The set of reversible 90/150 cellular automata is regular

Palash Sarkar [a,*], Rana Barua [b]

[a] *Computer Science Unit, Indian Statistical Institute, 203, B.T. Road, Calcutta 700035, India*
[b] *Stat./Math. Unit, Indian Statistical Institute, 203, B.T. Road, Calcutta 700035, India*

Received 11 October 1996; received 31 October 1997; accepted 17 November 1997

Abstract

The reversibility problem for 90/150 cellular automata (both null and periodic boundary) is tackled using continuants and regular expressions. A 90/150 cellular automata can be uniquely encoded by a string over the alphabet $\{0,1\}$. It is shown that the set of strings which correspond to reversible 90/150 cellular automata is a regular set. We use the regular expression to enumerate the number of reversible strings of a fixed length. As a consequence, it is shown that given a polynomial $p(x)$, it is not always possible to get a 90/150 cellular automata whose transition matrix has characteristic polynomial $p(x)$.

Keywords: Continuants; Regular expressions; Finite automata; 90/150 cellular automata

1. Introduction

In this article we pose and solve two problems regarding a special class of matrices over F_2, the field of two elements. Let, $M_b, b \in \{0,1\}$, be a square matrix over F_2, having the following structure:

$$M_b = \begin{bmatrix} a_1 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & b \\ 1 & a_2 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & a_3 & 1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & & & & & & & & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 1 & a_{n-1} & 1 \\ b & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & a_n \end{bmatrix}. \tag{1}$$

First, assume $b = 0$. Then M_0 is uniquely specified by the string $a_1 \ldots a_n$ over the alphabet $\{0,1\}$ and is said to encode the matrix M_0. Now, consider the following problems:

(1) Obtain a characterization of the set of strings $a_1 \ldots a_n$ which encode non-singular matrices of the form M_0.

(2) Find the number of non-singular matrices M_0 of the order n.

We show that the set of strings which encode non-singular matrices is a regular set with a very simple structure. This solves the first problem. Using the 'canonical' regular expression for this set, we completely solve the second problem. It turns out that approximately two-thirds of the strings encode non-singular matrices of the form M_0. The novel features of our proof are the use of continuants for tackling the first problem and the use of regular expression for counting. We believe that these ideas can be profitably applied to other similar situations.

For the case $b=1$, the situation seems more complicated. However, using the results for matrices of the form M_0, we are able to satisfactorily solve the corresponding problems for matrices of the form M_1.

These problems arise very naturally in connection with the study of what are known as 90/150 cellular automata (CA). CA are simple discrete dynamical systems, capable of exhibiting quite complex behaviour (see [6, 11]). A finite one-dimensional CA is an array of cells, where each cell can assume *state* 0 or 1, which are regarded as elements of the 0 1 field F_2. It is an autonomous machine and evolves deterministically in discrete time steps. This evolution is effected by each cell changing its value according to a local rule. The *local rule* R_i for the ith cell c_i is a three variable boolean function and the state of c_i in the tth time step, denoted by x_i^t, is given by

$$x_i^t = R_i(x_{i-1}^{t-1}, x_i^{t-1}, x_{i+1}^{t-1}).$$

If the subscript i of x_i^t is taken modulo n, the number of cells of the CA, then the CA is called *periodic boundary* CA. If on the other hand, the array is considered to be placed between two cells having a fixed value zero, the CA is called *null boundary* CA. If all the R_i's are linear functions (SHIFT, EX-OR), then the CA is called *linear* (or additive) and the *global rule* is a linear transformation of the vector space F_2^n into itself that yields the *configuration* at the next time step during the evolution of the CA.

Here we depart from the more usual definition of CA, where the local rule is same for all cells, as has been studied in [6, 11]. The kind of CA that we study allows each cell to have its own local rule, and hence is called *hybrid* CA. This class of CA is more important from the VLSI applications point of view [5, 8, 9]. Henceforth, by CA we will mean hybrid CA.

Linear CA have been proposed as a basic structure in several areas of VLSI design [5, 8, 9]. In fact, the most useful structure from the VLSI point of view is a 90/150 structure where the local rule R_i is given by

$$x_i^t = R_i(x_{i-1}^{t-1}, x_i^{t-1}, x_{i+1}^{t-1}) = x_{i-1}^{t-1} + a_i x_i^{t-1} + x_{i+1}^{t-1} \quad (1 \leqslant i \leqslant n),$$

where $a_i \in \{0,1\}$ and addition is modulo 2 i.e. over F_2. If $a_i = 0$, R_i is called rule 90, else R_i is called rule 150 (see [11] for a nomenclature of local rules). The global rule of such a CA is specified by the matrix (called its *transition matrix*) M_b of (1), where b is 0 or 1 accordingly as there is a null or periodic boundary condition. Once the boundary condition is fixed the string $a_1 \ldots a_n$ over $\{0,1\}$ completely specifies the

structure of the CA, and hence, we shall identify the CA and its transition matrix with the string $a_1 \ldots a_n$. The CA is *reversible* iff its transition matrix is non-singular.

The transition matrix for null boundary 90/150 CA is M_0 (from (1)). The determinant of M_0 can be elegantly expressed in terms of multivariate polynomials called continuants, which were first introduced and studied by Euler [2]. A continuant in n variables $K_n(x_1, \ldots, x_n)$ is defined by the following recurrence:

$$\begin{aligned} &K_0(\,) = 1, \qquad K_1(x_1) = x_1, \\ &K_n(x_1, \ldots, x_n) = x_1 K_{n-1}(x_2, \ldots, x_n) + K_{n-2}(x_3, \ldots, x_n). \end{aligned} \tag{2}$$

In fact, the continuants satisfy a more general recurrence [2, p. 289]

$$\begin{aligned} &K_{m+n}(x_1, \ldots, x_m, x_{m+1}, \ldots, x_{m+n}) \\ &\quad = K_m(x_1, \ldots, x_m) K_n(x_{m+1}, \ldots, x_{m+n}) \\ &\qquad K_{m-1}(x_1, \ldots, x_{m-1}) K_{n-1}(x_{m+2}, \ldots, x_{m+n}) \end{aligned} \tag{3}$$

and using the relation in [2, p. 304], we have

$$K_n(a_1, \ldots, a_n) = \det M_0.$$

Also the characteristic polynomial of M_0 is $K_n(x + a_1, \ldots, x - a_n)$ (note that over F_2, $-1 = -1$). Hence, M_0 is non-singular iff $K_n(a_1, \ldots, a_n) \neq 1$. Expanding M_0 by the first and the last row it is easy to see that

$$K_n(a_1, \ldots, a_n) = K_n(a_n, \ldots, a_1). \tag{4}$$

Thus, it is most natural to consider continuants in the analysis of 90/150 null boundary CA and we know of no other place where this has been done. In fact, the problems that we have posed can be framed entirely in terms of continuants, and hence our work can also be considered to be a contribution to the study of continuants. We would like to point out that the use of continuants in Sections 2 and 3 is not really necessary. However, in Section 4, we use (3) in an essential way which clearly highlights the importance of continuants in the present setting.

Finally, we point out the implications of our results to the theory of linear finite-state machines. The counting results show that certain kinds of linear machines cannot be synthesised using 90/150 CA.

In what follows, all arithmetic is over F_2 and ε will denote the empty string. Also, $x|$ denotes the length of a string x, and the cardinality of a set S is denoted by $|S|$.

2. Null boundary CA

As stated in the introduction, the characteristic polynomial of the transition matrix of a null boundary 90/150 CA is a continuant $K_n(x+a_1, \ldots, x+a_n)$. The CA is reversible iff the constant term of $K_n(x + a_1, \ldots, x + a_n)$ is 1. The constant term is obtained by

putting $x=0$ and is equal to $K_n(a_1,\ldots,a_n)$. Since the CA is uniquely identified by the string $a_1\ldots a_n$ over $\{0,1\}$, we will write that the string $a_1\ldots a_n$ is reversible to mean that the corresponding CA is reversible. First note that the empty string ε is reversible. Next, we have the following:

Lemma 2.1. *Let $y\in\{0,1\}^*$ and $i\in\{0,1\}$. Then*

(a) *$0iy$ is reversible iff y is reversible.*
(b) *$10y$ is reversible iff $1y$ is reversible.*
(c) *$11y$ is reversible iff $0y$ is reversible.*

Proof. (a) Using (3), we can write,

$$K_n(0,i,a_3,a_4,\ldots,a_n)=K_2(0,i)K_{n-2}(a_3,a_4,\ldots,a_n)+K_1(0)K_{n-3}(a_4,a_5,\ldots,a_n).$$

Now, $K_2(0,i)=0.i+1=1$.
Therefore, $K_n(0,i,a_3,a_4,\ldots,a_n)=K_{n-2}(a_3,a_4,\ldots,a_n)$. This proves (a).

(b) $K_n(1,0,a_3,\ldots,a_n)$

$$\begin{aligned}&-K_2(1,0)K_{n-2}(a_3,\ldots,a_n)+K_1(1)K_{n-3}(a_4,\ldots,a_n) \quad \text{by (3)}\\ &=K_{n-2}(a_3,\ldots,a_n)+K_{n-3}(a_4,\ldots,a_n)\\ &=K_{n-1}(1,a_3,\ldots,a_n) \quad \text{by (2)}.\end{aligned}$$

This proves (b).
(c) is similar to (b). □

Given a string y we can repeatedly 'reduce' it from the left to obtain shorter strings which are reversible iff the original string is reversible. To formalise this, for any two strings u,v we write $u\to v$ and say u reduces to v if

(1) $u=0iv$, $i\in\{0,1\}$ or
(2) $u=10x$ and $v=1x$ or
(3) $u=11x$ and $v=0x$.

Note that if $u\to v$ then $|v|<|u|$. By abuse of notation, we will write $u\to v$ (and also say u reduces to v) if there exist strings $u_0,\ldots,u_n$ such that

$$u=u_0\to u_1\to\cdots\to u_n=v.$$

Remark 2.1. Similar reduction from the right is also possible.

The irreducible strings are simple to characterize.

Proposition 2.1. *Let $y\in\{0,1\}^*$. Then y reduces in zero or more steps to exactly one of the strings in $\{\varepsilon,0,1\}$. Furthermore, y is reversible iff y can be reduced to either ε or 1.*

Proof. By the reduction rules, any string of length $\geqslant 2$ can be reduced. Hence, the only irreducible strings are $\{\varepsilon, 0, 1\}$. That the reduction is unique follows from the fact that at any stage at most one of the rules apply. The last statement holds since by Lemma 2.1, any reduction preserves reversibility. □

From this we get the following linear time algorithm for determining reversible null boundary 90/150 CA (see [10] for algorithms to determine reversibility of other kinds of CA).

Algorithm $\mathscr{A}$

input: A string $x = a_1 \ldots a_n$ over $\{0,1\}$
output: 'yes' if x is reversible, else 'no'
while (x not in $\{\varepsilon, 0, 1\}$) do
  if $((x = 0\,0\,y)$ or $(x = 0\,1\,y)$ then $x = y$
  else if $(x = 1\,0\,y)$ then $x = 1y$
  else if $(x = 1\,1\,y)$ then $x = 0y$
or
if ($x = \varepsilon$ or $x = 1$) then output 'yes' else output 'no'

Using the idea of reversibility preserving reduction, one can obtain a deterministic finite automata (DFA) to recognize all reversible strings. Since any initial prefix of the string can be reduced, all that the DFA has to do is to remember the effective (from the point of reversibility) amount of input seen so far. More formally, let $M = (\{0,1\}, Q, s_\varepsilon, \delta, F)$ be a DFA, where

(1) $Q = \{s_\varepsilon, s_0, s_1\}$ is the set of states.
(2) The transition function δ is defined as follows. Let $i \in \{0,1\}$. Then,
  (a) $\delta(s_\varepsilon, i) = s_i$,
  (b) $\delta(s_0, i) = s_i$,
  (c) $\delta(s_1, 0) = s_1$,
  (d) $\delta(s_1, 1) = s_0$.
(3) $F = \{s_\varepsilon, s_1\}$ is the set of final states.

The state s_ε correspond to the empty string, and any state $q \in Q$ remembers the effective amount of input seen so far. The transition function δ specifies the reduction rules. So we get the following:

Theorem 2.1. *Let $\mathscr{L}(M)$ be the language accepted by the DFA M. Then $y \in \mathscr{L}(M)$ iff y correspond to a reversible null boundary CA.*

Next, we obtain the corresponding regular expression. Let $\boldsymbol{R}, \boldsymbol{R}_0, \boldsymbol{R}_1$, respectively, correspond to the regular expressions for s_ε, s_0, s_1. Then we get

$$\boldsymbol{R} = \boldsymbol{R}_0(1 \mid 0) + \epsilon,$$

$$\boldsymbol{R}_0 = \boldsymbol{R}\boldsymbol{0} + \boldsymbol{R}_1\boldsymbol{1},$$

$$\boldsymbol{R}_1 = \boldsymbol{R}_1\boldsymbol{0} + \boldsymbol{R}\boldsymbol{1}.$$

We can solve this set of equations using Arden's lemma [4, p. 54], which states that for regular expressions P, Q, R if $R = P - RQ$, then $R - PQ^*$. So by a sequence of simple manipulations, we get

$$R = ((0 + 10^*1)(1 - 0))^*, \qquad R_1 = R10^*, \qquad R_0 - R(0 + 10^*1)$$

and the regular expression for $\mathscr{L}(M)$ is $R \mid R_1$. This leads us to the following:

Theorem 2.2. *The regular expression for the set of all reversible strings which correspond to null boundary CA is given by* $\alpha + \alpha 10^*$, *where* $\alpha = ((0 + 10^*1)(1 + 0))^*$.

Given this regular expression, it is possible to enumerate the number of reversible strings of length n. Let S denote the set of reversible strings. Then, $S = L_\varepsilon \cup L_1$, where L_ε (resp. L_1) is the set of all strings which reduces to ε (resp. 1). From Proposition 2.1, $L_\varepsilon \cap L_1 = \phi$. Let $S^{(n)}$, $L_\varepsilon^{(n)}$, $L_1^{(n)}$ denote the subset of all strings of length n belonging to S, L_ε, L_1, respectively. Then, $|S^{(n)}| = |L_\varepsilon^{(n)}| + |L_1^{(n)}|$. Next, we prove

Proposition 2.2. *For* $n \geqslant 0$, $|S^{(n)}| = \sum_{i=0}^{n} L_\varepsilon^{(i)}|$.

Proof. The regular expression for L_1 is $\alpha 10^*$ where α is the regular expression for L_ε. Let $x \in L_1^{(n)}$ be such that the last zero is chosen $i \geqslant 0$ times. Then $x = y10^i$ where $y \in L_\varepsilon^{(n-1-i)}$. Conversely, for any $y \in L_\varepsilon^{(n-1-i)}$ we get an unique $x \in L_1^{(n)}$. Therefore,

$$|L_1^{(n)}| = |L_\varepsilon^{(n-1)}| + |L_\varepsilon^{(n-2)}| + \cdots + |L_\varepsilon^{(0)}|.$$

Hence,

$$|S^{(n)}| = |L_\varepsilon^{(n)}| + |L_1^{(n)}| = \sum_{i=0}^{n} |L_\varepsilon^{(i)}|. \qquad \square$$

So the problem reduces to computing $|L_\varepsilon^{(n)}|$. It turns out that $|L_\varepsilon^{(n)}|$ satisfies a nice recurrence relation.

Lemma 2.2. $|L_\varepsilon^{(0)}| = 1$, $|L_\varepsilon^{(1)}| = 0$

$$|L_\varepsilon^{(n)}| = |L_\varepsilon^{(n-1)}| + 2|L_\varepsilon^{(n-2)}| \quad \text{for } n \geqslant 2.$$

Proof. Let $x \in L_\varepsilon^{(n)}$. If $|x| < 2$, then it is easy to see that $|L_\varepsilon^{(0)}| = 1$ and $|L_\varepsilon^{(1)}| = 0$. So for $|x| \geqslant 2$, x can be written as $x = aby$ where $|y| = n - 2$.

Case 1: $ab = 00$ or $ab = 01$. Then we have $x \to y$. So x reduces to ε iff y reduces to ε. Hence for each reversible string $y \in L_\varepsilon^{(n-2)}$, we get two strings in $L_\varepsilon^{(n)}$.

Case 2: $ab = 10$ or $ab = 11$.

If $ab = 10$, then x reduced to ε iff $1y$ reduces to ε.

If $ab = 11$, then x reduces to ε iff $0y$ reduces to ε.

So for each string in $L_\varepsilon^{(n-1)}$ there exists exactly one string in $L_\varepsilon^{(n)}$ and all strings in $L_\varepsilon^{(n)}$ arise as Cases 1 or 2. Hence,

$$|L_\varepsilon^{(n)}| = |L_\varepsilon^{(n-1)}| - 2|L_\varepsilon^{(n-2)}|. \qquad \square$$

Corollary 2.1. *For $n \geqslant 2$,*

(1) $L_\varepsilon^{(n)} = 2\sum_{i=0}^{n-2} |L_\varepsilon^{(i)}|$.

(2) $|S^{(n)}| = \frac{3}{2}|L_\varepsilon^{(n)}| - |L_\varepsilon^{(n-1)}|$.

Proof. (1) Follows from the above lemma by induction.

(2) Follows from (1) and Proposition 2.2. $\square$

The next step is to obtain an expression for $L_\varepsilon^{(n)}$ via its generating function.

Lemma 2.3. *For $n \geqslant 0$, $|L_\varepsilon^{(n)}|$ is the coefficient of x^n in*

$$G(x) = \frac{1-x}{1-x-2x^2}$$

and hence is given by

$$L_\varepsilon^{(n)}| = \tfrac{2}{3}[2^{n-1} + (-1)^n].$$

Proof. The generating function is obtained by standard manipulations and hence we shall omit it. To see the second statement, note that

$$G(x) = \frac{1-x}{1-x-2x^2} = \frac{1}{3}\left[\frac{2}{1+x} + \frac{1}{1-2x}\right].$$

Hence, the coefficient of x^n in $G(x)$ is

$$\tfrac{1}{3}[2\cdot(-1)^n + 2^n] = \tfrac{2}{3}[2^{n-1} + (-1)^n]. \qquad \square$$

We finally obtain:

Theorem 2.3. *For $n \geqslant 0$,*

$$|S^{(n)}| = \tfrac{1}{3}[2^{n-1} + (-1)^n].$$

Consequently, $|S^{(n)}|$ satisfies the following recurrence:

$$|S^{(0)}| = 1 \quad \textit{and} \quad |S^{(n)}| = 2|S^{(n-1)}| + (-1)^n \textit{ for } n \geqslant 1.$$

Proof. For $n = 0, 1$ it is easy to check that $|S^{(n)}| = 1$. From Corollary 2.1 for $n \geqslant 2$,

$$|S^{(n)}| = \tfrac{3}{2}|L_\varepsilon^{(n)}| + |L_\varepsilon^{(n-1)}|.$$

By the above lemma, $|L_e^{(n)}| = \frac{2}{3}[2^{n-1} + (-1)^n]$. Hence,

$$|S^{(n)}| = \tfrac{3}{2}[\tfrac{2}{3}(2^{n-1} + (-1)^n)] + \tfrac{2}{3}[2^{n-2} + (-1)^{n-1}] - \tfrac{1}{3}[2^{n-1} - (-1)^n]. \quad \square$$

Remark 2.2. Approximately two thirds of all strings of length n are reversible.

3. Periodic boundary CA

Next, we turn to the characterization of periodic boundary CA. The transition matrix for such a CA is of the following form:

$$M_1 = \begin{bmatrix} a_1 & 1 & 0 & 0 & \cdots & 0 & 1 \\ 1 & a_2 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & a_3 & 1 & \cdots & 0 & 0 \\ . & . & . & . & \cdots & . & . \\ . & . & . & . & \cdots & & 1 \\ 1 & 0 & . & . & \cdots & 1 & a_n \end{bmatrix}.$$

In analogy with continuants, let us denote the determinant of M_1 by $P_n(a_1, \ldots, a_n)$. We will only consider $n \geqslant 2$. Then we have the following:

Proposition 3.1. *$P_n(a_1, \ldots, a_n) = K_n(a_1, \ldots, a_n) + K_{n-2}(a_2, \ldots, a_{n-1})$. Consequently, $a_1 \ldots a_n$ is reversible under periodic boundary condition iff exactly one of $a_1 \ldots a_n$ and $a_2 \ldots a_{n-1}$ is reversible under null boundary condition.*

Proof. Expanding the determinant by the first row, we get,

$$\begin{aligned} &P_n(a_1, \ldots, a_n) \\ &= a_1 K_{n-1}(a_2, \ldots, a_n) \\ &\quad + \begin{vmatrix} 1 & 1 & \cdots & 0 & 0 \\ 0 & a_3 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ . & . & \cdots & . & . \\ . & . & \cdots & . & . \\ 0 & 0 & \cdots & . & 1 \\ 1 & 0 & \cdots & 1 & a_n \end{vmatrix} + \begin{vmatrix} 1 & a_2 & 1 & 0 & \cdots & 0 \\ 0 & 1 & a_3 & 1 & \cdots & 0 \\ . & . & . & . & \cdots & . \\ . & . & . & . & \cdots & . \\ . & . & . & . & \cdots & . \\ . & . & . & . & \cdots & a_{n-1} \\ 1 & 0 & . & . & \cdots & 1 \end{vmatrix} \\ &= a_1 K_{n-1}(a_2, \ldots, a_n) - K_{n-2}(a_3, \ldots, a_n) + 1 + K_{n-2}(a_2, \ldots, a_{n-1}) + 1 \end{aligned}$$

(by expanding each of the two determinants by the first column)

$$= K_n(a_1, \ldots, a_n) + K_{n-2}(a_2, \ldots, a_{n-1}),$$

by (2) and the fact that all operations are over F_2.

Consequently, under periodic boundary condition, $a_1,\ldots,a_n$ is reversible iff $P_n(a_1,\ldots,a_n)=1$, i.e. iff exactly one of $K_n(a_1,\ldots,a_n)$ and $K_{n-2}(a_1,\ldots,a_{n-1})$ is 1, i.e. iff exactly one of $a_1,\ldots,a_n$ and $a_2,\ldots,a_{n-1}$ is reversible under null boundary condition. □

Remark 3.1. (1) The continuant $K_n(a_1,\ldots,a_n)$ can be obtained by the following simple rule [2]. Start with the term $a_1a_2\ldots a_n$ and then cancel out pairs a_ka_{k+1} in all possible ways. From the above proposition a similar rule holds for $P_n(a_1,\ldots,a_n)$ with the following modification. When considering pairs a_ka_{k-1}, consider a_na_1 to be one such pair, i.e. consider the terms $a_1,\ldots,a_n$ to be arranged in a circle.

(2) The expression $P_n(a_1,\ldots,a_n)$ is invariant under a circular shift of its arguments.

Based on the above proposition we can construct a DFA G to recognize all the possible strings which correspond to reversible periodic boundary CA. The idea is to run two DFAs M_1 and M_2 in parallel, where M_1 and M_2 are copies of the DFA for recognizing reversible null boundary CA. The DFA M_1 will run on the entire string $a_1\ldots a_n$ while DFA M_2 will effectively run only on $a_2\ldots a_{n-1}$. Then we accept iff exactly one of M_1 and M_2 accepts. It is easy to design G such that M_2 skips the first symbol, i.e. a_1. When G reads a_i, $i>1$, it makes a transition from q_i to q_{i+1} in M_1 and from p_1 to p_2 in M_2, following the rules of Lemma 2.1. Skipping the last symbol is a bit more tricky since G cannot know that a_n is the last symbol until it has read it. To tackle this we allow the control of G to have one more bit of memory (say b), which is used in the following way. When G makes a transition from p_1 to p_2 in M_2, it puts a value of 1 in b if p_1 was an accepting state for M_2 else it puts a value of 0 in b. So at the end of the input b indicates whether $a_2\ldots a_{n-1}$ was an accepting string for M_2. Then G accepts iff either b is 1 and M_1 is in a rejecting state or b is 0 and M_1 is in an accepting state.

So from this description we get

Theorem 3.1. *The set of all strings which correspond to reversible periodic boundary CA, form a regular set.*

Consequently, there exists a linear time algorithm to determine reversibility of periodic boundary 90/150 CA.

Proof. We provide a formalization of the above description.

Let $M=(\{s_\varepsilon,s_0,s_1\},s_\varepsilon,\delta,\{s_\varepsilon,s_1\})$ be the DFA for the null boundary CA. Let

$$r:\{\varepsilon,0,1\}\to\{0,1\},\quad \text{where } r(\varepsilon)=r(1)=1,\ \text{and } r(0)=0.$$

Define $G=(Q_p,s,\delta_p,F_p)$, to be a DFA, where

(1) $Q_p=\{s,s_0,s_1\}\cup\{s_\varepsilon,s_0,s_1\}\times\{s_\varepsilon,s_0,s_1\}\times\{0,1\}$,

(2) Let $i,j\in\{0,1\}$ and $x,y\in\{0,1,a\}$.

(a) $\delta_p(s,i)=s_i$,

(b) $\delta_p(s_0,i)=(s_\varepsilon,s_i,1)$,

(c) $\delta_p(s_1, 0) = (s_1, s_0, 1)$,

(d) $\delta_p(s_1, 1) = (s_0, s_1, 1)$,

(e) $\delta_p((s_x, s_y, i), j) = (s_{\delta(x,j)}, s_{\delta(y,j)}, r(y))$,

(3) $F = \{(s_x, s_y, i)\colon r(x) \neq i\}$.

It is easy to see that G formalizes the DFA described above and G accepts a string x iff x correspond to a reversible periodic boundary 90/150 CA. □

We now enumerate the number of strings which correspond to reversible periodic boundary 90/150 CA. In this case the regular expression is more complicated, so we use the results for null boundary CA.

Let $T^{(n)}$ be the set of all strings of length n which correspond to reversible periodic boundary CA. From Proposition 3.1, $T^{(n)}$ can be written as

$$T^{(n)} = A^{(n)} \cup B^{(n)} \cup C^{(n)} \cup D^{(n)},$$

where

$$A^{(n)} = \{x \in T^{(n)}\colon x = azb,\ a, b \in \{0,1\} \text{ and } x \to \varepsilon,\ z \to 0\},$$

$$B^{(n)} = \{x \in T^{(n)}\colon x = azb,\ a, b \in \{0,1\} \text{ and } x \to 1,\ z \to 0\},$$

$$C^{(n)} = \{x \in T^{(n)}\colon x = azb,\ a, b \in \{0,1\} \text{ and } x \to 0,\ z \to \varepsilon\},$$

$$D^{(n)} = \{x \in T^{(n)}\colon x = azb,\ a, b \in \{0,1\} \text{ and } x \to 0,\ z \to 1\}$$

and $A^{(n)}$, $B^{(n)}$, $C^{(n)}$, $D^{(n)}$ are pairwise disjoint. Hence,

$$|T^{(n)}| = |A^{(n)}| + |B^{(n)}| + |C^{(n)}| + |D^{(n)}|. \tag{5}$$

Next, we prove two results which are crucial for enumerating $|T^{(n)}|$.

Proposition 3.2. *Let $v \in \{0, 1, \varepsilon\}$. Then there does not exist strings y ($|y| \geq 2$) such that $y = ax$, $a \in \{0, 1\}$ and both $y \to v$ and $x \to v$.*

Proof. We will only prove the result for $v = 0$. The other two cases are similar. We prove by induction (on the length of strings) that there does not exist strings z such that, $z \to 0$ and $az \to 0$.

Base step: For $|z| = 0$, $z = \varepsilon$ and the result is easy.

Inductive step: Suppose that the result holds for all strings of length less than n.

Let $|z| = n$. Suppose if possible $z \to 0$ and $az \to 0$. Recall that the regular expression for the strings reducing to 0 is $\alpha(0 + 10^*1)$, where α is the regular expression for strings reducing to ε. Since $z \to 0$, either

(1) $z = y0$ and $az = ay0$ or,

(2) $z = y10^i1$ and $az = ay10^i1$

with $|y| < |z|$.

Now, in both cases $y \to \varepsilon$ and $ay \to \varepsilon$. For Case 1 this is clear. In Case 2, if $ay \to 0$, then $az \to \varepsilon$ or $az \to 1$ according as i is odd or i is even. If on the other hand, $ay \to 1$, then $az \to \varepsilon$ or $az \to 1$ according as i is even or odd. Since $az \neq 0$ it follows that ay must reduce to ε.

If $|y| = 0$ then we immediately have a contradiction. So suppose $|y| > 0$. Now, $y \to \varepsilon$ implies $y = wc$ ($c \in \{0, 1\}$) such that $w \to 0$ and $ay \to \varepsilon$ implies $aw \to 0$, where $0 \leqslant |w| < |y| < |z|$.

By the induction hypothesis such w does not exist. Hence, the proof. □

Proposition 3.3. *For $n \geqslant 2$, let*

$$X_0^{(n)} = \{y \in L_\varepsilon^{(n)} : y = ax \text{ and } x \to 0\},$$
$$X_1^{(n)} = \{y \in L_\varepsilon^{(n)} : y = ax \text{ and } x \to 1\}.$$

Then, $|X_0^{(n)}| = |X_1^{(n)}| = \frac{1}{2}|L_\varepsilon^{(n)}|$.

Proof. Let $y \in L_\varepsilon^{(n)}$ with $y = ax$.

Let $x = zb$ so that $y = azb \to \varepsilon$. Now, $az0 \to \varepsilon$ iff $az1 \to \varepsilon$. So the strings in $L_\varepsilon^{(n)}$ can be paired as $az0$ and $az1$. Then exactly one of the strings $z0$ and $z1$ reduces to 1 and the other reduces to 0. (By Proposition 3.2 none can reduce to ε).

Hence, $|X_0^{(n)}| = |X_1^{(n)}| = \frac{1}{2}|L_\varepsilon^{(n)}|$. □

Now, we can find the cardinalities of $A^{(n)}, B^{(n)}, C^{(n)}, D^{(n)}$. Following [2], we will let $[\phi]$ denote the value of a boolean predicate ϕ.

Lemma 3.1. *For all $n \geqslant 2$,*

(1) $|A^{(n)}| = 0$,

(2) $|B^{(n)}| = [2 \not| n] = \frac{1}{2}\sum_{i=1}^{n-1} L_\varepsilon^{(i)}|$,

(3) $|C^{(n)}| = [2 | n] + \frac{1}{2}\sum_{i=1}^{n-2} L_\varepsilon^{(i)}$,

(4) $|D^{(n)}| = \frac{1}{2}|L_\varepsilon^{(n-1)}|$.

Proof. (1) This is proved by proving that $A^{(n)} = \phi$. To see this first note that $x \subset A^{(n)}$ iff $x = azb \to \varepsilon$ and $z \neq 0$. But $azb \to \varepsilon$ implies $az \to 0$, hence $x \subset A^{(n)}$ iff there exists string z such that $az \neq 0$ and $z \to 0$. But by Proposition 3.2 such strings do not exist.

(2) In this case $x \subset B^{(n)}$ iff $x = azb \to 1$ and $z \neq 0$.

If $b = 1$, $az \to \varepsilon$ and $z \to 0$. There are $\frac{1}{2}|L_\varepsilon^{(n-1)}|$ such strings (by Proposition 3.3).

If $b = 0$, then two cases arise

(a) $z = 0^{n-2}$, $a = 1$ where $10^{n-1} \to 1$ and $0^{n-2} \to 0$. But then $n - 2$ and hence n must be odd. This contributes the term $[2 \not| n]$ to $|B^{(n)}|$.

(b) $z = y10^i$ where $0 \leqslant i \leqslant n - 3$ and both $ay10^i \to 1$ and $y10^i \to 0$. Therefore $ay \to \varepsilon$ and $y \to c$ for some $c \in \{0, 1\}$. By Proposition 3.3 there are $\frac{1}{2}L_\varepsilon^{(n-2-i)}$ such strings.

So, $|B^{(n)}| = \lfloor 2^n/n \rfloor + \frac{1}{2}\sum_{i=1}^{n-1} L_i^{(i)}$.
(3) and (4) are similar to above. □

So finally, we get the following:

Theorem 3.2. *For all* $n \geqslant 2$,

$$|T^{(n)}| = |S^{(n-1)}| = \tfrac{1}{3}[2^n + (-1)^{n-1}].$$

Proof. Using the above lemma and (5),

$$\begin{aligned} |T^{(n)}| &= |A^{(n)}| + |B^{(n)}| + |C^{(n)}| + |D^{(n)}| \\ &= \sum_{i=0}^{n-1} |L_i^{(i)}| = |S^{(n-1)}| = \tfrac{1}{3}[2^n + (-1)^{n-1}]. \end{aligned}$$ □

Remark 3.2. $|T^{(n)}|$ is approximately half of $|S^{(n)}|$ and one-third of the total number of binary strings of length n.

4. Linear finite-state machines

In this section we point out the consequences of our results to the synthesis problem for CA. CA belong to the class of linear finite-state machines (LFSM). The most popular examples of LFSMs are the linear feedback shift registers (LFSR), which have been quite extensively studied [3]. A LFSM is completely characterized by its characteristic polynomial, which defines the linear recurrence satisfied by the output bits of the machine. A CA being an autonomous machine, there is no concept of output. However, the successive states of any particular cell (usually one of the end cells) can be considered to be its output. Next, we point out the relationship between the characteristic polynomial of the transition matrix of a CA and the linear recurrence satisfied by the output bits of any particular cell. To do that we need the following [9]:

Lemma 4.1. *Let M be the transition matrix for a* 90/150 *null boundary CA. Then M is nonderogatory, i.e. the minimal polynomial for M is the same as the characteristic polynomial for M.*

Now, we prove the following:

Proposition 4.1. *Let M be the transition matrix of a* 90/150 *null boundary CA and let*

$$p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_0,$$

be its characteristic polynomial. Then there exists a vector x, such that the temporal sequence of any cell of the corresponding CA loaded with initial configuration x, satisfies the linear recurrence defined by $p(x)$, i.e.

$$a_i^t = c_{n-1}a_i^{t-1} - \cdots + c_0 a_i^{t-n} \quad \text{for } t \geqslant n.$$

Proof. Let x be any nonnull vector and $q(x)$ be its minimal polynomial, i.e. the polynomial of the least degree such that $q(M)x = \mathbf{0}$.

Then $q(x) \mid p(x)$ and the output of any cell of the CA loaded with initial configuration x will satisfy the linear recurrence defined by $q(x)$.

By the above lemma, $p(x)$ is the minimal polynomial for M, and hence, there exists a vector x, whose minimal polynomial is $p(x)$. Therefore the result follows. □

Given this result it is easy to see that any two CA having the same characteristic polynomial will essentially generate the same bit sequence (modulo a shift).

Given any bit sequence it is possible to synthesize a minimum length LFSR whose output is the given sequence. This is done by the famous Berlekamp–Massey shift register synthesis Algorithm [1, 7]. The algorithm essentially finds the least-degree polynomial which defines a linear recurrence satisfied by the given bit sequence. Designing a LFSR from this polynomial is trivial. So the natural question to ask in the context of CA is the following:

Given any bit sequence can we design a 90/150 *CA whose output is the given bit sequence and the number of cells in the CA is equal to the number of cells in the minimum length LFSR which generates the same bit sequence*?

Unfortunately, the answer to this question is no and it follows from the fact that the answer to the following related question is also no:

Given any polynomial $p(x)$ of degree n, can we get an n-cell 90/150 *CA whose transition matrix has characteristic polynomial $p(x)$*?

For the following, let us decide to call a polynomial (and the corresponding LFSM) reversible iff its constant term is 1. So there are exactly 2^{n-1} reversible polynomials of degree n. A CA will be said to realize an LFSM characterized by a polynomial $p(x)$ iff the characteristic polynomial of its transition matrix is $p(x)$. Then we get the following:

Proposition 4.2. *Using* 90/150 *null boundary CA, it is not possible to realize all irreversible LFSMs.*

Proof. The number of reversible strings of length n is $|S^{(n)}|$, and hence, the number of irreversible strings is $2^n - |S^{(n)}| = \frac{1}{3}(2^n + (-1)^{n+1}) = |S^{(n-1)}|$.

The total number of irreversible machines is 2^{2^n-1} and the result follows from the fact that for $n \geqslant 2$, $S^{(n-1)}| < 2^{n-1}$. □

Using a similar argument it is possible to prove,

Proposition 4.3. *Using* 90/150 *periodic boundary CA, it is not possible to realize all reversible LFSMs.*

Since approximately two-thirds of all strings of length n correspond to reversible null boundary 90/150 CA and there are only 2^{n-1} reversible polynomials, one might expect that using null boundary 90/150 CA it is possible to realize all reversible LFSMs. However, this is not true and to prove it requires a more delicate argument. First note that it is possible for two CAs to have the same characteristic polynomial. If $a_1 \ldots a_n$ encodes a CA, then $K_n(x-a_1,\ldots,x+a_n)$ is its characteristic polynomial and since $K_n(x+a_1,\ldots,x+a_n)=K_n(x+a_n,\ldots,x+a_1)$ (from (4)), the CA encoded by $a_n \ldots a_1$ also has the same characteristic polynomial. Of course if $a_1 \ldots a_n$ is a palindrome, i.e. $a_i=a_{n-i}$ for all i, then this is trivially true. Otherwise, we have two distinct CAs with the same characteristic polynomial. Let $D^{(n)}$ be the set of all reversible palindromes of length n. Define,

$$A_n = 2^{n-1} - |D^{(n)}|,$$

$$B_n = \tfrac{1}{2}(|S^{(n)}| - |D^{(n)}|).$$

Then there are at least A_n reversible polynomials which are not realized by reversible palindromic strings and there are at most B_n reversible polynomials which are realized by reversible non palindromic strings. So if we can prove that $B_n < A_n$, then we are done. We proceed by first finding $|D^{(n)}|$.

Lemma 4.2. *For $n \geqslant 2$, $|D^{(n)}| = 2^{\lfloor n/2 \rfloor - 1} + |L_0^{(\lfloor n/2 \rfloor - 1)}|$, where $L_0^{(n)}$ is the set of all strings of length n which reduce to 0.*

Proof. We will prove the result for odd n. The result for even n is similar.

Let $n=2k+1$. Since n is odd any palindromic string x will have the following form,

$$x = a_1 \ldots a_k a_{k+1} a_k \ldots a_1.$$

Now, let us find the conditions under which x is reversible. We use (3) to get

$$\begin{aligned} &K_n(a_1,\ldots,a_k,a_{k+1},a_k,\ldots,a_1) \\ &\quad = K_k(a_1,\ldots,a_k)K_{k-1}(a_{k+1},a_k,\ldots,a_1) + K_{k-1}(a_1,\ldots,a_{k-1})K_k(a_k,\ldots,a_1) \\ &\quad = K_k(a_1,\ldots,a_k)[K_{k+1}(a_{k-1},a_k,\ldots,a_1) + K_{k-1}(a_1,\ldots,a_{k-1})]. \end{aligned}$$

So the condition for reversibility of x is the following:

$a_1 \ldots a_k$ is reversible and exactly one of $a_1 \ldots a_{k+1}$ and $a_1 \ldots a_{k-1}$ is reversible.

Three cases are to be considered.

(a) $a_1 \ldots a_{k-1} \to c$. Then $a_k = 1$ and $a_{k+1} = 1$. There are $|L_c^{(k-1)}|$ reversible palindromes of this type.

(b) $a_1 \ldots a_{k-1} \to 0$. Then $a_{k+1} = 1$ and a_k can be either 0 or 1. So there are $2|L_0^{(k-1)}|$ reversible palindromes of this type.

(c) $a_1 \ldots a_{k-1} \to 1$, then $a_k = 0$ and $a_{k-1} = 1$. In this case we get $|L_1^{(k-1)}|$ reversible palindromes.

So the total number of reversible palindromes of length n is

$$|L_c^{(k-1)}| + 2|L_0^{(k-1)}| + |L_1^{(k-1)}| = 2^{k-1} + |L_0^{(k-1)}|. \qquad \square$$

Now, we can prove

Theorem 4.1. *For $n \geqslant 3$, using n-cell null boundary 90/150 CA it is not possible to realize all reversible LFSMs.*

Proof. This is proved by showing that for $n \geqslant 3$, $A_n > B_n$. The above lemma gives the expression for $|D^{(n)}|$ in terms of $|L_0^{(n)}|$. Now, $|L_0^{(n)}| = 2^n - |S^{(n)}|$ and the value for $|S^{(n)}|$ is already known from Theorem 2.3. Since A_n and B_n is expressed in terms of $|D^{(n)}|$, it is easy to find the expressions for A_n and B_n and check that indeed $A_n > B_n$. □

Acknowledgements

The authors are grateful to the referees for their careful reading and critical comments.

References

[1] E.R. Berlekamp, Algebraic Coding Theory, McGraw-Hill, New York, 1968.
[2] R.L. Graham, D.E. Knuth, O. Patashnik, Concrete Mathematics, A Foundation for Computer Science, Addison-Wesley, Reading, MA, 1988.
[3] S.W. Golomb, Shift Register Sequences, Aegean Park Press, Laguna Hills, CA, 1982.
[4] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, Reading, MA, 1979.
[5] P.D. Hortensius et al., Parallel pseudo random number generation for VLSI systems using cellular automata, IEEE Trans. Comput. 38 (10) (1989) 1466–1473.
[6] O. Martin, A.M. Odlyzko, S. Wolfram, Algebraic properties of cellular automata, Commun. Math. Phys. 93 (1984) 219–258.
[7] J.L. Massey, Shift register synthesis and BCH decoding, IEEE Trans. Inf. Theory IT-15 (1969) 122–127.
[8] S. Nandi, P.P. Chaudhuri, Analysis of periodic and intermediate boundary 90/150 cellular automata, IEEE Trans. Comput. 45 (1) (1996) 1–11.
[9] M. Serra et al., The analysis of one dimensional cellular automata and their aliasing properties, IEEE Trans. Comput. Aided Design Circuits Systems 9 (7) (1990) 767–778.
[10] K. Sutner, Additive automata on graphs, Complex Systems 2 (1988) 649–661.
[11] S. Wolfram, Statistical mechanics of cellular automata, Rev. Mod. Phy. 55 (1983) 601–644.