown ECAs. In R1, after six iterations, the best individuals already replicated the triangular pattern of ECA 150, even though it is very hard to notice such a pattern in the observations available for the GA [see Fig. 3(c)]. Similarly, in R2, the linear pattern emerged very quickly, yet deciding on the slope of the lines turned out to be a bit more challenging.

VI. Experimental Results

A. Introduction

In this section, the results of a series of computational experiments are presented. The main goal of these experiments was to evaluate the effectiveness of the identification algorithm in specific circumstances, where the observation set is generated by a selected CA. The goal of the GA was to uncover this CA from observations. Five experiments were performed and in each of them a broad class of CAs and corresponding observation sets were considered. In Experiment 1, we evaluated the performance of the GA on spatially complete observation sets originating from the entire class of ECAs, considered as a subset of $\mathcal{A}_2$, i.e., CAs with radius 2. By only considering ECAs, we were able to examine a class of well-known CAs. Yet, considering such a specific subset of $\mathcal{A}_2$ may have had a strong impact on the results. To verify whether this was the case, Experiment 2 was conducted, where we evaluated the performance of the GA on observations originating from a randomly selected set of 350 CAs defined by local rules with radius 2 (but not ECAs). In Experiments 2 and 4, the lengths of the time gaps were random. To verify whether this uniform randomness influences the obtained results, we studied other types of time gaps (constant versus only odd versus only even) in Experiment 3. In Experiment 4, we additionally considered spatially incomplete observations. Finally, Experiment 5 covers the aspect of radius detection, i.e., the ability of the GA to uncover the correct radius of the underlying CA (which is not known upfront in many practical cases). Three cases were studied here, namely, CAs with radius 2, 3, and 4.

Due to the large number of cases in the experiments described in this section, the GA has been implemented using highly optimized code written in C and compiled by the Intel C compiler, utilizing OpenMP for concurrent fitness calculations. The specific executions of the GA were run on a large computing cluster, where each execution was bound to a specific node (no internode communication was needed). Each of the cluster's nodes had a 24-core Xeon E5-2670 CPU and 128 GB of RAM. The source code of this GA implementation along with the build and the execution scripts and technical documentation is available on: github.com/houp/identify.

B. Experiment 1: Identification of ECAs in Case of Spatial Completeness

In this experiment, the observation set $\mathcal{I}$ consists of spatially complete observations only, i.e., for every $I \in \mathcal{I}$, it holds that $I \in \{0, 1\}^{N_I \times M_I}$. In addition, we assume that the number of

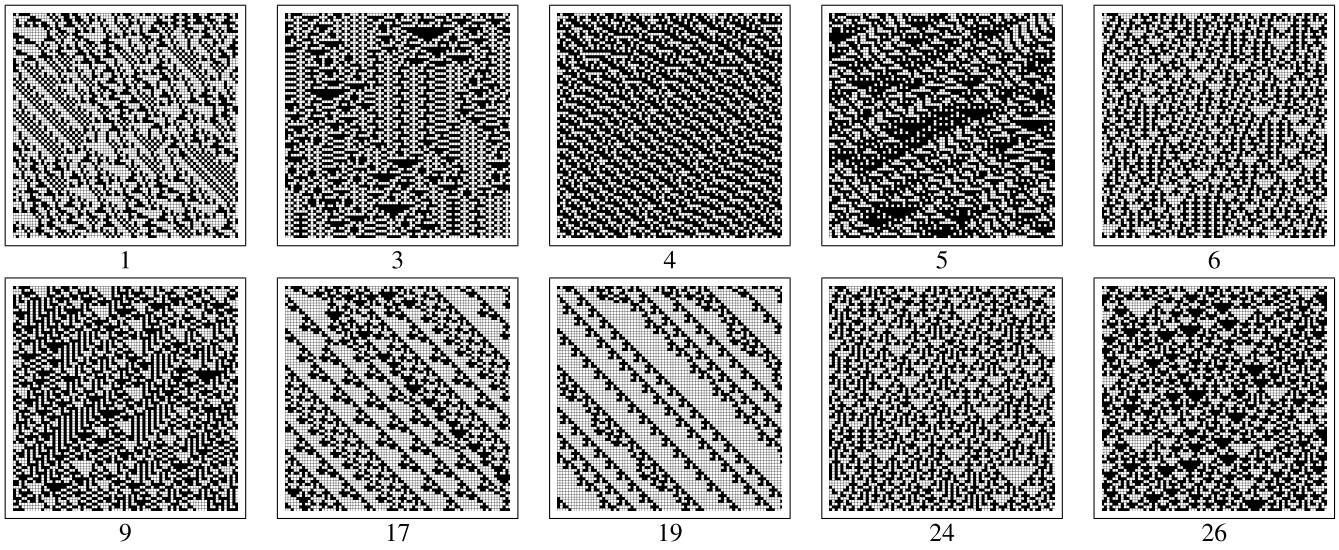


Fig. 5. Current best individuals visualized with their space-time diagram and the corresponding GA iteration number, in R1.

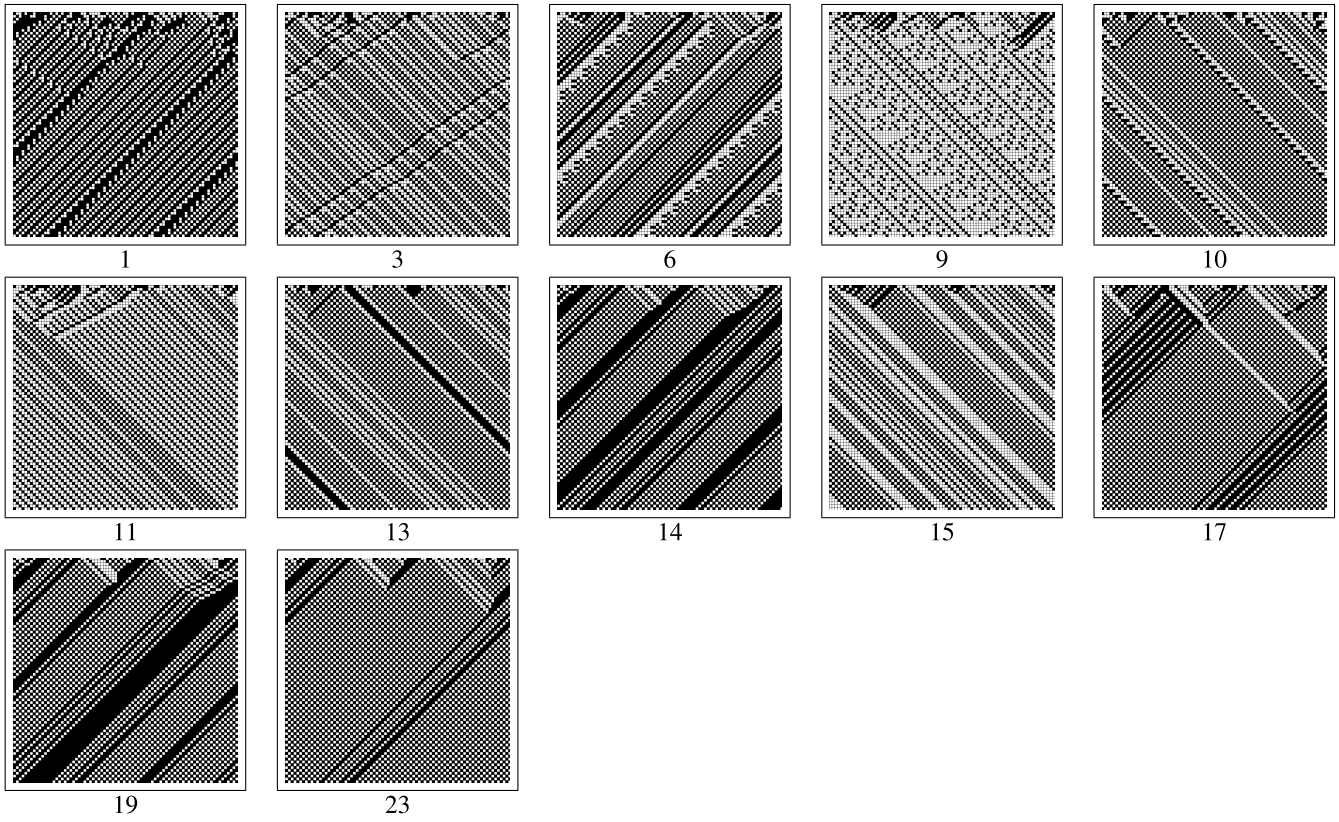


Fig. 6. Current best individuals visualized with their space-time diagram and the corresponding GA iteration number, in R2.

rows is equal to the number columns in all observations ($N_I = M_I = S > 0$).

The goal of the experiment is to measure the efficiency of the identification algorithm. More specifically, we are interested in the number of iterations needed in order to evolve to a solution. We assess the performance of the algorithm using the observation sets generated by ECA rules. Let $\mathcal{I}_A$ denote the observation set obtained by observing the behavior of an ECA A. It is assumed that for each A, the number of observations is the same, i.e., $|\mathcal{I}_A| = K$, for $K > 0$, and that the observations of different ECAs are evolved from the same set of random initial conditions. Observation set $\mathcal{I}_A$ contains observations with random time gaps bounded by $\Gamma > 0$, which are chosen for every row independently.

The values of the design parameters used in this experiment are the same as in Table II with the exception of: $P = 32$, $P_E = 8$, $\alpha = 0.0.25$, and $R = 100$. Due to the stochastic

TABLE III
EXPERIMENT 1: THE PERFORMANCE OF THE GA EXPRESSED IN TERMS OF THE AVERAGE OF THE MINIMUM, AVERAGE, AND MAXIMUM NUMBER OF ITERATIONS PER BEHAVIORAL CLASS ACCORDING TO (a) WOLFRAM'S CLASSIFICATION SCHEME AND (b) THE NMLE

(a)

class	**avg**($\min_A$)	**avg**(avg_A)	**avg**($\max_A$)
I	38	207	666
II	47	244	915
III	40	296	1264
IV	57	690	2650

(b)

nMLE	**avg**($\min_A$)	**avg**(avg_A)	**avg**($\max_A$)
$-\infty$	44	257	934
0	38	176	665
$> \mathbf{0}$	49	308	1187

nature of the GA, the experiment is repeated $\Omega = 50$ times for each ECA.

The performance of the GA is quantified for each A in terms of the average (avg_A), minimum ($\min_A$), and maximum ($\max_A$) number of GA iterations needed to find the solution. Across all ECAs, the average of the minimum, average, and maximum number of GA iterations was 46, 270, and 1018, respectively.

These results may serve as a reference when interpreting the statistics broken down for specific behavioral classes [Table III(a) and (b)]. In Table III(a), the grouping is done according to Wolfram's classification scheme [25], while the values of the normalized maximum Lyapunov exponent (nMLE) [31]–[33] are used in Table III(b). These tables show that the effort needed for identification is significantly lower than the overall average for ECAs belonging to the least complex classes (Wolfram class I and nMLE $= -\infty$). Additionally, the most complex classes require the highest average number of iterations, which shows that there is a connection between the complexity of a CA's dynamical nature and the performance of the identification algorithm. This connection is further confirmed by the results of a two-sided Mann–Whitney U-test for avg_A grouped according to Wolfram's classes [Table IV(a)] and the nMLE [Table IV(b)]. The p-values confirm that the difference in performance of the GA for the different Wolfram classes and for the considered (intervals of) nMLE values is statistically significant.

In Table V, we present the nMLE and GA statistics of the five ECAs that were the easiest to identify [Table V(a)] and the five ECAs that were the hardest to identify [Table V(b)] according to the values of avg_A. Not surprisingly, among the easiest to identify, we find ECAs 0 and 255, which correspond to the two simplest local rules mapping any neighborhood configuration to the same value. Interestingly, all of the ECAs that were hard to identify have similar relatively high nMLE values. Summing up, the identification algorithm turns out to be effective in all of the cases considered, although the effort of finding a solution differs greatly, depending on the ECA.

C. Experiment 2: Performance on the Class $\mathcal{A}_2$

Experiment 1 concerned ECAs only, considering them as a subclass of $\mathcal{A}_2$, by setting $r_* = 2$ in the GA, so that a measurable effort was needed to find a solution. As a natural continuation of Experiment 1, we repeated the experiment, but instead of using 256 ECAs to generate initial observation sets, we used 350 randomly selected CAs defined by a local rule with radius two, i.e., CAs belonging to the class $\mathcal{A}_2$. The remaining parameters and the experimental setup were the same as in Experiment 1. The goal of this experiment was to verify whether the results of Experiment 1 were affected by the nature of ECAs, which form a very specific subclass of $\mathcal{A}_2$.

The average of the minimum, average, and maximum number of GA iterations required to find a solution in this experiment was, respectively,

$$\text{avg}(\min{}_A) = 57,\ \ \text{avg}\big(\text{avg}_A\big) = 617,\ \ \text{avg}(\max{}_A) = 3040.$$

These results clearly suggest that, on average, ECAs as a subclass of $\mathcal{A}_2$ were easier to identify when compared to other CAs from the class $\mathcal{A}_2$, which may be attributed to the fact that many of the ECAs result in a relatively simple dynamical behavior. On the other hand, although the averages are slightly higher, the order of magnitude is not changed. We believe that these results show that the approach of analyzing ECAs as a subclass of $\mathcal{A}_2$ in Experiment 1 was fully justified. More interestingly, the results presented above are very close to the results for ECAs belonging to class IV

TABLE IV
EXPERIMENT 1: RESULTS OF THE TWO-SIDED MANN–WHITNEY U TEST OF avg_A GROUPED ACCORDING TO (a) WOLFRAM'S CLASSIFICATION SCHEME AND (b) NMLE

(a)

case	U-statistic	p-value
Class I vs. Class II	1858	1.227561e-01
Class I vs. Class III	167	5.016731e-03
Class I vs. Class IV	63	1.565208e-03
Class II vs. Class III	1606.5	3.226133e-03
Class II vs. Class IV	564	2.946143e-04
Class III vs. Class IV	123	9.714852e-02

(b)

case	U-statistic	p-value
nMLE $= -\infty$ vs. nMLE $= 0$	2095	1.859012e-02
nMLE $= -\infty$ vs. nMLE > 0	4671	3.475995e-01
nMLE $= 0$ vs. nMLE > 0	1848.5	5.748794e-05

TABLE V
ECAS REQUIRING THE (a) SMALLEST AND THE (b) LARGEST NUMBER OF ITERATIONS OF THE GA IN EXPERIMENT 1

(a)

ECA	**nMLE**	**min**$_A$	**avg**$_A$	**max**$_A$
255	$-\infty$	1	3	9
0	$-\infty$	4	30	183
221	$-\infty$	25	47	86
207	0	25	55	156
246	0	27	57	112

(b)

ECA	**nMLE**	**min**$_A$	**avg**$_A$	**max**$_A$
137	0.6577	48	1679	8270
193	0.6561	52	1625	6221
110	0.6574	49	1348	4269
25	0.5168	43	1262	5690
124	0.6567	59	1231	4431

TABLE VI
EXPERIMENT 3: THE PERFORMANCE OF THE GA EXPRESSED IN TERMS OF THE AVERAGE OF THE MINIMUM, AVERAGE, AND MAXIMUM NUMBER OF ITERATIONS IN THE CASE OF: 1. EVEN, 2. ODD, AND 3. ARBITRARY TIME GAPS

case	description	avg($\min_A$)	avg(avg_A)	avg($\max_A$)
1	even time gaps	35	158	783
2	odd time gaps	1603	7074	16306
3	arbitrary time gaps	830	4437	13152

of Wolfram's classification scheme [Table III(a)], suggesting that the identification is likely to be more complex when identifying the classes of CAs with higher radius.

D. Experiment 3: Identification of ECAs With Constant Time Gaps

While the time gaps had a random length in Experiment 1, they are kept constant in this experiment. Random time gaps correspond to a lack of temporal synchronization between the observed system and the observations, while the constant time gaps represent a simple form of synchronization of the clocks. Thus, the goal of this experiment is to verify whether such a synchronization makes it easier to solve the identification problem.

Each observation I used in this experiment is constructed by randomly selecting one number $t_I \in \{1, \ldots, \Gamma - 1\}$ and taking every t_Ith row of the original space-time diagram, thus in between every two time stamps of a given observation I exactly t_I time stamps are missing. Neither the numbers t_I nor the fact that the time gaps are equally known by the GA. We consider three cases: 1) even t_I (Table VII); 2) odd t_I (Table VIII); and 3) arbitrary t_I (Table IX). The overall performance of the algorithm across all ECAs and these three cases are presented in Table VI.

As can be inferred from these tables, the performance of the GA differs greatly from that in Experiment 1. Especially in the case of odd t_I, the performance is significantly lower. Moreover, in this case, for ECAs 105 and 150 (which are dynamically equivalent) the algorithm is not able to find a solution in fewer than $G = 9 \times 10^5$ iterations. A similar situation occurs in the case of arbitrary t_I (Table IX), where the effect of odd time gaps, which are present in about half of the cases, played an important role.

The reason for the unidentifiability of ECAs 105 and 150 lies in the specific properties of binary CAs. Let A_{105} and A_{150} be the corresponding global rules of the ECAs in question. It is relatively easy to check that for any configuration X it holds that $A_{105}(X) \neq A_{150}(X)$ and $A_{105}(A_{105}(X)) = A_{150}(A_{150}(X))$. In other words, every second row of a space-time diagram of A_{105} is identical to the corresponding row in the space-time diagram of A_{150} and this holds for any initial configuration. Hence, if the time gaps are odd, both CAs solve the same identification problem. In general, the identification algorithm can handle multiple solutions of the identification problem, but in this particular case, the LUT representations of those two CAs are each other's Boolean complement. This is an important challenge for the crossover operation which is likely to produce offspring weaker than the parents in most of the cases.

TABLE VII
EXPERIMENT 3: THE PERFORMANCE OF THE GA EXPRESSED IN TERMS OF THE AVERAGE OF THE MINIMUM, AVERAGE, AND MAXIMUM NUMBER OF ITERATIONS PER BEHAVIORAL CLASS ACCORDING TO (a) WOLFRAM'S CLASSIFICATION SCHEME AND THE (b) NMLE IN THE CASE OF CONSTANT, EVEN TIME GAPS (CASE 1)

(a)

class	avg($\min_A$)	avg(avg_A)	avg($\max_A$)
I	25	143	728
II	36	156	770
III	32	143	763
IV	41	236	1161

(b)

nMLE	avg($\min_A$)	avg(avg_A)	avg($\max_A$)
$-\infty$	33	141	722
0	32	145	697
> 0	37	171	850

TABLE VIII
EXPERIMENT 3: THE PERFORMANCE OF THE GA EXPRESSED IN TERMS OF THE AVERAGE OF THE MINIMUM, AVERAGE, AND MAXIMUM NUMBER OF ITERATIONS PER BEHAVIORAL CLASS ACCORDING TO (a) WOLFRAM'S CLASSIFICATION SCHEME AND THE (b) NMLE IN THE CASE OF CONSTANT, ODD TIME GAPS (CASE 2)

(a)

class	avg($\min_A$)	avg(avg_A)	avg($\max_A$)
I	32	316	1429
II	44	448	2015
III	16634	69590	148579
IV	134	3370	14535

(b)

nMLE	avg($\min_A$)	avg(avg_A)	avg($\max_A$)
$-\infty$	931	5411	11163
0	37	375	1785
> 0	2517	10324	24327

Interestingly, ECAs 105 and 150 are not the only ECAs with this property. The same happens for ECAs 15 and 240 (nMLE = 0), 23 and 232 (nMLE ≈ 0.001), 43 and 212 (nMLE $= -\infty$), 51 and 204 (nMLE = 0), 77 and 178 (nMLE = 0), 85 and 170 (nMLE = 0), and 113 and 142 (nMLE $= -\infty$). Yet, all of those rules are correctly identified in Experiment 2. We argue that this must be because they are much less sensitive to the initial configuration (in terms of nMLE) than ECAs 105 and 150.

Finally, it can be shown that there does not exist a pair of 1-D binary CAs A and B such that $A(X) \neq B(X)$, $A(A(X)) \neq B(B(X))$, and $A^3(X) = B^3(X)$ for any X, which is clearly reflected in the high performance of the algorithm in the case of even time gaps.

All together, the main result of this experiment is that the nature of the time gaps can greatly affect the performance of identification algorithm, up to the extent that it prevents the algorithm to successfully identify a solution.

E. Experiment 4: Impact of Spatial Incompleteness of Observations

In this experiment, we measure the effect of introducing spatial incompleteness. The previous experiments covered

TABLE IX
EXPERIMENT 3: THE PERFORMANCE OF THE GA EXPRESSED IN TERMS OF THE AVERAGE OF THE MINIMUM, AVERAGE, AND MAXIMUM NUMBER OF ITERATIONS PER BEHAVIORAL CLASS ACCORDING TO (a) WOLFRAM'S CLASSIFICATION SCHEME AND THE (b) NMLE IN THE CASE OF CONSTANT TIME GAPS (CASE 3)

(a)

class	**avg**($\min_A$)	**avg**(avg_A)	**avg**($\max_A$)
I	41	802	3642
II	50	594	2589
III	8353	40465	110388
IV	94	2239	10449

(b)

nMLE	**avg**($\min_A$)	**avg**(avg_A)	**avg**($\max_A$)
$-\infty$	503	2406	7128
0	38	338	1567
$>$ **0**	1285	6981	20608

TABLE X
EXPERIMENT 4: MINIMAL, AVERAGE, AND MAXIMAL PERCENTAGE S_A DENOTING THE MAXIMAL DENSITY OF "?" SYMBOLS IN THE OBSERVATION SET AT WHICH THE IDENTIFICATION IS SUCCESSFUL FOR AT LEAST ONE OF THE 50 RUNS, PER BEHAVIORAL CLASS ACCORDING TO (a) WOLFRAM'S CLASSIFICATION SCHEME AND THE (b) NMLE

(a)

class	**min**(S_A)	**avg**(S_A)	**max**(S_A)
I	89.36 %	98.14 %	100 %
II	4.26 %	74.46 %	100 %
III	36.17 %	58.10 %	80.85 %
IV	29.79 %	55.17 %	65.96 %

(b)

nMLE	**min**(S_A)	**avg**(S_A)	**max**(S_A)
$-\infty$	23.40 %	81.63 %	100 %
0	36.17 %	83.74 %	100 %
$>$ **0**	4.26 %	66.61 %	100 %

only temporal incompleteness, but here we validate how the addition of an additional source of incompleteness affects the performance of the algorithm. The parameters of the identification algorithm are set to the same values as in Experiment 1. For each of the 256 ECAs, we build a set of spatially complete observations. For each of those sets we run the identification algorithm $\Omega = 50$ times. Then, we gradually introduce the spatial incompleteness by changing 2000 randomly selected entries (either 0 or 1) to "?". This process is repeated multiple times until we end up with observations of which only the first row is known. After each step, the identification algorithm is executed $\Omega = 50$ times. Note that we do not execute the identification algorithm after the final step of introducing the "?" symbols everywhere, since it is a trivial case.

For each ECA we measure the percentage S_A denoting the maximal percentage of "?" symbols in the observation set at which the identification is successful for at least one of the 50 runs in fewer than $G = 9 \times 10^5$ GA iterations (Table X). Overall, the average is $\text{avg}(S_A) = 73.96\%$ and ECA 125 turns out to be the most sensitive to the spatial incompleteness, resulting in $S_A = 4.26\%$.

As can be inferred from Table X, ECAs differ when it comes to their value of S_A, which may be understood informally as their tolerance to spatial incompleteness. Interestingly, for most of the class I ECAs, introduction of spatial incompleteness does not impact identifiability. As the CAs get more complex, also the tolerance to spatial incompleteness tends to lower. Yet, for some class II rules, there are visible differences, which are not yet fully understood. To further analyze the obtained results, we applied the two-sided Mann–Whitney U-test on S_A grouped according to Wolfram's classification scheme and the nMLE (Table XI). As in the case of Experiment 1, the difference in performance of the GA for the different Wolfram classes and for the considered (intervals of) nMLE values is statistically significant.

TABLE XI
EXPERIMENT 4: RESULTS OF THE TWO-SIDED MANN–WHITNEY U TEST OF S_A GROUPED ACCORDING TO (a) WOLFRAM'S CLASSIFICATION SCHEME AND THE (b) NMLE

(a)

case	U-statistic	p-value
Class I vs. Class II	4248	1.411803e-11
Class I vs. Class III	624	4.573460e-10
Class I vs. Class IV	336	5.870633e-08
Class II vs. Class III	3976	8.808950e-07
Class II vs. Class IV	2227	3.921596e-05
Class III vs. Class IV	195.5	7.111968e-01

(b)

case	U-statistic	p-value
nMLE $= -\infty$ vs. nMLE $= 0$	1909	1.786718e-01
nMLE $= -\infty$ vs. nMLE > 0	7326.5	9.120496e-08
nMLE $= 0$ vs. nMLE > 0	4397	1.747132e-05

F. Experiment 5: Radius Detection

The goal of this experiment is to verify whether the identification algorithm is able to correctly evolve the radius of an unknown local rule. We study randomly selected CAs from the sets $\mathcal{A}_2$, $\mathcal{A}_3$, and $\mathcal{A}_4$. From each of these sets, 100 CAs are selected. The parameters of the identification algorithm are the same as in Experiment 1, with the exception of the minimal radius, which is now $r_* = 1$, the population size $P = 512$, and size of the elite $P_E = 128$. The latter two settings are motivated by preliminary experiments showing that a larger population size assures a stable behavior of the algorithm if the expected solution is likely to have a higher radius. In contrast to the previous experiment, we adjust the halting condition. In this experiment, the algorithm is stopped when finding a solution or after $G = 10^4$ iterations. Since the CAs are selected randomly, we do not repeat the identification multiple times per CA, as in the previous experiments. The observations generated by the CAs are spatially complete and the time gaps are random.

Among the 300 tested cases, only six runs of the identification algorithm were unsuccessful. In the remaining 294 cases, the algorithm was able to find a solution within an average number of 369 iterations. More detailed statistics on the number of iterations, broken down among different radii are shown in Table XII. The average number of iterations increases as the radius of the underlying CA increases, which can be attributed to the growth of $\mathcal{A}_r$ as r increases.

TABLE XII
EXPERIMENT 5: THE PERFORMANCE OF THE GA EXPRESSED IN TERMS OF THE AVERAGE OF THE MINIMUM, AVERAGE, AND MAXIMUM NUMBER OF ITERATIONS, PER RADIUS OF THE SOLUTION

radius	min(iter)	avg(iter)	max(iter)
2	25	217	4988
3	77	325	2457
4	250	572	2173

TABLE XIII
EXPERIMENT 5: MINIMAL, AVERAGE, AND MAXIMAL PERCENTAGE OF THE NUMBER OF GA ITERATIONS, BEFORE FINDING THE SOLUTION, DURING WHICH THE MAJORITY OF THE POPULATION HAD THE RADIUS OF THE TARGET SOLUTION, PER RADIUS OF THE SOLUTION

radius	min(% iter)	avg(% iter)	max(% iter)
2	0.00%	57.74%	88.24%
3	4.64%	63.26%	94.49%
4	10.60%	75.00%	98.49%

We also analyze the structure of the populations with regard to the ratio of individuals with a specific radius to the overall size of population. We observed that most of the evolved individuals have the radius of the final solution starting from some point in time, early during the evolution. In other words, a majority of individuals aligns to the radius of the solution, long before the solution itself is found.

More precisely, for radii 3 and 4, in all of the cases considered, the solution always has the correct radius, and a strict majority (more than half of individuals) has this radius some time before finding the solution. In the case of radius 2, the same happens for 93% of the test cases. Otherwise, the final solution and the majority of individuals have a larger radius than the original rule, which is possible due to Fact 1.

A more detailed breakdown of the results is presented in Table XIII. This table shows the minimum, average, and maximal percentage (calculated from all of the runs of the GA) of the total number of GA iterations before finding the solution, during which the strict majority of the population has the same radius as the CA that is looked for. High percentages correspond to cases where the correct radius is detected early in the evolution.

The results shown in Table XIII suggest that the algorithm is typically able to discover the correct radius of the solution long before finding the solution itself. This justifies the use of individuals with variable length and show that the GA is able to correctly identify the radius.

VII. SUMMARY

The main goal of this paper was to formally define, discuss, and solve the identification problem for CAs in the context of incomplete observations. This setting is motivated by real-world situations in which a CA-based model is desired.

A solution algorithm was presented, and its performance was evaluated for all ECAs and various CA families with larger radii by performing a series of computational experiments. The results show that our identification algorithm can solve the problem in many complex settings and that the effort associated with the identification of CAs is connected to their dynamical properties, i.e., the effort of identification grows with the complexity of the CA. The identification algorithm is capable of correctly identifying the neighborhood radius and is able to solve the problem even in cases where a relatively high number of cell states are missing in the observations. Moreover, simple synchrony effects expressed by constant time gaps were studied and the results show that in some of such settings the problem becomes either insolvable or at least very hard to solve. Therefore, introducing randomness to the observation timing is recommended.

These positive experimental results and the general structure of the identification algorithm, which can be easily adapted to higher-dimensional and multistate CAs, will direct our future research on this topic. The ultimate goal is to establish a general identification framework applicable in a broad range of real-world modeling scenarios.

The preliminary results of our ongoing works suggest that the identification algorithm can be further generalized to solve identification problems involving both incomplete and noisy observations, i.e., observations where some of the states were wrongly recorded. In the future, we will also address the identification problem for stochastic CAs and continuous CAs. Preliminary results on the identification of nondeterministic CAs are already available in the case of α-asynchronous CAs [34] and diploid CAs [35].

ACKNOWLEDGMENT

The computational resources used in simulations for Experiments 1, 3–5 (STEVIN Supercomputer Infrastructure) and services used in this paper were kindly provided by Ghent University, the Flemish Supercomputer Center (VSC), the Hercules Foundation, and the Flemish Government Department EWI. Computations used in simulations for Experiment 2 were carried out at the Academic Computer Centre in Gdansk (TASK KDM).

REFERENCES

[1] D. Das, "A survey on cellular automata and its applications," in *Global Trends in Computing and Communication Systems* (Communications in Computer and Information Science), vol. 269, P. V. Krishna, M. R. Babu, and E. Ariwa, Eds. Heidelberg, Germany: Springer, 2012, pp. 753–762.

[2] S. Al-Kheder, J. Wang, and J. Shan, "Cellular automata urban growth model calibration with genetic algorithms," in *Proc. Urban Remote Sens. Joint Event*, Paris, France, Apr. 2007, pp. 1–5.

[3] E. Sapin, L. Bull, and A. Adamatzky, "Genetic approaches to search for computing patterns in cellular automata," *IEEE Comput. Intell. Mag.*, vol. 4, no. 3, pp. 20–28, Aug. 2009.

[4] P. L. Rosin, "Image processing using 3-state cellular automata," *Comput. Vis. Image Understanding*, vol. 114, no. 7, pp. 790–802, 2010.

[5] F. C. Richards, T. P. Meyer, and N. H. Packard, "Extracting cellular automaton rules directly from experimental data," *Physica D Nonlinear Phenomena*, vol. 45, nos. 1–3, pp. 189–202, 1990.

[6] M. Mitchell, J. P. Crutchfield, and R. Das, "Evolving cellular automata with genetic algorithms: A review of recent work," in *Proc. 1st Int. Conf. Evol. Comput. Appl. (EvCA)*, 1996. [Online]. Available: http://csc.ucdavis.edu/~evca/evca1/papers.htm

[7] T. Bäck, R. Breukelaar, and L. Willmes, "Inverse design of cellular automata by genetic algorithms: An unconventional programming paradigm," in *Unconventional Programming Paradigms* (LNCS 3566), J.-P. Banâtre, P. Fradet, J.-L. Giavitto, and O. Michel, Eds. Heidelberg, Germany: Springer-Verlag, 2005, pp. 161–172.

[8] E. Sapin, O. Bailleux, and J.-J. Chabrier, "Research of a cellular automaton simulating logic gates by evolutionary algorithms," in *Proc. 6th Eur. Conf. Genet. Program. (EuroGP)*, 2003, pp. 414–423.

[9] S. Bandini, S. Manzoni, and L. Vanneschi, "Evolving robust cellular automata rules with genetic programming," in *Automata*, A. Adamatzky *et al.*, Eds. Frome, U.K.: Luniver Press, 2008, pp. 542–556.
[10] K.-I. Maeda and C. Sakama, "Identifying cellular automata rules," *J. Cellular Automata*, vol. 2, no. 1, pp. 1–20, 2007.
[11] D. Andre, F. H. Bennett, III, and J. R. Koza, "Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem," in *Proc. 1st Annu. Conf. Genet. Program.*, 1996, pp. 3–11.
[12] C. Ferreira, "Gene expression programming: A new adaptive algorithm for solving problems," *Complex Syst.*, vol. 13, no. 2, pp. 87–129, 2001.
[13] L. Kroczek and I. Zelinka, "Investigation on cellular automaton rule estimation," *J. Cellular Automata*, vol. 13, no. 4, pp. 307–323, 2018.
[14] X. Liu, X. Li, L. Liu, J. He, and B. Ai, "A bottom-up approach to discover transition rules of cellular automata using ant intelligence," *Int. J. Geograph. Inf. Sci.*, vol. 22, nos. 11–12, pp. 1247–1269, 2008.
[15] L. Bull and A. Adamatzky, "A learning classifier system approach to the identification of cellular automata," *J. Cellular Automata*, vol. 2, no. 1, pp. 21–38, 2007.
[16] A. Adamatzky, *Identification of Cellular Automata*. London, U.K.: Taylor & Francis Group, 1994.
[17] Y. Yang and S. A. Billings, "Extracting Boolean rules from CA patterns," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 4, pp. 573–580, Aug. 2000.
[18] Y. Yang and S. A. Billings, "Neighborhood detection and rule selection from cellular automata patterns," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 30, no. 6, pp. 840–847, Nov. 2000.
[19] X. Sun, P. L. Rosin, and R. R. Martin, "Fast rule identification and neighborhood selection for cellular automata," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 3, pp. 749–760, Jun. 2011.
[20] P. Gács, G. L. Kurdyumov, and L. A. Levin, "One-dimensional uniform arrays that wash out finite Islands," *Problemy Peredachi Informatsii*, vol. 14, no. 3, pp. 92–96, 1978.
[21] N. H. Packard, *Adaptation Toward the Edge of Chaos*. Champaign, IL, USA: Univ. Illinois Urbana–Champaign, 1988.
[22] B. Wolnik, M. Dembowski, W. Bołt, J. M. Baetens, and B. De Baets, "Density-conserving affine continuous cellular automata solving the relaxed density classification problem," *J. Phys. A Math. Theor.*, vol. 50, no. 34, Jul. 2017, Art. no. 345103.
[23] M. Dembowski, B. Wolnik, W. Bołt, J. M. Baetens, and B. De Baets, "Affine continuous cellular automata solving the fixed-length density classification problem," *Nat. Comput.*, vol. 17, no. 3, pp. 467–477, Jul. 2017.
[24] E. Chuvieco and A. Huete, *Fundamentals of Satellite Remote Sensing*. Boca Raton, FL, USA: CRC, 2009.
[25] S. Wolfram, "Statistical mechanics of cellular automata," *Rev. Modeling Phys.*, vol. 55, pp. 601–644, Jul. 1983.
[26] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI, USA: Univ. Michigan Press, 1975.
[27] M. Tomassini and M. Perrenoud, "Cryptography with cellular automata," *Appl. Soft. Comput.*, vol. 1, no. 2, pp. 151–160, 2001.
[28] F. Seredynski and J. Skaruz, "Evolutionary algorithms and cellular automata towards image reconstruction," in *Proc. 3rd Int. Conf. Emerg. Appl. Inf. Technol.*, 2012, pp. 283–286.
[29] P. H. T. Schimit, "On exploring the genetic algorithm for modeling the evolution of cooperation in a population," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 19, no. 8, pp. 2801–2810, 2014.
[30] D. Whitley, S. Rana, and R. B. Heckendorn, "The island model genetic algorithm: On separability, population size and convergence," *J. Comput. Inf. Technol.*, vol. 7, no. 1, pp. 33–47, 1998.
[31] F. Bagnoli, R. Rechtman, and S. Ruffo, "Damage spreading and Lyapunov exponents in cellular automata," *Phys. Lett. A*, vol. 172, nos. 1–2, pp. 34–38, 1992.
[32] S. Wolfram, "Universality and complexity in cellular automata," *Physica D Nonlinear Phenomena*, vol. 10, nos. 1–2, pp. 1–35, 1984.
[33] M. A. Shereshevsky, "Lyapunov exponents for one-dimensional cellular automata," *J. Nonlinear Sci.*, vol. 2, no. 1, pp. 1–8, 1992.
[34] W. Bołt, B. Wolnik, J. M. Baetens, and B. De Baets, "On the identification of α-asynchronous cellular automata in the case of partial observations with spatially separated gaps," in *Challenging Problems and Solutions in Intelligent Systems*, G. De Tré *et al.*, Eds. Cham, Switzerland: Springer, 2016, pp. 23–36.
[35] W. Bołt, A. Bołt, B. Wolnik, J. M. Baetens, and B. De Baets, "A statistical approach to the identification of diploid cellular automata," in *Theory and Practice of Natural Computing*, C. Martín-Vide, R. Neruda, and M. A. Vega-Rodríguez, Eds. Cham, Switzerland: Springer, 2017, pp. 37–48.

Witold Bołt received the M.Sc. degrees in computer science and in theoretical mathematics from the University of Gdańsk, Gdańsk, Poland, in 2008 and 2009, respectively. He is currently pursuing the Ph.D. degree with the Institute of Systems Research, Polish Academy of Science, Warsaw, Poland.

He is a Software Architect and a Consultant for enterprise customers in Europe. His current research interests include identification of cellular automata and evolutionary algorithms.

Jan M. Baetens received the M.Sc. degree in bioscience engineering: land and forestry management and the Ph.D. degree in applied biological sciences from Ghent University, Ghent, Belgium, in 2007 and 2012, respectively.

He is an Assistant Professor with the Department of Data Analysis and Mathematical Modeling, Ghent University, Ghent, in 2017, where he is a member of the research unit Knowledge-Based Systems (KERMIT). His current research interests include analysis of spatially explicit models and case-based development of such models.

Bernard De Baets received the M.Sc. degree (*summa cum laude*) in mathematics, the Postgraduate degree in knowledge technology (*summa cum laude*), and the Ph.D. degree (*summa cum laude*) in mathematics from Ghent University, Ghent, Belgium, in 1988, 1991, and 1995, respectively, and the Honorary Doctorate degree from the University of Turku, Turku, Finland, in 2017.

He became a Senior Full Professor in applied mathematics with Ghent University in 1999, where he is leading KERMIT, the research unit Knowledge-Based Systems. He has co-authored nearly 500 papers in international peer-reviewed journals.

Dr. De Baets was a recipient of the Government of Canada Award in 1988, the Honorary Professor of Budapest Tech in 2006, and the Profesor Invitado of the Universidad Central "Marta Abreu" de Las Villas, Cuba, in 2017. He serves on the editorial boards of various international journals, in particular, as the Co-Editor-in-Chief of *Fuzzy Sets and Systems*. He became an IFSA Fellow in 2011.