Characterization of Maximal Length Cellular Automata Using NSRTD

Som Banerjee
Mamata Dalui

Department of Computer Science and Engineering
National Institute of Technology, Durgapur
West Bengal 713209, India

Maximal length cellular automata (CAs) have gained significant attention from researchers due to their applications in different areas like random number generation, cryptography and test pattern generation. This paper reports the theoretical framework of the next state rule min term transition diagram (NSRTD) for the characterization of maximal length CAs in null-boundary condition. The proposed solution helps in verifying whether a given cellular automaton (CA) is a maximal length CA or not in $O(2^n)$ time. Also, the proposed solution can identify and eliminate a CA candidate that fails to configure a maximal length CA in linear time.

Keywords: cellular automata; maximal length CA; fixed-point attractor; NSRTD; null boundary CA

1. Introduction

Cellular automata (CAs) were first proposed and introduced in the literature by John von Neumann [1]. Further progress on CAs has been made by Stephen Wolfram [2] with the introduction of different structural and theoretical properties of a cellular automaton (CA). Many other researchers have also explored the structure and properties of CAs. A CA consists of a grid of discrete cells that are usually arranged in a regular lattice, such as a one-dimensional, two-dimensional or even higher-dimensional structures. A CA evolves over discrete space and time depending upon a set of rules. A CA rule decides how a CA cell state will change based upon the state of the cell itself and its neighborhood. The rules for updating the cells are typically specified in the form of a next state function, also called the transition function. These rules can be very simple or highly complex, depending on the specific CA and the behavior it is intended to model. As CAs can be used to explore emergent phenomena and complex patterns that can arise from simple rules, they have been used to model a wide variety of applications like authentication and cryptography [3, 4], fault detection [5], test pattern generation [6], language recognition [7], cache coherence verification [8] and memory testing [9].

https://doi.org/10.25088/ComplexSystems.35.2.141

Maximal length CAs are a special class of CA that form two connected components in their state transition diagram (STD), one with cycle length $2^n - 1$ and the other with state 0 as a fixed-point attractor for the CA with n cells [6, 10, 11, 12]. Since maximal length CAs can generate sequences of maximum possible length and the sequence repeats only after a very long period, these CAs are often used in cryptography [13] and random number generation [6] because of their ability to produce pseudorandom sequences that have desirable statistical properties. The authors in [14] conducted an experimental approach and were the first to find that a null-boundary hybrid CA with certain combinations of rules 150 and 90 gives a maximal length CA, whereas a periodic boundary hybrid CA cannot give a maximal length CA. There exist two methods to evaluate whether a CA is a maximal length CA: that is, searching exhaustively (check if the CA is reversible and then check if the cycle is of maximal length) and primitive-based analysis (determine the characteristic polynomial, then check if it is primitive) [15], but both approaches take exponential time. The researchers in [11, 16] used primitive polynomials to identify maximal length CAs. The researchers in [17] explored the nonlinear elementary CA (ECA) rules for maximal length CAs, but no verification process was presented to determine nonlinear maximal length CAs. Their shortcomings were overcome in [12], where the authors presented an algorithm that verifies whether a nonlinear CA shows isomorphism as a linear maximal length CA by utilizing the concepts of information flow, isomorphism and cycle length.

Most of the research done so far to identify maximal length CAs has used the STD. The STD suffers from resource constraints because the number of states increases exponentially with the CA size, becoming a limitation for large CA sizes. The next state rule min term transition diagram (NSRTD) can be a good characterization tool for verifying maximal length CAs, as it can identify the existence of self-loop(s) along with multi-length cycle(s) [18]. Some works on CA characterization have been done using the NSRTD [19, 20]. However, identification of maximal length CAs under the null-boundary condition using the NSRTD tool have not yet been explored. This idea motivates us to carry out research on how to identify the maximal length CA rules using the NSRTD concept. The underlying theoretical basis for the NSRTD is given in [18]. Now, based upon the theoretical concept of the NSRTD, this paper focuses on the identification of maximal length CAs from a given rule vector of any length under the null-boundary condition.

2. Cellular Automaton Preliminaries

CAs are dynamical systems that can be successfully employed in the behavioral analysis of complex systems. A CA cell can be characterized by two states, either "0" or "1." The next state (NS) of the CA cell i at time step $t+1$ is $S_i^{t+1} = f_i(S_{i-1}^t, S_i^t, S_{i+1}^t)$, where S_i^t, S_{i+1}^t and S_{i-1}^t are the present state (PS) (at time t) of cell i and its right and left neighbors, respectively, and f_i is known as the next state function. The collection of states represented by $S^t = (S_1^t, S_2^t, \ldots, S_n^t)$ represents the PS of the CA cells at time step t. A truth table (Table 1) depicts the next state function f_i. The decimal representation of the eight outputs of f_i is known as its *rule* R_i. There can be $2^8 = 256$ possible rules in 2-state 3-neighborhood CAs. Four such rules: 90, 150, 13 and 240, are illustrated in Table 1. The first row shows the possible $2^3 = 8$ combinations of the PS of cells $i-1$, i and $i+1$ at time t.

PS	111	110	101	100	011	010	001	000	Rule
RMT	T(7)	T(6)	T(5)	T(4)	T(3)	T(2)	T(1)	T(0)	
NS	0	1	0	1	1	0	1	0	90
NS	1	0	0	1	0	1	1	0	150
NS	0	0	0	0	1	1	0	1	13
NS	1	1	1	1	0	0	0	0	240

Table 1. Rule min terms (RMTs) of the rules 90, 150, 13 and 240.

The following key definitions on CA theory are essential for understanding this paper.

- *Rule vector*. The combination of rules $\langle R_1, R_2, \ldots, R_n \rangle$ that configures the CA cells is known as the rule vector (RV).
- *Uniform and nonuniform CAs*. If all the rules in an RV are identical, then it is known as a uniform CA. Otherwise, the CA is a nonuniform CA.
- *Null-boundary CA*. A CA can be called a null-boundary CA (NBCA) if the left (resp., right) neighbor of the leftmost (resp., rightmost) terminal CA cell is permanently connected to logic 0-state [3] as shown in Figure 1.

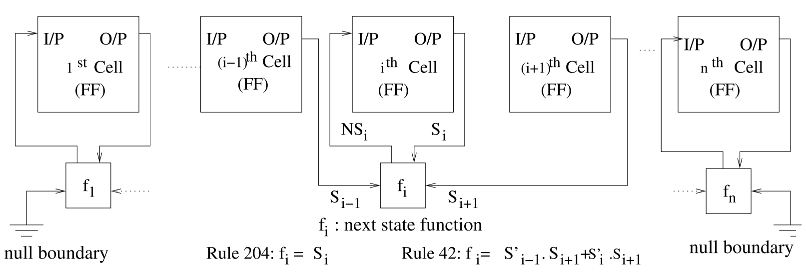


Figure 1. n-cell NBCA.

- *RMT*. The combination of the PS (S_{i-1}^t, S_i^t, S_{i+1}^t) in a 3-neighborhood CA forms a rule min term (RMT), as illustrated in the first row of Table 1. An RMT can be expressed by T(x), where $x = \{0, 1, \ldots, 6, 7\}$. The RMT for the CA cell i at time t is usually expressed by $\mathrm{T}_i^t(x)$. For example, $\mathrm{T}_2^t(5)$ refers to the 2-cell RMT T(5) at time t. However, in all the pictorial representations in this paper, the RMT is expressed by x; that is, $\mathrm{T}_i^t(5)$ is expressed as 5 for ease of representation.
- *RMT string*. An RMT string (RS) can be expressed as a consecutive sequence of RMTs that appears in a CA state. RS is expressed as $(\mathrm{T}_1, \mathrm{T}_2, \ldots, T_n)$ for a CA with n cells, where $\mathrm{T}_i \in \{\mathrm{T}(0), \mathrm{T}(1), \ldots, \mathrm{T}(7)\}$. For a 2-state 3-neighborhood 6-cell NBCA, state 101101 can be expressed by the RS

$$(\mathrm{T}_1, \mathrm{T}_2, \mathrm{T}_3, \mathrm{T}_4, \mathrm{T}_5, \mathrm{T}_6) = (\mathrm{T}(2), \mathrm{T}(5), \mathrm{T}(3), \mathrm{T}(6), \mathrm{T}(5), \mathrm{T}(2)).$$

- *Attractor*. The states that create cycle(s) during a CA state transition are called attractors of a CA. An attractor that forms a self-loop ($10 \to 10$ of Figure 2(a)) is known as a single length cycle attractor or a fixed-point attractor. The cycle that is formed with multiple states ($6 \to 9 \to 6$ of Figure 2(b)) is known as a multi-length cycle attractor.
- *Reversible and irreversible CAs*. If all the configurations of a CA are cyclic during its state transition, then it is called a reversible CA. A reversible CA has no unreachable states (which cannot be reached from any other state). On the other hand, an irreversible CA has unreachable state(s) during its state transition (state 0, 1, 2, 3 of Figure 2(a)).
- *Maximal length CA*. An n-cell maximal length CA is characterized by the presence of a cycle of length $2^n - 1$ with all nonzero states [3]. Figure 2(c) shows a four-cell maximal length CA.

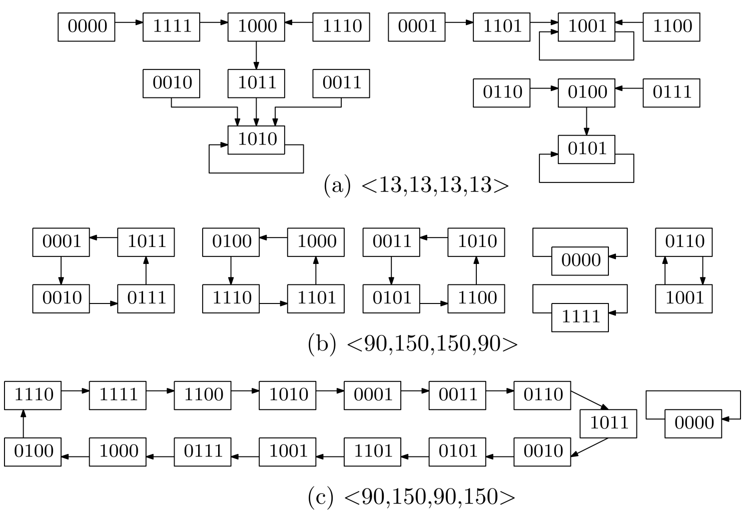


Figure 2. State transitions of a four-cell NBCA.

3. NSRTD

The NSRTD is a graph-based CA characterization tool that was first reported in [18] to identify the existence of self-loops(s) and multi-length cycle(s) during a CA state transition. Further, constructing an NSRTD considers the spatial relationship of the RMTs of a CA cell (following the next cell RMT (NCR) relationship) as well as the temporal relationship of RMTs of a CA cell (following the next state RMT (NSR) relationship). So, in a nutshell, an NSRTD can provide a detailed analysis of the spatio-temporal relationship of the RMTs of a CA, which enables in-depth analysis of the intricate structure of a CA, as essential for devising different real-life applications of CAs. Some of the definitions and terminologies that are relevant for proper understanding of the NSRTD theory include NCR, NSR, NSRS, NSRS graph (NSRS-G) and compatible classes of NSRSs.

During the CA state transition at time t, the RMTs T_i^t and T_{i+1}^t of two consecutive CA cells (cells i and $i+1$) are related via the NCR relationship, which is reported in Table 2.

RMT T_i	NCR of RMT T_i
(cell i rule)	(RMTs of the cell $i+1$ rule)
T(0)/T(4)	T(0), T(1)
T(1)/T(5)	T(2), T(3)
T(2)/T(6)	T(4), T(5)
T(3)/T(7)	T(6), T(7)

Table 2. NCR relationship between cell i and cell $i+1$.

The RMT T_i^t of cell i at time t is updated to T_i^{t+1} at time $t+1$ during the CA state transition, and T_i^{t+1} is called the NSR of T_i^t.

The next state RMT sequence (NSRS) is the sequence of NSRs, depending upon which the state of a CA cell changes during its state transition.

In a CA cell, all the possible NSRSs can be represented by using a directed graph known as the NSRS graph (NSRS-G), where an RMT represents a vertex and the edge represents the NSRs.

Further, the NSRSs between the two consecutive CA cells are said to be compatible if all the self-loop or multi-length cycles of the cell $i+1$ follow the NCR relationship with self-loop or multi-length cycles of the cell i.

The NSRTD is a two-dimensional directed graph where one dimension represents the NSR relationship of a CA cell and the other represents the NCR relationship between the RMTs of consecutive CA cells, that is, it is constructed with the compatible class of NSRSs (self-loop(s) and/or multi-length cycle(s)).

The details of NSRTD theory have been reported in [18]. The next section elaborates the identification of a maximal length CA from a given rule vector (RV) by developing and using the NSRTD concept.

4. Proposed Work

This section presents the identification of a maximal length CA using the concept of NSRTD.

Given the RV of a CA with n cells, the solution steps for verifying whether the given CA is a maximal length CA or not are depicted in Algorithm 1, which accepts as input the RV, the CA length n and the set of NCRs to give a decision on whether or not the CA is a maximal length CA.

Algorithm 1 starts by identifying the NSRs (here, NSRs represent a set of NSR) of CA cells $i = 1, 2, 3, \ldots, n$ (represented as NSRs_i) using CA rule R_i as described in step 1. In step 2, each cell's NSRs are merged to get the NSRS graph (represented as NSRS-G_i) for CA cells $i = 1, 2, 3, \ldots, n$. Now, each of the NSRS-G_i is examined to find the presence of the cycle(s) and cycle length. Steps 3(a) and 3(b) find the self-loop(s) and multi-length cycle(s) represented by SL_i and ML_i, respectively, from the NSRS-G_i. Now in step 3(c), the SL_i and ML_i for each NSRS-G_i are grouped together in a set that we call the cycle set Cyc_i.

Further, in step 4, if any of the NSRS-G_i do not have a self-loop in RMT T(0) (i.e., no graph in the NSRTD with a self-loop in the RMT T(0) can be formed, thereby violating the necessary condition of a maximal length CA), the algorithm terminates. Next, the algorithm checks if any of the NSRS-G_i do not form a multi-length cycle composed of all possible RMTs (i.e., {T(0), T(1), T(2), T(3)} for the first cell, {T(0), T(1), …, T(7)} for all other cells, and {T(0), T(2), T(4), T(6)} for the last cell), the algorithm terminates. Else, the algorithm jumps to step 5.

In step 5, the compatible class of NSRSs $C_{i(i+1)}$ between the cycles (i.e., self-loop(s) along with the multi-length cycle(s)) of two consecutive CA cells, for each of n CA cells, is identified. Further in step 6, each RMT sequence S_u is generated by using the compatibility class of NSRSs $C_{i(i+1)}$ identified in step 5.

Step 7 counts the number of S_u. If the number of $S_u > 2$, the algorithm terminates, as it violates the necessary conditions for forming a maximal length CA (i.e., for a maximal length CA, there should be exactly two RMT sequences). Otherwise, the algorithm returns the decision that the given CA with RV is a maximal length CA.

Algorithm 1. Maximal length CA identification.

Input: CA RV $= \langle R_1, R_2, R_3, \ldots, R_n \rangle$, CA length n, set of NCRs.

Output: Decision on whether the given CA with RV is a maximal length CA or not.

Start

1. Identify NSRs$_i$ using rule R_i, $\forall_{1 \le i \le n}$.
   /* Here, i represents a CA cell index */
2. Create NSRS-G$_i \forall_{1 \le i \le n}$.
3. (a) Identify self-loop(s) $\text{SL}_i = \text{SL}_{iu}$, $\forall_{1 \le i \le n}$ and $u = 1$ to x, where x is the count of self-loop(s) in NSRS-G$_i$
   (b) Identify multi-length cycle(s) $\text{ML}_i = \text{ML}_{iv}$, $\forall_{1 \le i \le n}$ and $v = 1$ to y, where y is count of multi-length cycle(s) in NSRS-G$_i$
   (c) $\text{Cyc}_i = \{\text{SL}_i \cup \text{ML}_i\}$
4. **if** any of the NSRS-G$_i$ do not have a self-loop in RMT T(0), $\forall_{1 \le i \le n}$
   then
   Exit
   else if any of the NSRS-G$_i$ do not form a multi-length cycle consisting of all RMTs (i.e., {T(0), T(1), T(2), T(3)} for $i = 1$, $\{T(0), T(1), \ldots, T(7)\} \forall_{2 \le i \le (n-1)}$ and {T(0), T(2), T(4), T(6)} for $i = n$)
   then
   Exit
   else
   Goto step 5
5. Find compatibility class between
   Cyc_i and $\text{Cyc}_{i+1} = C_{i(i+1)} = \left\{\text{Cyc}_{ia}, \text{Cyc}_{(i+1)b} \forall_{a,b}\right\}$; $\forall_{1 \le i \le n}$, such that $\text{Cyc}_{(i+1)b}$ is the NCR of $\text{Cyc}_{ia} \forall_{a,b}$
6. Generate each RMT sequence S_u as
   $S_u = S_{1u} \to S_{2u} \to \cdots \to S_{(n-1)}u \to S_{nu}$;
   where $(\text{S}_{iu}, \text{S}_{(i+1)u}) \in \text{C}_{i(i+1)}$, $\forall_{1 \le i \le (n-1)}$.
7. **if** number of $S_u > 2$ **then**
   Exit
   else
   CA with rule vector RV is a maximal length CA
   Stop

4.1 Complexity Analysis

In Algorithm 1, step 1 identifies the NSRs for each individual CA cell. For an NBCA in a 2-state 3-neighborhood configuration, the maximum number of RMTs can be eight and each RMT can have at most four possible NSRs. Therefore, computing the NSRs for each cell takes constant time. So, step 1 of Algorithm 1 finds the NSRs for an n-cell CA in $O(n)$ time. Creating each cell in the NSRS graph by merging the NSRs of each individual CA cell identified in step 1 takes constant time. So, in step 2, NSRS-G computation for an n-cell CA takes $O(n)$ time. Steps 3(a) and 3(b) find the possible cycle(s) (self-loop(s) and multi-length cycle(s), respectively) for each individual CA cell from the NSRS-G. There can be a maximum of eight self-loops corresponding to eight RMTs of a CA cell, and the count of multi-length cycles is also finite and independent of the CA length n. For an n-cell CA, step 3 identifies the possible cycle(s) in $O(n)$ time. Step 4 checks each of the NSRS-Gs for the presence of self-loops in RMT T(0) and the presence of a multi-length cycle involving all RMTs in constant time, and therefore, for an n-cell CA, it takes $O(n)$ time. Further, step 5 finds the compatible class of NSRSs between two consecutive CA cells, which depends upon the count of cycles in the NSRS-G and finding the NCR-compatible RMTs between cycles of two consecutive CA cells. The count of cycle(s) in the NSRS-G is independent of n, taking constant time. For each CA cell, there can be a maximum of eight RMTs involved in a cycle. Each RMT can have a maximum of two out-degree edges (which corresponds to its NCRs). So, the time taken to find each compatibility class between two consecutive CA cells is independent of n and takes constant time. For an n-cell CA, there will be $n-1$ checks for compatible class, and it takes $O(n-1)$ time. Step 6 generates each individual RMT sequence S_u. Generating each RMT sequence S_u with a self-loop NSRS takes $O(n)$ time for an n-cell CA. There exist two cases when the complexity can go exponential.

First, in the case of a uniform CA that is configured by using rule 204, the total number of RMT sequences S ($S = \sum S_u$) will be 2^n, making the time complexity $O(n * 2^n)$. But it is also to be noted that a uniform CA configured with rule 204 will not satisfy the second condition given in step 4 of Algorithm 1, thus it will not be considered as potential maximal length CA. So, we are not considering this case here.

Second, we know that for a maximal length CA, there exists a multi-length cycle consisting of all the RMTs in each cell NSRS-G. We also know that each RMT involved in the cycle can have two possible NCRs. So, for the first cell, there can be the four RMTs involved in a cycle that forms a compatibility class with the eight RMTs involved in the cycle of the second cell. Similarly, for the second cell, the eight RMTs involved in the cycle can create a compatibility class with 16 RMTs (two same cycles containing eight RMTs each) of the third cell, and so on. So, during the creation of an RMT sequence in a maximal length CA, the compatibility class of NSRSs increases exponentially with CA length n, making the time complexity $O(2^n)$. Step 7 counts at most two RMT sequences, and if the number of RMT sequences exceeds two, the algorithm terminates. So, Step 7 takes constant time. Therefore, the overall running time complexity of Algorithm 1 becomes $O(2^n)$.

Note: During the development of the proposed NSRTD theoretical framework, the structure and properties of a CA to form maximal length have been studied meticulously, keeping in mind how to identify and eliminate a non-potential CA candidate that fails to configure a maximal length CA in the early stage of Algorithm 1. After careful observation, we have introduced two conditions (later presented as two theorems) in step 4 of Algorithm 1, as they form two necessary criteria for a potential CA candidate to form maximal length. So, a non-potential CA candidate is identified in the early stage of Algorithm 1 in $O(n)$ time.

The steps of Algorithm 1 are illustrated using the following examples.

Example 1. Consider the scenario where Algorithm 1 accepts rule vector RV $= \langle 90, 150, 90, 150 \rangle$ and CA length $n = 4$. NSRs of each CA cell (represented by NSRs$_i$) are obtained by applying rule R_i for $i = 1, 2, 3, 4$ (where i represents CA cell i) in step 1. Step 2 finds the NSRS graph (represented by NSRS-G$_i$) for $i = 1, 2, 3, 4$. From the NSRS-Gs as shown in Figure 3, we can identify the self-loops of each CA cell using step 3(a):

- First cell: $\{0 \to 0, 3 \to 3\}$.
- Second cell: $\{0 \to 0, 7 \to 7\}$.
- Third cell: $\{0 \to 0, 5 \to 5\}$.
- Last cell: $\{0 \to 0, 2 \to 2\}$.

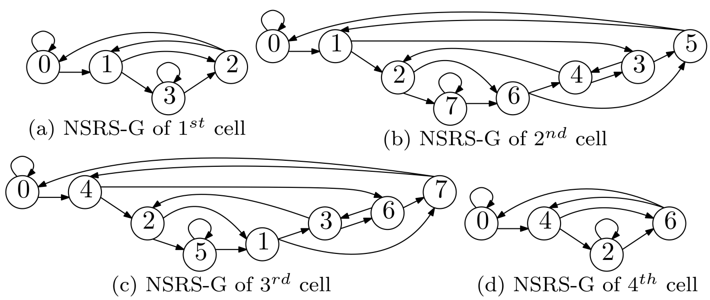


Figure 3. NSRS-G for the CA $\langle 90, 150, 90, 150 \rangle$.

On the other hand, the multi-length cycles of each CA cell are identified using step 3(b), which are:

- First cell: $\{0 \to 1 \to 3 \to 2 \to 0,\ 0 \to 1 \to 2 \to 0,\ 1 \to 3 \to 2 \to 1,\ 1 \to 2 \to 1\}$.

- Second cell: {0 → 1 → 2 → 7 → 6 → 4 → 3 → 5 → 0, 0 → 1 → 2 → 7 → 6 → 5 → 0, 0 → 1 → 2 → 6 → 4 → 3 → 5 → 0, 0 → 1 → 2 → 6 → 5 → 0, 0 → 1 → 3 → 5 → 0, 1 → 2 → 7 → 6 → 4 → 3 → 5 → 1, 1 → 2 → 7 → 6 → 5 → 1, 1 → 2 → 6 → 4 → 3 → 5 → 1, 1 → 2 → 6 → 5 → 1, 1 → 3 → 5 → 1, 2 → 7 → 6 → 4 → 2, 2 → 6 → 4 → 2, 4 → 3 → 4}.
- Third cell: {0 → 4 → 2 → 5 → 1 → 3 → 6 → 7 → 0, 0 → 4 → 2 → 5 → 1 → 7 → 0, 0 → 4 → 2 → 1 → 3 → 6 → 7 → 0, 0 → 4 → 2 → 1 → 7 → 0, 0 → 4 → 6 → 7 → 0, 4 → 2 → 5 → 1 → 3 → 6 → 7 → 4, 4 → 2 → 5 → 1 → 7 → 4, 4 → 2 → 1 → 3 → 6 → 7 → 4, 4 → 2 → 1 → 7 → 4, 4 → 6 → 7 → 4, 2 → 5 → 1 → 3 → 2, 2 → 1 → 3 → 2, 3 → 6 → 3}.
- Last cell: {0 → 4 → 2 → 6 → 0, 0 → 4 → 6 → 0, 4 → 2 → 6 → 4, 4 → 6 → 4}.

In step 4, the algorithm checks if any NSRS-G_i do not a have self-loop in RMT T(0). Here, as all the NSRS-G_i have a self-loop on RMT T(0) (i.e., 0 → 0 in all the CA cells), the algorithm proceeds to check the next condition. Further, the algorithm checks if any NSRS-G_i do not form a multi-length cycle consisting of all RMTs (i.e., {T(0), T(1), T(2), T(3)} for $i = 1$; {T(0), T(1), …, T(7)} for $i = 2$ to $n - 1$; and {T(0), T(2), T(4), T(6)} for $i = n$). Here, as all the NSRS-G_i are composed of NSRSs with multi-length cycles involving all the RMTs (i.e., the first cell NSRS is 0 → 1 → 3 → 2 → 0, and the second, third and fourth cell NSRSs are 0 → 1 → 2 → 7 → 6 → 4 → 3 → 5 → 0, 0 → 4 → 2 → 5 → 1 → 3 → 6 → 7 → 0, and 0 → 4 → 2 → 6 → 0, respectively), the algorithm jumps to step 5. Further, from the identified cycles Cyc_i of the CA cells as in step 3, the compatibility classes $C_{i(i+1)}$ between the CA cells i and $i+1$ are identified by examining the NCR relationship between the RMTs involved in the cycle(s) of each consecutive CA cell (i.e., cells i and $i+1$). The compatibility classes are identified as:

- C_{12} = {((T(0), T(0)), (T(0), T(0))); ((T(3), T(3)), (T(7), T(7))); ((T(0), T(1), T(3), T(2), T(0)), (T(0), T(1), T(2), T(7), T(6), T(4), T(3), T(5), T(0))); …}
- C_{23} = {((T(0), T(0)), (T(0), T(0))); ((T(0), T(1), T(2), T(7), T(6), T(4), T(3), T(5), T(0)), (T(0), T(4), T(2), T(5), T(1), T(3), T(6), T(7), T(0)))); …}
- C_{34} = {((T(0), T(0)), (T(0), T(0))); ((T(5), T(5)), (T(2), T(2))); ((T(0), T(4), T(2), T(5), T(1), T(3), T(6), T(7), T(0)), ((T(0), T(4), T(2), T(6), T(0))); …}

In step 6, each unique RMT sequence (also known as graphs in an NSRTD) is constructed by picking one compatible NSRS from each compatibility class $C_{i(i+1)}$ for each n CA cells. Here, S_1 is a graph in an NSRTD that is made up of self-loop NSRSs only and is represented by graph G_1, and S_2 is a graph in an NSRTD that is made up of multi-length cycle(s) only and is represented by graph G_2 in Figure 4. In step 7, as the number of S_u identified is exactly 2, the algorithm returns that the CA ⟨90, 150, 90, 150⟩ is a maximal length CA.

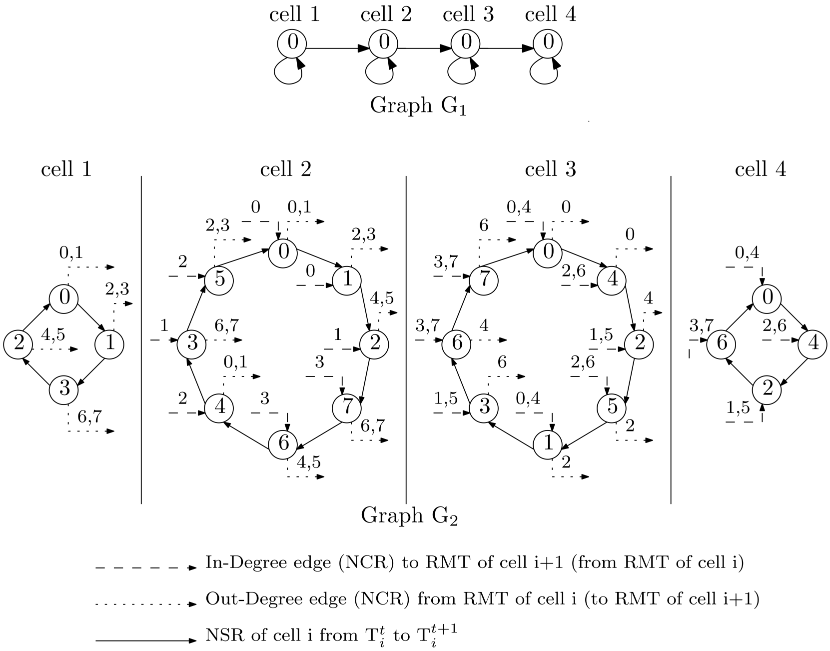


Figure 4. NSRTD for the CA ⟨90, 150, 90, 150⟩.

Discussion: The discussion part depicts how the NSRTD of the CA ⟨90, 150, 90, 150⟩ complies with its STD. Column 1 of Table 3 shows the number of graphs in the NSRTD. Here, we have two graphs: G_1 and G_2 in the NSRTD (Figure 4). Column 2 depicts the RMT strings and column 3 represents the CA states corresponding to these RMT strings. Here, graph G_1 is composed of a single path with single length cycles only, thereby representing a single RMT string, whereas graph G_2, composed of multi-length cycles only, represents $2^n - 1$ number of RMT strings for an n-cell CA (here it is 15 for $n = 4$). For the CA ⟨90, 150, 90, 150⟩, one RMT string corresponding to G_1 and the 15 RMT strings corresponding to G_2 are represented in column 2. From Table 3, it can be observed that the CA states as listed in column 3 comply with the states in the STD of the CA as shown in Figure 2(c). It can easily be seen that graph G_1 complies with a fixed-point attractor with an all-zero state and graph G_2 complies with a multi-length (15-length) cycle attractor involving the states 1, 2, 3, …, 15.

NSRTD Graph (1)	RMT String (2)				CA State (3)
	Cell 1	Cell 2	Cell 3	Cell 4	
Graph G_1	0	0	0	0	0000
Graph G_2	0	0	1	2	0001
	0	1	2	4	0010
	0	1	3	6	0011
	1	2	4	0	0100
	1	2	5	2	0101
	1	3	6	4	0110
	1	3	7	6	0111
	2	4	0	0	1000
	2	4	1	2	1001
	2	5	2	4	1010
	2	5	3	6	1011
	3	6	4	0	1100
	3	6	5	2	1101
	3	7	6	4	1110
	3	7	7	6	1111

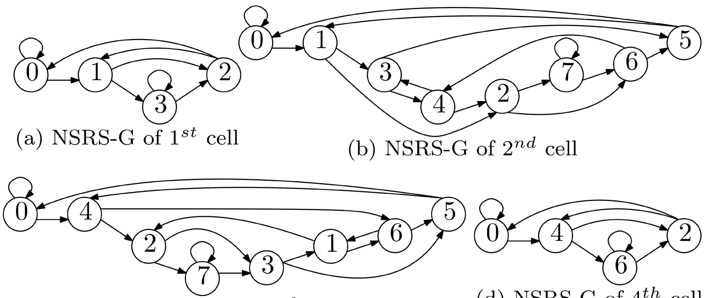
Table 3. Compliance of NSRTD and STD for the CA ⟨90, 150, 90, 150⟩.

Example 2. Consider the scenario where Algorithm 1 accepts rule vector RV = ⟨90, 150, 150, 90⟩ and CA length $n = 4$. NSRs of each CA cell (represented by NSRs_i) are obtained by applying rule R_i for $i = 1, 2, 3, 4$ (where i represents CA cell i) in step 1. Step 2 finds the NSRS graph (represented by NSRS-G_i) for $i = 1, 2, 3, 4$. From the NSRS-G_i as shown in Figure 5, we can identify the self-loops of each CA cell using step 3(a):

- First cell: {0 → 0, 3 → 3}.
- Second cell: {0 → 0, 7 → 7}.
- Third cell: { 0 → 0, 7 → 7}.
- Last cell: { 0 → 0, 6 → 6}.

On the other hand, the multi-length cycles of each CA cell are identified using step 3(b):

- First cell: {0 → 1 → 3 → 2 → 0, 0 → 1 → 2 → 0, 1 → 3 → 2 → 1, 1 → 2 → 1}.
- Second cell: {0 → 1 → 3 → 4 → 2 → 7 → 6 → 5 → 0, 0 → 1 → 3 → 4 → 2 → 6 → 5 → 0, 0 → 1 → 2 → 7 → 6 → 5 → 0, 0 → 1 → 2 → 6→ 5 → 0, 0 → 1 → 3 → 5 → 0, 1 → 3 → 4 → 2 → 7 → 6 → 5 → 1, 1 → 3 → 4 → 2 → 6 → 5 → 1, 1 → 2 → 7 → 6 → 5 → 1, 1 → 2 → 6→ 5 → 1, 1 → 3 → 5 → 1, 3 → 4 → 3, 4 → 2 → 7 → 6 → 4, 4 → 2 → 6 → 4}.



(a) NSRS-G of 1^{st} cell (b) NSRS-G of 2^{nd} cell (c) NSRS-G of 3^{rd} cell (d) NSRS-G of 4^{th} cell

Figure 5. NSRS-G for the CA ⟨90, 150, 150, 90⟩.

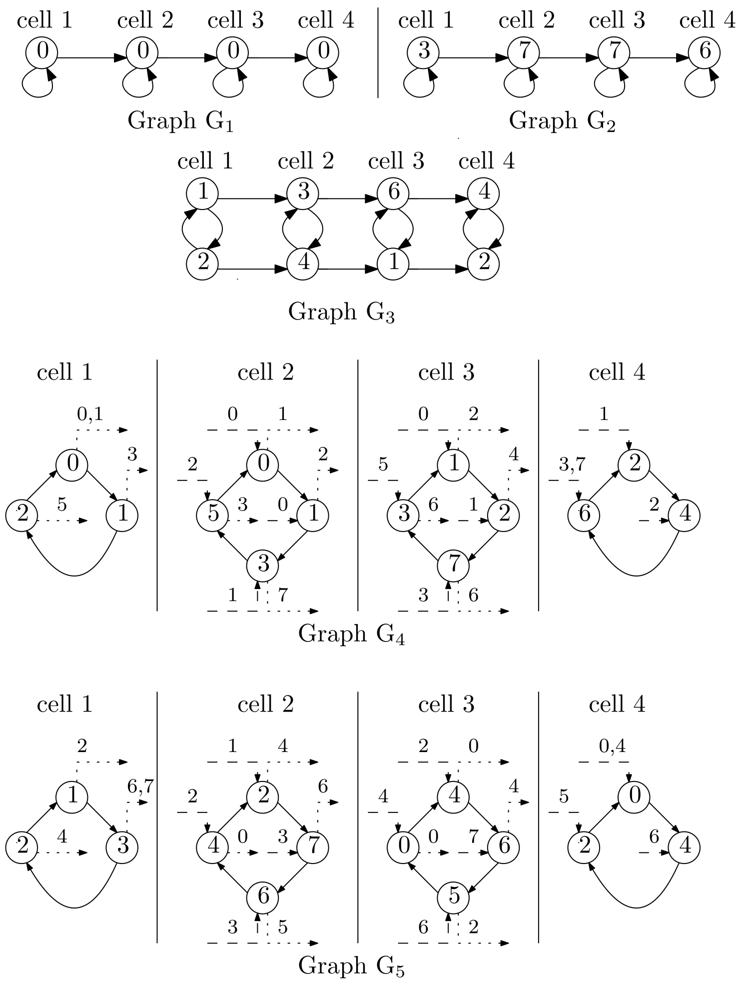
- Third cell: {0 → 4 → 2 → 7 → 3 → 1 → 6 → 5 → 0, 0 → 4 → 2 → 7 → 3 → 5 → 0, 0 → 4 → 2 → 3 → 1 → 6 → 5 → 0, 0 → 4 → 2 → 3 → 5 → 0, 0 → 4 → 6 → 5 → 0, 4 → 2 → 7 → 3 → 1 → 6 → 5 → 4, 4 → 2 → 7 → 3 → 5 → 4, 4 → 2 → 3 → 1 → 6 → 5 → 4, 4 → 2 → 3 → 5 → 4, 4 → 6 → 5 → 4, 2 → 7 → 3 → 1 → 2, 2 → 3 → 1 → 2, 1 → 6 → 1}.
- Last cell: {0 → 4 → 6 → 2 → 0, 0 → 4 → 2 → 0, 4 → 6 → 2 → 4, 4 → 2 → 4}.

In step 4, the algorithm checks if any NSRS-G_i do not have a self-loop in RMT T(0). Here, as all the NSRS-G_i have a self-loop on RMT T(0) (i.e., 0 → 0 in all the CA cells), the algorithm proceeds to check for the next condition. Further, the algorithm checks if any NSRS-G_i do not form a multi-length cycle consisting of all RMTs (i.e., {T(0), T(1), T(2), T(3)} for $i = 1$, {T(0), T(1), …, T(7)} for $i = 2$ to $n - 1$, and {T(0), T(2), T(4), T(6)} for $i = n$). Here, as all the NSRS-G_i are composed of NSRSs with multi-length cycles involving all the RMTs (i.e., the first cell NSRS is 0 → 1 → 3 → 2 → 0, and the second, third and fourth cell NSRSs are 0 → 1 → 3 → 4 → 2 → 7 → 6 → 5 → 0, 0 → 4 → 2 → 7 → 3 → 1 → 6 → 5 → 0, and 0 → 4 → 6 → 2 → 0, respectively), the algorithm jumps to step 5. Further, from the identified cycles Cyc_i of the CA cells as in step 3, the compatibility classes $C_{i(i+1)}$ between the cells i and $i + 1$ are identified by examining the NCR relationship between the RMTs involved in the cycle(s) of each consecutive CA cell (i.e., cells i and $i + 1$). The compatibility classes identified are:

- C_{12} = {((T(0), T(0)), (T(0), T(0))); ((T(3), T(3)), (T(7), T(7))); ((T(1), T(2), T(1)), (T(3), T(4), T(3))); ((T(0), T(1), T(2), T(0)), (T(0), T(1), T(3), T(5), T(0))); …}
- C_{23} = {((T(0), T(0)), (T(0), T(0))); ((T(7), T(7)), (T(7), T(7))); ((T(3), T(4), T(3)), (T(6), T(1), T(6))); ((T(0), T(1), T(3), T(5), T(0)), (T(1), T(2), T(7), T(3), T(1))); …}

- C_{34} = {((T(0), T(0)), (T(0), T(0))); ((T(7), T(7)), (T(6), T(6))); ((T(6), T(1), T(6)), (T(4), T(2), T(4))); ((T(1), T(2), T(7), T(3), T(1)), (T(2), T(4), T(6), T(2))); …}

In step 6, each unique RMT sequence (also known as a graph in an NSRTD) is constructed by picking one compatible NSRS from each compatibility class $C_{i(i+1)}$ for each n CA cells. Here, S_1 and S_2 are graphs in the NSRTD made up of self-loop NSRSs only, and they are represented by graphs G_1 and G_2, respectively, in Figure 6. Further, S_3, S_4, S_5 and S_6 are graphs in the NSRTD made up of multi-length cycles, and they are represented by the graphs G_3, G_4, G_5 and G_6, respectively (Figure 6). In step 7, as the number of S_u identified is 6, the algorithm terminates and returns that the CA ⟨90, 150, 150, 90⟩ is not a maximal length CA.



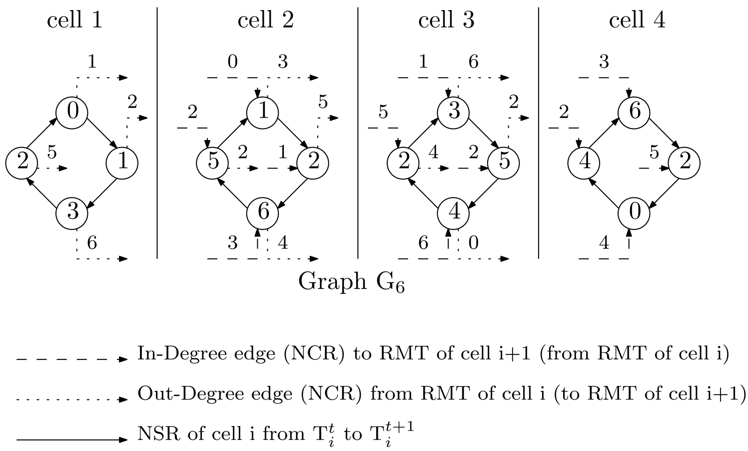


Figure 6. NSRTD for the CA ⟨90, 150, 150, 90⟩.

Discussion: The discussion part depicts how the NSRTD of the CA ⟨90, 150, 150, 90⟩ complies with its STD. Column 1 of Table 4 shows the number of graphs in the NSRTD. Here, we have a total of six graphs: G_1, G_2, G_3, G_4, G_5 and G_6 in the NSRTD (Figure 6). Columns 2 and 3 represent the RMT strings and the CA states corresponding to these RMT strings, respectively. Here, each of the graphs G_1 and G_2 is composed of a single path with single length cycles only, thereby representing a single RMT string each. Similarly, graphs G_3, G_4, G_5 and G_6, are composed of multi-length cycles only, representing 2, 4, 4 and 4 RMT strings, respectively. From Table 4, we can see that the CA states in column 3 comply with the states in the STD of the CA as shown in Figure 2(b). Here, graphs G_1 and G_2 comply with fixed-point attractors 0 and 15, respectively, whereas graph G_3 complies with the multi-length (2-length) cycle involving the states 6, 9; G_4 complies with the multi-length (4-length) cycle involving the states 1, 2, 7, 11; G_5 complies with the multi-length (4-length) cycle involving the states 4, 8, 13, 14; and G_6 complies with the multi-length (4-length) cycle involving the states 3, 5, 10, 12, respectively.

Theorem 1. A CA with rule vector $\mathrm{RV} = \langle R_1, R_2, \ldots, R_n \rangle$ can be considered as a potential candidate for a maximal length CA if and only if for each R_i, $i = 1, 2, \ldots, n$, there exists a self-loop NSRS with RMT T(0).

Proof. If there exists a self-loop NSRS with RMT T(0) in all the CA cells of rule vector $\mathrm{RV} = \langle R_1, R_2, \ldots, R_n \rangle$, then it will form an NSRTD graph G with single length cycles only (as the NCR of RMT T(0) is RMT T(0) itself). This corresponds to a single RMT string $(\mathrm{T}_1, \mathrm{T}_2, \ldots, \mathrm{T}_n) = (\mathrm{T}(0), \mathrm{T}(0), \ldots, \mathrm{T}(0))$, which complies with a fixed-point attractor with an all-zero state (a necessary condition for a maximal length CA). Hence, the proof. □

NSRTD Graph (1)	RMT String (2)				CA State (3)
	Cell 1	Cell 2	Cell 3	Cell 4	
Graph G_1	0	0	0	0	0000
Graph G_2	3	7	7	6	1111
Graph G_3	1	3	6	4	0110
	2	4	1	2	1001
Graph G_4	0	0	1	2	0001
	0	1	2	4	0010
	1	3	7	6	0111
	2	5	3	6	1011
Graph G_5	1	2	4	0	0100
	2	4	0	0	1000
	3	6	5	2	1101
	3	7	6	4	1110
Graph G_6	0	1	3	6	0011
	1	2	5	2	0101
	2	5	2	4	1010
	3	6	4	0	1100

Table 4. Compliance of NSRTD and STD for the CA ⟨90, 150, 150, 90⟩.

Theorem 2. A CA with rule vector RV $= \langle R_1, R_2, \ldots, R_n \rangle$ can be considered as a potential candidate for a maximal length CA if and only if for each R_i, $i = 1, 2, \ldots, n$, there exists a multi-length cycle NSRS involving all the RMTs.

Proof. If there exists a multi-length cycle NSRS involving all the RMTs in all the CA cells of rule vector RV $= \langle R_1, R_2, \ldots, R_n \rangle$, then it will form an NSRTD graph G with multi-length cycles only, representing 2^n RMT strings. Further, as discussed in Theorem 1, there will be an NSRTD graph representing RMT string (T(0), T(0), … , T(0)) in a maximal length CA. So, this RMT string will be excluded here. This will lead to the creation of NSRTDs with multi-length cycles, representing $2^n - 1$ RMT strings, which complies with multi-length cycle of length $2^n - 1$ comprised of all nonzero states (another necessary condition for a maximal length CA). Hence, the proof. □

Discussion: This paper explores and utilizes the NSRTD framework for the identification of maximal length CAs. There is a lot of work in the literature that employs maximal length CAs for modeling applications like random number generators [15, 21], cryptography [13], test pattern generators [22] and signature analysis [23]. Some of the unique properties of the maximal length CA that make it suitable for modeling and use in the above-mentioned applications are:

- Maximal length CAs can be used to produce long, deterministic output based upon the current CA state configuration and the rule vector used. The produced output, despite being deterministic, appears to be random and chaotic in nature.
- By definition, maximal length CAs produce a cycle of length 2^{n-1} for an n-cell CA. This means the CA state configuration must go on cycling through all possible configurations (except, in the case of an all-zero configuration) before a particular configuration is repeated. This maximizes the randomness of the generated sequence.
- The sequences generated by a maximal length CA also have good statistical properties, like uniform distribution and minimal correlation between the successive outputs.
- As a maximal length CA can easily be implemented in software or hardware with minimum resources, it inherently became a modeling tool for random number generation.

This paper identifies a maximal length CA that can be used further to model different CA-based applications by utilizing the unique property of a maximal length CA as discussed earlier.

5. Conclusion

This paper presents a theoretical framework based on the next state rule min term transition diagram (NSRTD) for characterizing maximal length cellular automata (CAs). The primary contribution is determining whether a cellular automaton (CA) exhibits maximal length behavior or not. In line with this, an algorithm was developed that can efficiently determine whether a given CA with n cells exhibits maximal length behavior in $O(2^n)$ time. Additionally, the developed algorithm benefits by facilitating the identification and elimination of non-potential CA candidates that fail to realize maximal length configurations in linear time. The proposed solution would be helpful for researchers who use maximal length CAs for devising applications in the field of cryptography, random number generation and test pattern generation. As a future extension, the NSRTD framework can be extended to analyze k-neighborhood CA (e.g., 5-neighborhood or 9-neighborhood CAs) to identify maximal length behavior. This would enable the study of richer dynamics and increased state complexity.

References

[1] J. von Neumann, *Theory of Self-Reproducing Automata* (A. W. Burks, ed.), Urbana, IL: University of Illinois Press, 1966.

[2] S. Wolfram, *Cellular Automata and Complexity: Collected Papers*, Reading, MA: Addison-Wesley Publishing Company, 1994.

[3] P. P. Chaudhuri, D. R. Chowdhury, S. Nandi and S. Chattopadhyay, *Additive Cellular Automata: Theory and Applications*, Vol. 1, Los Alamitos, CA: IEEE Computer Society Press, 1997.

[4] P. Dasgupta, S. Chattopadhyay and I. Sengupta, "An ASIC for Cellular Automata Based Message Authentication," in *Proceedings of 13th International Conference on VLSI Design (VLSI Design 2000)*, Science City, Calcutta, India, Los Alamitos, CA: IEEE, 2000 pp. 538–541. doi:10.1109/ICVD.2000.812663.

[5] S. Das, N. N. Naskar, S. Mukherjee, M. Dalui and B. K. Sikdar, "Characterization of CA Rules for SACA Targeting Detection of Faulty Nodes in WSN," in *9th International Conference on Cellular Automata for Research and Industry (ACRI 2010)*, Ascoli Piceno, Italy (S. Manzoni, H. Umeo and G. Vizzari, eds.), Berlin, Heidelberg: Springer, 2010 pp. 300–311. doi:10.1007/978-3-642-15979-4_32.

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

[7] N. Bacquey, "Complexity Classes on Spatially Periodic Cellular Automata," in *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014)*, Lyon, France (E. W. Mayr and N. Portier, eds.), Leibniz International Proceedings in Informatics, **25**, 2014 pp. 112–124. doi:10.4230/LIPIcs.STACS.2014.112.

[8] M. Dalui and B. K. Sikdar, "Design of Directory Based Cache Coherence Protocol Verification Logic in CMPs around TACA," in *International Conference on High Performance Computing & Simulation (HPCS 2013)*, Helsinki, Finland (W. W. Smari and V. Zeljkovic, eds.), Piscataway, NJ: IEEE, 2013 pp. 318–325. doi:10.1109/HPCSim.2013.6641433.

[9] M. Saha, M. Dalui and B. K. Sikdar, "A Cellular Automata Based Highly Accurate Memory Test Hardware Realizing March C^{-}," *Microelectronics Journal,* **52**, 2016 pp. 91–103. doi:10.1016/j.mejo.2016.03.009.

[10] M. Serra, T. Slater, J. C. Muzio and D. M. Miller, "The Analysis of One-Dimensional Linear Cellular Automata and Their Aliasing Properties," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **9**(7), 1990 pp. 767–778. doi:10.1109/43.55213.

[11] K. Cattell and J. C. Muzio, "Synthesis of One-Dimensional Linear Hybrid Cellular Automata," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **15**(3), 1996 pp. 325–335. doi:10.1109/43.489103.

[12] S. Adak and S. Mukherjee, "On Non-Linear Maximal Length Cellular Automata," in *Asian Symposium on Cellular Automata Technology (ASCAT 2023)*, Kolkata, India (S. Das and G. J. Martinez, eds.), Singapore: Springer, 2023 pp. 115–126. doi:10.1007/978-981-99-0688-8_10.

[13] S. Das and D. Roychowdhury, "Cryptographically Suitable Maximum Length Cellular Automata," *Journal of Cellular Automata*, **6**(6), 2011 pp. 439–459.

[14] W. Pries, A. Thanailakis and H. C. Card, "Group Properties of Cellular Automata and VLSI Applications," *IEEE Transactions on Computers*, **C-35**(12), 1986 pp. 1013–1024. doi:10.1109/TC.1986.1676709.

[15] S. Adak, "Maximal Length Cellular Automata," Ph.D. thesis, Indian Institute of Engineering Science and Technology, Shibpur, 2020. doi:10/13140/RG.2.2.11330.45766.

[16] P. H. Bardell, W. H. McAnney and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*, New York: Wiley, 1987.

[17] S. Adak, S. Mukherjee and S. Das, "Do there Exist Non-Linear Maximal Length Cellular Automata? A Study," in *13th International Conference on Cellular Automata for Research and Industry (ACRI 2018)*, Como, Italy (G. Mauri, S. El Yacoubi, A. Dennunzio, K. Nishinari and L. Manzoni, eds.), Cham: Springer, 2018 pp. 289–297. doi:10.1007/978-3-319-99813-8_26.

[18] M. Dalui, B. Chakraborty, N. Das and B. K. Sikdar, "NSRT Diagram for Identification of SACA and TACA Rules in Null-Boundary," *International Journal of Modern Physics C,* **33**(6), 2022 2250071. doi:10.1142/S0129183122500711.

[19] S. Hazra and M. Dalui, "Characterization of Single Length Cycle Two-Attractor Cellular Automata Using Next-State Rule Minterm Transition Diagram," *Complex Systems*, **31**(4), 2022 pp. 363–388. doi:10.25088/ComplexSystems.31.4.363.

[20] S. Banerjee, M. Dalui and S. Hazra, "A Generic NSRTD Framework for Characterizing Cellular Automaton Rules in Constant (0/1) Boundary Conditions," *Complex Systems*, **33**(1), 2024 pp. 61–86. doi:10.25088/ComplexSystems.33.1.61.

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

[22] A. K. Das and P. P. Chaudhuri, "Vector Space Theoretic Analysis of Additive Cellular Automata and Its Application for Pseudoexhaustive Test Pattern Generation," *IEEE Transactions on Computers*, **42**(3), 1993 pp. 340–352. doi:10.1109/12.210176.

[23] A. K. Das, D. Saha, A. R. Chowdhury, S. Misra and P. P. Chaudhuri, "Signature Analysers Based on Additive Cellular Automata," in *[1990] Digest of Papers. Fault-Tolerant Computing: 20th International Symposium*, Newcastle Upon Tyne, UK, IEEE, 1990 pp. 265–272. doi:10.1109/FTCS.1990.89374.