Replication of a Binary Image on a One-Dimensional Cellular Automaton with Linear Rules

U Srinivasa Rao[*]
Jeganathan L[†]
School of Computing Science and Engineering
Vellore Institute of Technology
Vandalur-Kelambakkam Road, Chennai, India-600 127
[*]*umitty.srinivasarao@vit.ac.in*
[†]*jeganathan.l@vit.ac.in*

A two-state, one-dimensional cellular automaton (1D CA) with uniform linear rules on an $(r+1)$-neighborhood replicates any arbitrary binary image given as an initial configuration. By these linear rules, any cell gets updated by an EX-OR operation of the states of extreme (first and last) cells of its $(r+1)$-neighborhood. These linear rules replicate the binary image in two ways on the 1D CA: one is without changing the position of the original binary image at time step $t = 0$ and the other is by changing the position of the original binary image at time step $t = 0$. Based on the two ways of replication, we have classified the linear rules into three types. In this paper, we have proven that the binary image of size m gets replicated exactly at time step 2^k of the uniform linear rules on the $(r+1)$-neighborhood 1D CA, where k is the least positive integer satisfying the inequality $m \big/ r \leq 2^k$. We have also proved that there are exactly $\left(r * 2^k - m\right)$ cells between the last cell of the binary image and the first cell of the replicated binary image (or the first cell of the binary image and the last cell of the replicated image).

Keywords: replication; binomial coefficients; linear rules; binary image; rule 90

1. Introduction

A one-dimensional cellular automaton (1D CA) consists of an array of cells: each cell can be in one of a finite number of states and the state of each cell is updated on every time step, according to a deterministic rule based on the state of the neighboring cells and its own state. Properties of 1D CAs have been analyzed in great detail by varying the cell neighborhood, transition rules, boundary conditions and so forth.

Replication (generation of exactly two perfect non-overlapping copies of the given binary image) is a very useful application in image processing, textile design, DNA genetics research, statistical physics, artificial life and other fields. In [1] Wolfram discussed self-similar patterns like Sierpiński's triangle on 1D CAs; Willson [2–4] studied the generation of fractals and fractal dimension with linear rules; Culik and Dube [5] have proved that linear rules will always generate a highly regular behavior on any initial configuration; Fredkin [6] discussed the fractal replicator (calling it Fredkin's replicator) on the two-dimensional cellular automaton (2D CA); Mitra and Kumar [7] have discussed fractal replication on 1D CAs with look-ahead (cell's own future state)—the replication can happen at time step 2^k, where k is a non-negative integer. In [5, 7] and Gravner and Griffeath [8], it was independently observed that replication occurs at time step $2^{\lceil(\log_2(m-1))\rceil}$ ($m > 1$ is the size of the initial configuration).

Assuming that the uniform linear rules are repeatedly applied on the binary image on the $(r+1)$-neighborhood 1D CA, we have addressed the following questions with mathematical rigor in Section 4, related to the replication of the binary image in the 1D CA, which have not been addressed by any researcher so far.

1. Why does the replication occur exactly after time step 2^k? Is it possible to compute the exact value of k?
2. Does k depend on the size of the initial configuration alone or depend on the size of the initial configuration and the size of the neighborhood r?
3. Is it possible to compute the space (in terms of the number of cells) that gets generated between the initial configuration and its replicated image?
4. Is it possible to regulate the replication process in the sense that the replicated image is positioned in a place of our choice?

2. Preliminaries

In this section, we discuss some elementary definitions pertaining to cellular automata and notations, which are required in this paper.

Definition 1. A rule of a CA is defined as a function (local transition) that determines the new state of each cell in terms of the current state of the cell and the states of its neighborhood cells.

For example, $q_i^{(t+1)} = f\left(q_i^{(t)}, q_{i+1}^{(t)}, \ldots, q_{i+r}^{(t)}\right)$, where i is the index of an individual cell in the 1D CA array, t is the time step, $q_i^{(t+1)}$ is the state of cell i at time step $(t+1)$, and f is the local transition function.

Definition 2. Let a_p and a_q be two binary digits. Then EX-OR ($\oplus$) is defined as

$$a_p \oplus a_q = \begin{cases} 0, & \text{if } a_p = a_q. \\ 1, & \text{if } a_p \neq a_q. \end{cases}$$

Remark 1. The EX-OR operator satisfies commutative and associative properties.

Remark 2. t_{C_x} is the number of ways to choose x unordered items from a set of t items; that is,

$$t_{C_x} = t! \big/ (t-x)!x!$$

and

$$t_{C_x} = (t-1)_{C_{x-1}} + (t-1)_{C_x}.$$

Remark 3. Throughout this paper,

$$na_p = \bigoplus_n a_p = \underbrace{a_p \oplus a_p \oplus \cdots \oplus a_p}_{\oplus^n \text{ times}}.$$

Remark 4. $\oplus_n a_p = 0$ if and only if n is even and $\oplus_n a_p = a_p$ if and only if n is odd.

Definition 3. A rule is said to be a linear rule if f involves EX-OR operation only. The algebraic normal form [9] of the linear rule is

$$f(q_1, q_2, q_3, \ldots, q_n) = \alpha_1 q_1 \oplus \alpha_2 q_2 \oplus \cdots \oplus \alpha_n q_n,$$

where $\alpha_i \in \{0, 1\}$, $\forall\, i \in \{1, 2, \ldots, n\}$.

Definition 4. A rule is said to be a uniform rule if all cells are updated by the same rule.

Definition 5. Replication is defined as the process of generating exactly two perfect non-overlapping copies of the initial configuration. Here initial configuration is the binary image.

3. Classification of Linear Rules Based on the Nature of Replication

In [10] Wolfram classified the rules of three-neighborhood 1D CAs into four categories based on their statistical properties. In this paper, we classify all linear rules based on the nature of replication. The linear rule 90 [5, 7, 8] replicates the given binary image at step 2^k, where k depends on the size of the binary image.

3.1 Rule 90 and Its Modifications

In rule 90 [11], the local transition function depends on the left neighbor and right neighbor of the central cell on the three-neighborhood 1D CA. The central cell gets updated as an EX-OR operation of state of the left neighbor and right neighbor; that is,

$$q_i^{(t+1)} = f\left(q_{i-1}^{(t)}, q_i^{(t)}, q_{i+1}^{(t)}\right) = q_{i-1}^{(t)} \oplus q_{i+1}^{(t)}.$$

The first row of Table 1 describes all the possibilities of a three-neighborhood CA; the updated state of the central cell by rule 90 in the respective neighborhood is given in the second row of Table 1.

Neighborhood state	1 1 1	1 1 0	1 0 1	1 0 0	0 1 1	0 1 0	0 0 1	0 0 0
Update state of the central cell	0	1	0	1	1	0	1	0

Table 1. Rule 90.

Let $B = a_1 a_2 a_3 a_4 a_5$ be an initial configuration of size five on the three-neighborhood 1D CA array. Assume that all the cells that do not involve the image are in zero state at time step $t = 0$. Rule 90 updates each binary bit of image B as shown in Table 2. From Table 2, we have observed that at each time step t, the coefficient of any image cell (a_i) resembles row t of the Pascal triangle. The sequence of coefficients of a_1 (wherever a_1 occurs) at time step $t = 3$ is 1, 3, 3, 1, which resembles the third row of the Pascal triangle. The same is true for any a_i at any time t, since $n a_p = 0$, if n is even; otherwise $n a_p = \oplus_n a_p = a_p$. The rule replicates the binary image after $t = 4 = 2^2$ time steps. The space between the binary image and its replicated image has three cells, which are in the zero state.

In Table 2, at time step $t = 4$, we observe that the binary bits of image B have changed their position on the 1D CA array. The initial image B occupied the cells from index i to $i + 4$. After replication, image B and its replicated image occupy the positions from $i - 4$ to i and from $i + 4$ to $i + 8$, respectively. That is, replication by rule 90 repositions image B. Then we raise the question, Is it possible to replicate an image without changing its position in 1D CA array?

In rule 90, the neighborhood set of a cell includes that cell, its left neighbor and its right neighbor; that is, the neighborhood set of q_i will be $\{q_{i-1}, q_i, q_{i+1}\}$. Instead, we can modify the neighborhood set of q_i as $\{q_i, q_{i+1}, q_{i+2}\}$. That is, the neighborhood set of a cell includes that cell and the two successive cells to the right. If we apply rule 90 to this neighborhood at the $t+1$ time step,

$$q_i^{(t+1)} = f\left(q_i^{(t)}, q_{i+1}^{(t)}, q_{i+2}^{t}\right) = q_i^{(t)} \oplus q_{i+2}^{(t)}.$$

We refer to this modified rule as *R rule 90* (Table 3). Here the prefix R indicates the fact that all the neighbors are to the right of the cell that gets updated.

Index / Steps	*i*-4	*i*-3	*i*-2	*i*-1	*i*	*i*+1	*i*+2	*i*+3	*i*+4	*i*+5	*i*+6	*i*+7	*i*+8
t=0	0	0	0	0	a_1	a_2	a_3	a_4	a_5	0	0	0	0
t=1	0	0	0	a_1	a_2	$a_1 \oplus a_3$	$a_2 \oplus a_4$	$a_3 \oplus a_5$	a_4	a_5	0	0	0
t=2	0	0	a_1	a_2	$2a_1 \oplus a_3$	$2a_2 \oplus a_4$	$a_1 \oplus 2a_3 \oplus a_5$	$a_2 \oplus 2a_4$	$a_3 \oplus 2a_5$	a_4	a_5	0	0
t=3	0	a_1	a_2	$3a_1 \oplus a_3$	$3a_2 \oplus a_4$	$3a_1 \oplus 3a_3 \oplus a_5$	$3a_2 \oplus 3a_4$	$a_1 \oplus 3a_3 \oplus 3a_5$	$a_2 \oplus 3a_4$	$a_3 \oplus 3a_5$	a_4	a_5	0
t=4	a_1	a_2	$4a_1 \oplus a_3 = a_3$	$4a_2 \oplus a_4 = a_4$	$6a_1 \oplus 4a_3 \oplus a_5 = a_5$	$6a_2 \oplus 4a_4 = 0$	$4a_1 \oplus 6a_3 \oplus 4a_5 = 0$	$4a_2 \oplus 6a_4 = 0$	$a_1 \oplus 4a_3 \oplus 6a_5 = a_1$	$a_2 \oplus 4a_4 = a_2$	$a_3 \oplus 4a_5 = a_3$	a_4	a_5

Table 2. Image *B* and its replicated image after four steps with rule 90.

Neighborhood state	1 1 1	1 1 0	1 0 1	1 0 0	0 1 1	0 1 0	0 0 1	0 0 0
Update state of the leftmost cell	0	1	0	1	1	0	1	0

Table 3. R rule 90.

Then, we apply R rule 90 to the initial configuration $B = a_1 a_2 a_3 a_4 a_5$. We observe from Table 4 that the position of the initial image does not get changed and the replicated image appears to the left of the image. In other words, if the neighborhood set is oriented to the right, the application of R rule 90 makes a copy to the left of *B*.

Index / Steps	i-8	i-7	i-6	i-5	i-4	i-3	i-2	i-1	i	i+1	i+2	i+3	i+4
t=0	0	0	0	0	0	0	0	0	a_1	a_2	a_3	a_4	a_5
t=1	0	0	0	0	0	0	a_1	a_2	$a_1 \oplus a_3$	$a_2 \oplus a_4$	$a_3 \oplus a_5$	a_4	a_5
t=2	0	0	0	0	a_1	a_2	$2a_1 \oplus a_3$	$2a_2 \oplus a_4$	$a_1 \oplus 2a_3 \oplus a_5$	$a_2 \oplus 2a_4$	$a_3 \oplus 2a_5$	a_4	a_5
t=3	0	0	a_1	a_2	$3a_1 \oplus a_3$	$3a_2 \oplus a_4$	$3a_1 \oplus 3a_3 \oplus a_5$	$3a_2 \oplus 3a_4$	$a_1 \oplus 3a_3 \oplus 3a_5$	$a_2 \oplus 3a_4$	$a_3 \oplus 3a_5$	a_4	a_5
t=4	a_1	a_2	$4a_1 \oplus a_3 = a_3$	$4a_2 \oplus a_4 = a_4$	$6a_1 \oplus 4a_3 \oplus a_5 = a_5$	$6a_2 \oplus 4a_4 = 0$	$4a_1 \oplus 6a_3 \oplus 4a_5 = 0$	$4a_2 \oplus 6a_4 = 0$	$a_1 \oplus 4a_3 \oplus 6a_5 = a_1$	$a_2 \oplus 4a_4 = a_2$	$a_3 \oplus 4a_5 = a_3$	a_4	a_5

Table 4. *B* and its replicated image after four steps with R rule 90.

Instead of having all the neighbors on the right side of the cell, we can have all the neighbors to the left. In that case, we have rule 90 as

$$q_i^{(t+1)} = f\left(q_{i-2}^{(t)}, q_{i-1}^{(t)}, q_i^{(t)}\right) = q_{i-2}^{(t)} \oplus q_i^{(t)}$$

as in Table 5. We refer to this modified rule as *L rule 90*, where the prefix L indicates the fact that all the neighbors are to the left of the cell that gets updated.

Thus, if every cell gets updated as the EX-OR operation of itself and its rightmost neighborhood, the original image is not repositioned and the replicated copy occurs on the left-hand side of the initial image *B*; that is, if the neighborhood moves to the right, the replicated copy occurs to the left of the initial image *B* and vice versa.

Neighborhood state	1 1 1	1 1 0	1 0 1	1 0 0	0 1 1	0 1 0	0 0 1	0 0 0
Update state of the rightmost cell	0	1	0	1	1	0	1	0

Table 5. L rule 90.

Based on the nature of the neighborhood set, we classify the linear rules that update by the EX-OR operation of the extreme (first and last) cells of its neighborhood; that is, we classify rules of the form

$$f(q_i, q_{i+1}, \dots, q_{i+r}) = q_j = q_i \oplus q_{i+r},$$

where i is an integer, $i \le j \le i + r$ and $r = 1, 2, 3, \dots$.

Type-1 rules: $(r+1)$-neighborhood rules such that all the r neighbors lie to the right-hand side of the cell that gets updated and the updating of the cells depends only on the state of itself and the state of the last (rightmost) neighbor; that is,

$$q_i^{(t+1)} = f\left(q_i^{(t)}, q_{i+1}^{(t)}, q_{i+2}^{(t)}, \dots, q_{i+r}^{(t)}\right) = q_i^{(t)} \oplus q_{i+r}^{(t)}$$

where $r = 1, 2, 3, \dots$.

Type-2 rules: $(r+1)$-neighborhood rules such that all the r neighbors lie on the left-hand side of the cell that gets updated and the updating of the cells depends only on the state of itself and the state of the last neighbor; that is,

$$q_i^{(t+1)} = f\left(q_{i-r}^{(t)}, q_{i-r+1}^{(t)}, q_{i-r+2}^{(t)}, \dots, q_i^{(t)}\right) = q_{i-r}^{(t)} \oplus q_i^{(t)}$$

where $r = 1, 2, 3, \dots$.

Type-3 rules: $(r+1)$-neighborhood rules such that r_1 neighbors lie on the left side of the cell and r_2 neighbors lie on the right side of the cell that gets updated. Here $r_1 + r_2 = r$, and r_1 and r_2 are positive integers. Every cell gets updated by the rule

$$q_i^{(t+1)} = \\ f\left(q_{i-r_1}^{(t)}, q_{i-r_1+1}^{(t)}, \dots, q_i^{(t)}, \dots, q_{i+r_2-1}, q_{i+r_2}^{(t)}\right) = q_{i-r_1}^{(t)} \oplus q_{i+r_2}^{(t)}$$

on the $(r+1)$-neighborhood, where $r_1 + r_2 = r$ and $r_1 \ge 1$, $r_2 \ge 1$.

4. Computation for the Exact Time Steps of Replication

In this section, we compute the exact number of time steps for replication with the Type-1 rules on the $(r+1)$-neighborhood 1D CA. In [7, 5, 12] it was observed that the linear rules render multiple copies of any binary image on the 1D CA, when the linear rule is applied n times, where $n = 2^k$ and k is dependent on the size of the initial configuration. But [7] and [5] do not contain any proof or justification of their observation and do not include any attempt to compute the exact value of k.

We have proved that the number of time steps required to replicate the initial configuration of the binary image does not depend on the size of the binary image alone, but also depends on r of the $(r+1)$-neighborhood. If we increase the neighborhood r, then the number of steps will be decreased to replicate the initial configuration of the binary image. We have proved the queries for the Type-1 rules in this section, which are discussed in Section 1. Analogously, we can prove our results for Type 2 rules and Type 3 rules also.

In [13] Fine proved a result on the divisibility of binomial coefficients, which is stated as Theorem 1 without proof. We need Theorem 1 to prove our results.

Theorem 1. The binomial coefficient t_{C_x}, where $0 < x < t$, is divisible by a prime p if and only if t is a power of p.

Proof of Theorem 1 can be seen in [13], so we omit the details.

The Type-1 rule replicates the initial configuration of the binary image within one step on the $(r+1)$-neighborhood 1D CA, if the total number of neighboring cells r is greater than or equal to the size of the initial configuration. The space between the replicated image and the initial configuration is the difference between the total number of neighboring cells and the size of the initial configuration.

Theorem 2. Let f be a Type-1 rule on an $(r+1)$-neighborhood. Assume B is an initial configuration of the binary image that starts and ends with 1. The length of B is m, and all cell states that do not involve B are in state zero. If $r \geq m$, then rule f replicates the image on the left-hand side of B within one step, with $r-m$ cells in zero state between B and its replicated image.

Proof. Let $B = a_1 a_2 \ldots a_m$ be an initial configuration of the binary image that starts and ends with 1 on the 1D CA. All cell states that do not involve B are in state zero. Assume image B is placed on the 1D CA array as follows: a_1 is placed at index i, a_2 is placed at index $i+1$, and so on. The last bit a_m is placed at index $i+m-1$ on the array (see Figure 1).

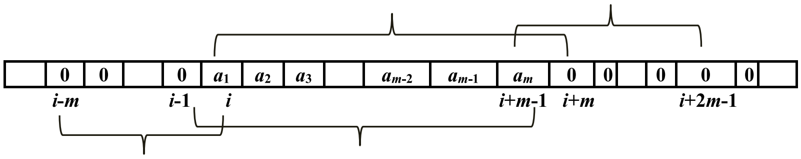

Figure 1. Image B on the $(r+1)$-neighborhood 1D CA.

If $r \geq m$, then $\exists$ a non-negative integer $b \ni r = m + b$. Since f is a Type-1 rule on an $(r+1)$-neighborhood, every state of index cell i gets updated as a result of the EX-OR operation between the state of index cell i and the state of index cell $(i+m+b)$ on the 1D CA array; that is,

$$q_i^{(t+1)} = f\left(q_i^{(t)}, q_{i+1}^{(t)}, q_{i+2}^{(t)}, \ldots, q_{i+r}^{(t)}\right) = q_i^{(t)} \oplus q_{i+r}^{(t)} = q_i^{(t)} \oplus q_{i+m+b}^{(t)}.$$

At time step $t = 1$, rule f updates the initial configuration as follows:

- Index cell $(i-m-b)$ gets updated to state $q_{i-m-b}^{(0)} \oplus q_i^{(0)} = 0 \oplus a_1 = a_1$.
- Index cell $(i-m-b+1)$ gets updated to state $q_{i-m-b+1}^{(0)} \oplus q_{i+1}^{(0)} = 0 \oplus a_2 = a2$, and so forth.
- Index cell $(i-b-1)$ gets updated to state $q_{i-b-1}^{(0)} \oplus q_{i+m-1}^{(0)} = 0 \oplus a_m = a_m$.

Since rule f is on an $(m+b+1)$-neighborhood, the zero state of cell $(i-b)$ will interact with the zero state of cell $(i+m)$ with the EX-OR operation.

Thus, index cell $(i-b)$ gets updated to state $q_{i-b}^{(0)} \oplus q_{i+m}^{(0)} = 0 \oplus 0 = 0$. A total of b number of cells from index $(i-b)$ to index $(i+1)$ will get updated as $0 \oplus 0 = 0$.

The space between the original image and its replicated image is $b = r - m$. Index cell i gets updated to state

$$q_i^{(0)} \oplus q_{i+m+b}^{(0)} = a_1 \oplus 0 = a_1,$$

index cell $(i+1)$ gets updated to state

$$q_{i+1}^{(0)} \oplus q_{i+m+b+1}^{(0)} = a_2 \oplus 0 = a_2,$$

and so forth; index cell $(i+m-1)$ gets updated to state

$$q_{i+m-1}^{(0)} \oplus q_{i+2m+b-1}^{(0)} = a_m \oplus 0 = a_m.$$

As an illustration, only cells of the boundary interaction of image B for $r = m$ are shown in Figure 1. This completes the proof. □

A Type-1 rule on an $(r+1)$-neighborhood updates the single-bit initial configuration of the image as the rows of the Pascal triangle. After time step t, the image is updated as row t of the Pascal triangle.

Theorem 3. Let f be a Type-1 rule on an $(r+1)$-neighborhood. Let $B = a_1 = 1$ be an initial configuration on the 1D CA array, where a_1 is presented at index i on the array. All the states of the cell that do not involve the initial configuration are in zero states. At step t, the Type-1 rule updates image B in such a way that

Result (A):

- Index cell$(i - tr)$ has state $t_{C_0}a_1$; cell indices from $(i - tr + 1)$ to $(i - (t - 1)r - 1)$ have state zero.
- Index cell $(i - (t - 1)r)$ has state $t_{C_1}a_1$; cell indices from $(i - (t - 1)r + 1)$ to $(i - (t - 2)r - 1)$ have state zero.
- Index cell $(i - r)$ has state $t_{C_{t-1}}a_1$; cell indices from $(i - r + 1)$ to $(i - 1)$ have state zero.
- Index cell i has state $t_{C_t}a_1$.

That is, the updated image is shown in Figure 2.

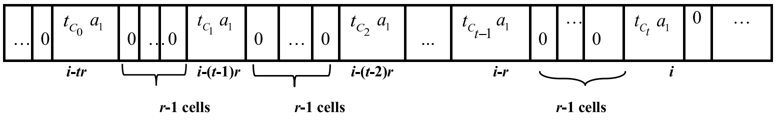

Figure 2. Image B after step t on the 1D CA.

Proof. We prove result (A) by induction on the time steps t. On an $(r + 1)$-neighborhood, the Type-1 rule will be like

$$q_i^{(t+1)} = f\left(q_i^{(t)}, q_{i+1}^{(t)}, q_{i+2}^{(t)}, \dots, q_{i+r}^{(t)}\right) = q_i^{(t)} \oplus q_{i+r}^{(t)}$$

with $r \geq 1$. At time step $t = 0$, a 1D CA array will have a_1 at index cell i, and all other cell states are zero.

Base Case: As is known, rule f updates a cell at index i with $q_i^{(0)} \oplus q_{i+r}^{(0)}$; that is, index cell $(i - r)$ will be updated as $0 \oplus a_1 = a_1$, index cell i will be updated as $a_1 \oplus 0$, and all other cells will be updated as $0 \oplus 0$. Therefore, index cell $(i - r)$ will have state $t_{C_0}a_1$, index cell i will have state $t_{C_1}a_1$, and all other cells will have state zero. Result (A) is true for $t = 1$. After the first time step, the image is shown in Figure 3.

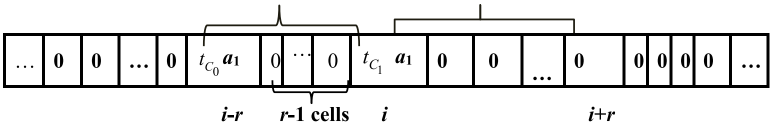

Figure 3. Image B after the first step on the 1D CA.

Induction Step: We assume that result (A) is true for step $(t - 1)$. That is, rule f updates the 1D CA array at step $(t - 1)$ as in Figure 4.

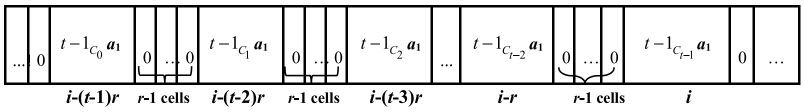

Figure 4. Image B after step $(t-1)$ on the 1D CA.

When we apply rule f to the step $(t-1)$ configuration, index cell $(i-tr)$ gets updated to state

$$q_{i-tr}^{(t-1)} \oplus q_{i-(t-1)r}^{(t-1)} = 0 \oplus (t-1)_{C_0} a_1 = a_1 = t_{C_0} a_1,$$

all the cells from index $(i-tr+1)$ to index cell $(i-(t-1)r-1)$ will get updated as $0 \oplus 0 = 0$, index cell $(i-(t-1)r)$ gets updated to state

$$q_{i-(t-1)r}^{(t-1)} \oplus q_{i-(t-2)r}^{(t-1)} = (t-1)_{C_0} a_1 \oplus (t-1)_{C_1} a_1 = t_{C_1} a_1, \ldots,$$

and index cell i gets updated to state

$$q_i^{(t-1)} \oplus q_{i+r}^{(t-1)} = (t-1)_{C_{(t-1)}} a_1 \oplus 0 = a_1 \oplus 0 = a_1 = t_{C_t} a_1.$$

At time step t, we have the configuration as in Figure 2; that is, result (A) is true for time step t. By induction on the time steps, result (A) is true for all $t \geq 1$. This completes the proof. □

Remark 5. The Type-1 rule updates the single-bit initial configuration a_1 on an $(r+1)$-neighborhood 1D CA at step t as a string

$$\text{"} \underset{t_{C_0}}{\oplus} a_1 0^{r-1} \underset{t_{C_1}}{\oplus} a_1 0^{r-1} \underset{t_{C_2}}{\oplus} a_1 0^{r-1} \cdots \underset{t_{C_{t-1}}}{\oplus} a_1 0^{r-1} \underset{t_{C_t}}{\oplus} a_1 \text{"},$$

where the binary bit $\oplus_{t_{C_x}} a_1 = 0$ if and only if t_{C_x} is even, and the binary bit $\oplus_{t_{C_x}} a_1 = a_1$ if and only if t_{C_x} is odd.

Remark 6. The total number of cells in Figure 2 from the cell state of $t_{C_0} a_1$ to the cell state of $t_{C_t} a_1$ is $1 + t * r$.

Unless otherwise stated, we assume the following: let $B = a_1 a_2 \ldots a_m$ be a binary image of length m on the 1D CA that starts and ends with 1, where $a_1 = 1$ is positioned at index i on the 1D CA, a_2 is positioned at index $(i+1)$ on the 1D CA and so on; $a_m = 1$ is positioned at index $(i+m-1)$. All cell states that do not involve the initial configuration of the binary image B are in zero.

Corollary 1. Let f be a Type-1 rule on an $(r+1)$-neighborhood. At time step t, rule f with $1 \leq r < m$ updates each cell content of the image as row t of the Pascal triangle, and the update image occupies $m + t * r$

cells on the 1D CA array. The update image after time step t is shown in Figure 5.

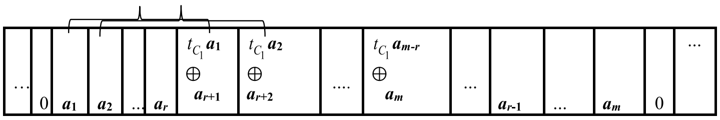
0
a_1
a_2
a_r
$t_{C_1} a_1 \oplus a_{r+1}$
$t_{C_1} a_2 \oplus a_{r+2}$
$t_{C_1} a_{m-r} \oplus a_m$
a_{r-1}
a_m
0

Figure 5. Image B after step t on $m + t * r$ cells of 1D CA.

Proof. Let $B = a_1 a_2 \ldots a_m$ be an initial configuration of the binary image on the 1D CA array, and the states of the other cells are zero.

Apply Theorem 3 to each bit of image B individually; cell a_1 is updated as in Figure 2, cell a_2 is updated as in Figure 6, and cell a_m is updated as in Figure 7.

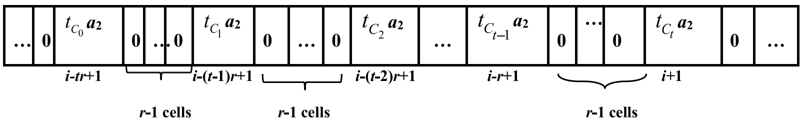

Figure 6. Image cell a_2 of B after step t on the 1D CA.

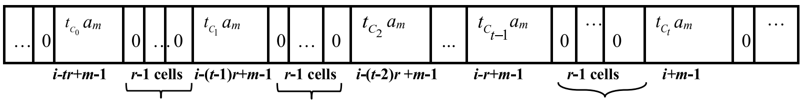

Figure 7. Image cell a_m of B after step t on the 1D CA.

All cells update their states in parallel by interacting with defined neighbors in the rule. Therefore, after t steps, the image updates on $m + t * r$ cells as shown in Figure 5. This completes the proof. □

Remark 7. The Type-1 rule updates every cell based on the r-neighboring cells that lie on the right side of that cell. By the Type-1 rule, cells on the left side of the image interact with cells of image B or zero-state cells. The cell states of image B interact with state cells of the image or zero-state cells on the right. But right-hand side cells of image B interact only with zero-state cells. Since all the neighbors lie to the right of B and the cells to the right of B do not interact with B, the original image is not repositioned due to the EX-OR operation and the replicated image occurs on the left-hand side of B.

In the proof of Theorem 2, we have seen that the Type-1 rules on the $(r + 1)$-neighborhood 1D CA replicate the initial configuration of the image within one step if $r \geq m$, where m is the size of the initial configuration. In other words, if $r \geq m$, then the rule takes $\lceil m / r \rceil$ steps to replicate the initial configuration of the binary image.

In Theorem 4, we try to define the relationship among the size of the initial configuration, cell neighborhood r in rule f and number of steps to replicate the initial configuration, and also discuss space between the first cell of the binary image and the last cell of its replicating image.

Theorem 4. Let f be a Type-1 rule on an $(r+1)$-neighborhood. If $1 \leq r < m$, then $\exists$ a least positive integer k such that $m / r \leq 2^k$, the f rule replicates image B at step 2^k, and the space between the original image and its replicated image is $r * 2^k - m$ cells of zero state.

Proof. By Corollary 1, each bit of the binary image $B = a_1 a_2 \ldots a_m$ gets updated with binomial coefficients resembling row t of the Pascal triangle, and the updated image occupies $m + t * r$ cells on the 1D CA array, as shown in Figure 5.

Now we have to evaluate the state of the cell. By Theorem 1, all binomial coefficients $t_{C_1}, t_{C_2}, \ldots, t_{C_{t-1}}$ are divisible by 2 if and only if t is a power of 2. Thus, $t_{C_1}, t_{C_2}, \ldots, t_{C_{t-1}}$ are even numbers when $t = 2^k$. If $t_{C_1}, t_{C_2}, \ldots, t_{C_{t-1}}$ are even numbers, then we have $t_{C_1} a_p = 0$, $t_{C_2} a_p = 0, \ldots, t_{C_{t-1}} a_p = 0$, but $t_{C_0} a_p = a_p$ and $t_{C_t} a_p = a_p$. This is true for all bits in image B. Therefore, the original image is replicated on the 1D CA array.

Assume the original image replicates at step t; at this step the image updates on $m + t * r$ cells. The original image and its replicated image occupy $2m$ cells out of $m + t * r$ cells on the 1D CA array. This implies that

$$2m \leq m + t * r \Rightarrow m \leq t * r \Rightarrow \frac{m}{r} \leq t.$$

Therefore, $\exists$ a least positive integer k such that $m / r \leq t = 2^k$, rule f replicates the image at step 2^k, and image B updates on $m + 2^k * r$ cells on the 1D CA array. Image B and its replicated image occupy $2m$ cells out of $m + 2^k * r$, $m + 2^k * r - 2m = r * 2^k - m$ cells with zero state between them. Therefore, the space between B and its replicated image is $r * 2^k - m$ cells of zero state. This completes the proof. □

Since we have obtained the relationship among the size of the initial configuration (m), cell neighborhood (r) and k (2^k is the exact time step for replication), we can easily compute the exact time step at which an image of specified size gets replicated on an

$(r+1)$-neighborhood 1D CA. In short, given any two parameters among k, r, m, we can easily compute the other.

Corollary 2. Type-1 rule f replicates image B exactly at time step 2^k on the $(r+1)$-neighborhood 1D CA if and only if $\exists$ a least positive integer r such that $r \geq m / 2^k$.

Proof. ($\Longrightarrow$) Type-1 rule f replicates image B at $t = 2^k$ time steps on the $(r+1)$-neighborhood 1D CA. By Corollary 1, the image updates on $m + 2^k * r$ cells after time step $t = 2^k$. The original image and its replicated image occupy $2m$ cells out of $m + 2^k * r$ cells. Therefore,

$$2m \leq m + 2^k * r \Rightarrow m \leq 2^k * r \Rightarrow r \geq \frac{m}{2^k}.$$

We have to prove that r is the least positive integer such that $r \geq m / 2^k$.

We prove this statement by proof by contradiction. Suppose r is the least positive integer such that $r < m / 2^k$. This implies that

$$r < \frac{m}{2^k} \Rightarrow m + 2^k * r < 2m.$$

This is a contradiction to the hypothesis, because the original image and the replicated image occupy at least $2m$ cells on the 1D CA. Therefore, $\exists$ a least positive integer r such that $r \geq m / 2^k$.

($\Longleftarrow$) Since r is the least positive integer such that $r \geq m / 2^k$, this implies that $m / r \leq 2^k$. We have to prove Type-1 rule f replicates image B exactly at time step $t = 2^k$ on the $(r+1)$-neighborhood 1D CA. We prove this statement by proof by contradiction.

Suppose that Type-1 rule f does not replicate image B exactly at $t = 2^k$ time step on the $(r+1)$-neighborhood 1D CA.

By Corollary 1, the image updates on $m + 2^k * r$ cells after $t = 2^k$ time steps and $m + 2^k * r < 2m$.

Since r is the least positive integer such that $r \geq m / 2^k$, this implies that $m + 2^k * r \geq m + m$, $2m \leq m + 2^k * r < 2m \Rightarrow 2m < 2m$. This is not possible; our supposition is false. Therefore, Type-1 rule f replicates image B exactly at time step 2^k on the $(r+1)$-neighborhood 1D CA. This completes the proof. □

An illustration of Corollary 2: In Table 4, the size of image B is five; assume that the Type-1 rule f replicates image B exactly at time step $2^2 = 4$ on the $(r+1)$-neighborhood 1D CA. By the result of

Corollary 2, the value of r is the least positive integer satisfying the inequality

$$r \geq \frac{m}{2^k} \Rightarrow r \geq \frac{5}{2^2}.$$

Therefore, the value of r is 2; that is, Type-1 rule f replicates image B exactly at time step 4 on the three-neighborhood 1D CA. Similarly, let $r = 2$ be the least positive integer satisfying the inequality $r \geq m \big/ 2^k$. After simplifying the inequality with the values of m and r, the value of k is 2.

5. Conclusions and Future Work

With the help of Type-1 rules and the result on the divisibility of binomial coefficients [13], we have proved that an image of size m gets replicated exactly at step 2^k on the $(r+1)$-neighborhood one-dimensional cellular automaton (1D CA), where k is the least positive integer satisfying the relation $m \big/ r \leq t = 2^k$.

If we know any two parameters among k, r, m, then we can easily compute the other one. Apart from the computation of the exact time step for replication, this paper will help us in choosing an appropriate rule if the user wishes to have the replicated image to the right (or left) of the original image.

We have also computed the space with zero states between the original binary image and the replicated image. In this process, we have also identified the position where the replicated image occurs. This will help us to identify the appropriate rule for the replication of an image at a particular position. Thus, all the queries raised in Section 1 are addressed.

Similarly, we can use the trinomial coefficients [14] to compute the exact number of time steps required for the replication of exactly three non-overlapping copies of the original binary image and extend the results to generate multiple copies of the binary image. As a future extension, we can extend this work to explore the analogous scenario in the higher-dimensional cellular automata.

References

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

[2] S. J. Willson, "Cellular Automata Can Generate Fractals," *Discrete Applied Mathematics*, **8**(1), 1984 pp. 91–99. doi:10.1016/0166-218X(84)90082-9.

[3] S. J. Willson, "Computing Fractal Dimensions for Additive Cellular Automata," *Physica D: Nonlinear Phenomena*, **24**(1–3), 1987 pp. 190–206. doi:10.1016/0167-2789(87)90074-1.

[4] S. J. Willson, "Growth Rates and Fractional Dimensions in Cellular Automata," *Physica D: Nonlinear Phenomena*, **10**(1–2), 1984 pp. 69–74. doi:10.1016/0167-2789(84)90250-1.

[5] K. Culik II and S. Dube, "Fractal and Recurrent Behavior of Cellular Automata," *Complex Systems*, **3**(3), 1989 pp. 253–267. complex-systems.com/pdf/03-3-3.pdf.

[6] E. Fredkin, "An Informational Process Based on Reversible Universal Cellular Automata," *Physica D: Nonlinear Phenomena*, **45**(1–3), 1990 pp. 254–270. doi:10.1016/0167-2789(90)90186-S.

[7] S. Mitra and S. Kumar, "Fractal Replication in Time-Manipulated One-Dimensional Cellular Automata," *Complex Systems*, **16**(3), 2006 pp. 191–197. complex-systems.com/pdf/16-3-1.pdf.

[8] J. Gravner and D. Griffeath, "The One-Dimensional *Exactly 1* Cellular Automaton: Replication, Periodicity, and Chaos from Finite Seeds," *Journal of Statistical Physics*, **142**(1), 2011 pp. 168–200. doi:10.1007/s10955-010-0103-9.

[9] P. P. Choudhury, S. Sahoo, M. Chakraborty, S. Kumar Bhandari and A. Pal, "Investigation of the Global Dynamics of Cellular Automata Using Boolean Derivatives," *Computers and Mathematics with Applications*, **57**(8), 2009 pp. 1337–1351. doi:10.1016/j.camwa.2008.11.012.

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

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

[12] S. Uguz, U. Sahin, H. Akin and I. Siap, "Self-Replicating Patterns in 2D Linear Cellular Automata," *International Journal of Bifurcation and Chaos*, **24**(01), 2014 1430002. doi:10.1142/S021812741430002X.

[13] N. J. Fine, "Binomial Coefficients Modulo a Prime," *The American Mathematical Monthly*, **54**(10), 1947 pp. 589–592. www.jstor.org/stable/2304500?origin=JSTOR-pdf.

[14] E. W. Weisstein. "Trinomial Coefficient" from Wolfram MathWorld—A Wolfram Web Resource. mathworld.wolfram.com/TrinomialCoefficient.html.