The Reversibility of Cellular Automata on Trees with Loops

A. Martín del Rey[1(✉)], E. Frutos Bernal[2], D. Hernández Serrano[3], and R. Casado Vara[4]

[1] Department of Applied Mathematics, Institute of Fundamental Physics and Mathematics, University of Salamanca, 37008 Salamanca, Spain
delrey@usal.es

[2] Department of Statistics, University of Salamanca, 37007 Salamanca, Spain
efb@usal.es

[3] Department of Mathematics, Institute of Fundamental Physics and Mathematics, University of Salamanca, 37008 Salamanca, Spain
dani@usal.es

[4] BISITE Research Group, University of Salamanca, 37007 Salamanca, Spain
rober@usal.es

Abstract. In this work the notion of linear cellular automata on trees with loops is introduced and the reversibility problem in some particular cases is tackled. The explicit expressions of the inverse cellular automata are computed.

Keywords: Cellular automata on graphs · Reversibility · Trees with loops · Evolutionary computation

1 Introduction

A cellular automaton can be considered as a finite state machine formed by a finite number of identical memory units (called cells) which are endowed with a state at every step of time. The state of each cell is updated according to a local transition rule whose variables are the states of its neighbor cells. Cellular automata are simple models of computation capable to simulate complex behaviors; consequently, several applications to all fields of Science and Technology can be found in the scientific literature [6,9].

Of special interest are those cellular automata whose state set is $\mathcal{F}_2 = \{0, 1\}$ (boolean cellular automata). The vector whose coordinates stand for the states of the ordered set of cells at time t is called configuration of the cellular automata at t: $C^t \in \mathbb{F}_2^n$, where n is the total number of cells. When the dynamics of the cellular automaton is governed by means of deterministic transition rules, every configuration will have an unique successor in time. However, every configuration may have several distinct predecessors and, consequently, the time evolution mapping of the cellular automaton is not invertible in general. When there is only one predecessor, the cellular automaton is called reversible; that is, the global

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2021
Y. Dong et al. (Eds.): DCAI 2020, AISC 1237, pp. 241–250, 2021.
https://doi.org/10.1007/978-3-030-53036-5_26

transition function $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$, that yields the configuration at the next-step during the evolution of the cellular automata, is injective [8]. The reversibility is an interesting and desirable property in some applications of cellular automata in Cryptography [13] and Computer Science [7].

The reversibility problem for cellular automata consists of both determining when a certain cellular automaton is reversible, and computing the inverse cellular automaton -if it is possible-. This is not a new problem [5] and it has been tackled for different classes of cellular automata: elementary cellular automata [10], linear cellular automata (see, for example, [3]), two-dimensional cellular automata with hexagonal cellular space [1], memory cellular automata [11], multidimensional finite cellular automata [2,4], etc.

The main goal of this work is to study the reversibility problem of a particular and interesting type of cellular automata on graphs: the linear cellular automata defined on full trees with loops. Specifically, we will show that some cellular automata of this type are reversible, and the corresponding inverse cellular automata will be explicitly computed.

The rest of the paper is organized as follows: the basic definitions and results concerning linear cellular automata on trees with loops are introduced in Sect. 2; in Sect. 3 the reversibility problem for this type of cellular automata is tackled. Finally, the conclusions and further work are shown in Sect. 4.

2 Linear Cellular Automata on Trees with Loops

2.1 General Considerations

Let $G = (V, E)$ be an undirected multigraph such that $V = \{v_1, v_2, \ldots, v_n\}$. A *boolean cellular automaton on* G (CA for short) is a 4-uplet $\mathcal{A}_G = (\mathcal{C}, \mathbb{F}_2, \mathcal{N}, \mathcal{F})$ where:

(1) The set $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ is the cellular space of the CA such that the i-th cell c_i stands for the node v_i of G, where $1 \leq i \leq n = |V|$. For the sake of simplicity will be denote the i-th cell by i in place of c_i.
(2) The Galois field $\mathbb{F}_2 = \{0, 1\}$ is the finite set of states that can be assumed by each cell/node at each step of time. In this sense, the state of the i-th cell at time step t is denoted by $s_i^t \in \mathbb{F}_2$, with $1 \leq i \leq n$. Moreover, $C^t = (s_1^t, s_2^t, \ldots, s_n^t) \in \mathbb{F}_2^n$ is called configuration of the CA at step of time t.
(3) $\mathcal{N}$ denotes the function which assigns to each node its neighborhood (the adjacent nodes). Then:

$$\mathcal{N}\colon \mathcal{C} \to 2^{\mathcal{C}} \tag{1}$$

$$i \mapsto \mathcal{N}_i = \{i, i_1, i_2, \ldots, i_{m_i}\} \tag{2}$$

Note that as $i \in \mathcal{N}_i$ for every node i, then there is a loop on every node. Moreover, $(i, j) \in E$ iff $i \in \mathcal{N}_j$ or $j \in \mathcal{N}_i$.

(4) $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ is the transition functions family that governs the dynamic of the cellular automaton. The state of the node i at a particular time step $t+1$ is computed by means of the boolean function f_i whose variables are the states of the neighbor nodes at the previous step of time t:

$$s_i^{t+1} = f_i\left(s_i^t, s_{i_1}^t, s_{i_2}^t, \ldots, s_{i_{m_i}}^t\right) \in \mathbb{F}_2. \tag{3}$$

Note that these local transition functions define a global transition function F:

$$\begin{aligned} F\colon \mathbb{F}_2^n &\to \mathbb{F}_2^n \\ C^t &\mapsto C^{t+1} = F\left(C^t\right) \end{aligned} \tag{4}$$

The CA is said to be *linear* if f_i is a linear function for every i, that is:

$$s_i^{t+1} = a_{ii} s_i^t \oplus \bigoplus_{k=1}^{m_i} a_{ii_k} s_{i_k}^t, \quad a_{ii}, a_{ii_k} \in \mathbb{F}_2. \tag{5}$$

In this case, the global dynamics of the CA can be interpreted in terms of matrix algebra as follows: $C^{t+1} = F\left(C^t\right) = A \cdot C^t$, where $A = (a_{ij})_{1 \le i,j \le n}$ is the local transition matrix.

Note that the dynamics of the linear cellular automata is biunivocally determined by the multigraph G (where each edge stands for a XOR summation of the states of the corresponding adjacent nodes). This does not happen in the case of non-linear cellular automata since the same multigraph yields to several families of transition functions.

A cellular automaton is *reversible* when F is bijective, that is, the evolution backwards is possible [12]; in this case $F^{-1}\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ is the global transition function of the inverse cellular automaton. Note that a reversible CA yields an invertible global behavior from a set of local transition functions which are not reversible. The reversibility of a linear CA depends on the nature of its local transition matrix: the linear cellular automaton is reversible iff its local transition matrix A is non-singular and, consequently, A^{-1} is the local transition matrix of the inverse CA.

2.2 Linear CA on Full Trees with Loops

This work deals with linear cellular automata on full trees with loops. A full binary tree is a rooted tree in which each internal vertex has exactly two children. Note that if the full binary tree has k internal vertices then it has $n = 2k+1$ vertices, $n-1$ edges and $\frac{n+1}{2}$ leaves. If there is a loop in each vertex, the notion of full binary tree with loops is derived.

Let $\mathcal{T}_k$ be the set of full binary trees with loops with k internal vertices. Set $T_k \in \mathcal{T}_k$ the tree with loops such that for every node i the following holds:

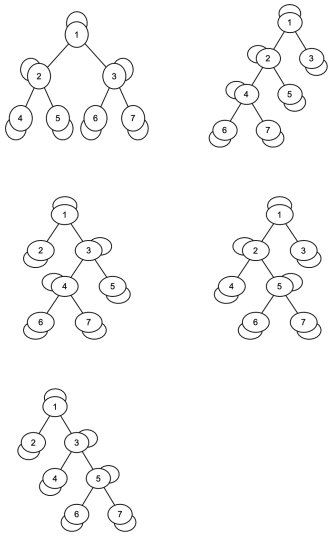

Fig. 1. Family $\mathcal{T}_3$ of full binary trees with loops.

- $i = 1$ is the root.
- If i is even or $i = 2k + 1$ then i is a leaf.
- If i is odd with $i \neq 2k + 1$, then i is an internal node.

This particular tree is called the *characteristic representative* of $\mathcal{T}_k$. In Fig. 1 the family of full tree with loops $\mathcal{T}_3$ is shown (the full binary tree of the last row is the characteristic representative T_3).

A linear cellular automaton on a full tree with loops $T \in \mathcal{T}_k$ is $\mathcal{A}_T = (\mathcal{C}, \mathbb{F}_2, \mathcal{N}, \mathcal{F})$ where $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ is a family of boolean linear functions. In particular, when $\mathcal{A}_{T_k}$ is considered, the following hold:

(1) The root node is $1 \in \mathcal{C}$.
(2) The neighborhood function is defined as follows:

$$\begin{aligned} \mathcal{N}\colon \mathcal{C} &\to 2^{\mathcal{C}} \\ 1 &\mapsto \mathcal{N}_1 = \{1, 2, 3\} \\ i &\mapsto \mathcal{N}_i = \{p_i, i, i^+, i^-\} \\ l &\mapsto \mathcal{N}_l = \{p_l, l\} \end{aligned}$$

where $i \in \mathcal{C}$ is an internal vertex ($i \neq 1$), $l \in \mathcal{C}$ is a leaf, $p_i \in \mathcal{C}$ is the parent of the node i, and $i^+, i^- \in \mathcal{C}$ are the right and left children, respectively, of i.

(3) The local transition functions are as follows:

$$\begin{aligned} s_1^{t+1} &= s_1^t \oplus s_2^t \oplus s_3^t, \\ s_i^{t+1} &= s_{p_i}^t \oplus s_i^t \oplus s_{i^-}^t \oplus s_{i^+}^t, \quad i \neq 1 \text{ internal vertex}, \\ s_l^{t+1} &= s_{p_l}^t \oplus s_l^t, \quad l \text{ leaf}. \end{aligned}$$

Note that the transition matrix of $\mathcal{A}_{T_k}$ is $A = (a_{ij})_{1\leq i,j\leq 2k+1}$ where:

$$a_{ii} = 1, \text{with } 1 \leq i \leq 2k+1 \tag{6}$$

$$a_{i,i+1} = \begin{cases} 1, \text{if } i \text{ is odd} \\ 0, \text{if } i \text{ is even} \end{cases} \text{ with } 1 \leq i \leq 2k \tag{7}$$

$$a_{i,i+2} = \begin{cases} 1, \text{if } i \text{ is odd} \\ 0, \text{if } i \text{ is even} \end{cases} \text{ with } 1 \leq i \leq 2k-1 \tag{8}$$

$$a_{ij} = 0, \text{ if } i+2 \leq j \leq 2k+1 \tag{9}$$

$$a_{ij} = a_{ji}, \text{ with } 1 \leq i,j \leq 2k+1 \tag{10}$$

3 Solving the Reversibility Problem

Theorem 1. *The linear cellular automaton $\mathcal{A}_{T_k}$ is reversible for every k, and its inverse cellular automaton is defined by the symmetric transition matrix $B = (b_{ij})_{1\leq i,j\leq 2k+1}$, where:*

$$b_{ii} = \begin{cases} 1, \text{if } i = 4l+1 \text{ or } i = 4l+4 \text{ with } l \geq 0, l \in \mathbb{N} \\ 0, \text{if } i = 4l+2 \text{ or } i = 4l+3 \text{ with } l \geq 0, l \in \mathbb{N} \end{cases} \tag{11}$$

$$b_{ij} = \begin{cases} 1, \text{if } i = 4l+1 \text{ or } i = 4l+2 \text{ with } l \geq 0, l \in \mathbb{N} \\ 0, \text{if } i = 4l+3 \text{ or } i = 4l+4 \text{ with } l \geq 0, l \in \mathbb{N} \end{cases} \quad j \geq i+1. \tag{12}$$

Proof. To proof the above statement, it is enough to show that the boolean matrix $B = (b_{ij})_{1\leq i,j\leq 2k+1}$ defined by (11)–(12) satisfies the following: $A \cdot B = B \cdot A = Id$.

Let us suppose that $A \cdot B = C$ where $C = (c_{ij})_{1\leq i,j\leq 2k+1}$, then we have to prove that: (1) $c_{ii} = 1$ for $1 \leq i \leq 2k+1$, and (2) $c_{ij} = 0$ for every $i \neq j$.

(1) For the sake of clarity, we will distinguish seven cases:
- Computation of c_{11}: as $a_{14} = a_{15} = \ldots = a_{1,2k+1} = 0$, then

$$\begin{aligned} c_{11} &= \sum_{h=1}^{2k+1} a_{1h} b_{h1} \\ &= a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} + a_{14}b_{41} + \ldots + a_{1,2k+1}b_{2k+1,1} \\ &= 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 + 0 + \ldots + 0 = 3 \\ &\equiv 1 (\text{mod } 2). \end{aligned} \tag{13}$$

- Computation of c_{22}: as $a_{23} = a_{24} = \ldots = a_{2,2k+1} = 0$, then

$$\begin{aligned} c_{22} &= \sum_{h=1}^{2k+1} a_{2h} b_{h2} \\ &= a_{21}b_{12} + a_{22}b_{22} + a_{23}b_{32} + \ldots + a_{2,2k+1}b_{2k+1,2} \\ &= 1 \cdot 1 + 1 \cdot 0 + 0 + \ldots + 0 = 1. \end{aligned} \tag{14}$$

– Computation of c_{33}: as $a_{36} = a_{37} = \ldots = a_{3,2k+1} = 0$, then

$$\begin{aligned} c_{33} &= \sum_{h=1}^{2k+1} a_{3h} b_{h3} \\ &= a_{31}b_{13} + a_{32}b_{23} + a_{33}b_{33} + a_{34}b_{43} + a_{35}b_{53} \\ &\quad + a_{36}b_{63} + \ldots + a_{3,2k+1}b_{2k+1,3} \\ &= 1 \cdot 1 + 0 \cdot 1 + 1 \cdot 0 + 1 \cdot 0 + 1 \cdot 0 + 0 + \ldots + 0 = 1. \end{aligned} \tag{15}$$

– Computation of c_{ii} with $3 \leq i \leq 2k-1$. In this case we can also consider four subcases depending on the value of the subindex i.

- If $i = 4l + 1$ with $l \geq 0$, then:

$$\begin{aligned} c_{ii} = c_{4l+1,4l+1} &= \sum_{h=1}^{2k+1} a_{4l+1,h} b_{h,4l+1} = \sum_{h=4l-1}^{4l+3} a_{4l+1,h} b_{h,4l+1} \\ &= \sum_{h=4l-1}^{4l+3} b_{h,4l+1}, \end{aligned} \tag{16}$$

  since $i = 4l+1$ is odd and, consequently, $a_{4l+1,h} = 1$ for every value of h. Moreover, as $b_{4l-1,4l+1} = b_{4l,4l+1} = 0$ and $b_{4l+1,4l+1} = b_{4l+2,4l+1} = b_{4l+3,4l+1} = 1$ then $c_{4l+1,4l+1} = 3 \equiv 1 (\mathrm{mod}\, 2)$.
- If $i = 4l + 2$ with $l \geq 0$, then:

$$\begin{aligned} c_{ii} = c_{4l+2,4l+2} &= \sum_{h=1}^{2k+1} a_{4l+2,h} b_{h,4l+2} = \sum_{h=4l}^{4l+4} a_{4l+2,h} b_{h,4l+2} \\ &= b_{4l+1,4l+2} + b_{4l+2,4l+2} = 1 + 0 = 1, \end{aligned} \tag{17}$$

  since $a_{4l+2,4l} = a_{4l+2,4l+3} = a_{4l+2,4l+4} = 0$ and $a_{4l+2,4l+1} = a_{4l+2,4l+2} = 1$.
- If $i = 4l + 3$ with $l \geq 0$ then:

$$\begin{aligned} c_{ii} = c_{4l+3,4l+3} &= \sum_{h=1}^{2k+1} a_{4l+3,h} b_{h,4l+3} = \sum_{h=4l+1}^{4l+5} a_{4l+3,h} b_{h,4l+3} \\ &= a_{4l+3,4l+1} + a_{4l+3,4l+2} = 1 + 0 = 1, \end{aligned} \tag{18}$$

  since $b_{4l+1,4l+3} = b_{4l+2,4l+3} = 1$ and $b_{4l+3,4l+3} = b_{4l+4,4l+3} = b_{4l+5,4l+3} = 0$.
- If $i = 4l + 4$ with $l \geq 0$, then:

$$\begin{aligned} c_{ii} = c_{4l+4,4l+4} &= \sum_{h=1}^{2k+1} a_{4l+4,h} b_{h,4l+4} = \sum_{h=4l+2}^{4l+6} a_{4l+4,h} b_{h,4l+4} \\ &= b_{4l+3,4l+4} + b_{4l+4,4l+4} = 0 + 1 = 1, \end{aligned} \tag{19}$$

  since $a_{4l+4,4l+2} = a_{4l+4,4l+5} = a_{4l+4,4l+6} = 0$ and $a_{4l+4,4l+3} = a_{4l+4,4l+4} = 1$.

– Computation of $c_{2k-1,2k-1}$:

$$\begin{aligned} c_{2k-1,2k-1} &= \sum_{h=1}^{2k+1} a_{2k-1,h} b_{h,2k-1} \\ &= a_{2k-1,1} b_{1,2k-1} + \ldots + a_{2k-1,2k-4} b_{2k-4,2k-1} \\ &\quad + a_{2k-1,2k-3} b_{2k-3,2k-1} + a_{2k-1,2k-2} b_{2k-2,2k-1} \\ &\quad + a_{2k-1,2k-1} b_{2k-1,2k-1} + a_{2k-1,2k} b_{2k,2k-1} \\ &\quad + a_{2k-1,2k+1} b_{2k+1,2k-1} \\ &= b_{2k-3,2k-1} + b_{2k-1,2k-1} + b_{2k,2k-1} + b_{2k+1,2k-1}. \end{aligned} \tag{20}$$

As

$$b_{2k-3,2k-1} = \begin{cases} 1, \text{ if } 2k-3 = 4l+1 \iff k \text{ is even} \\ 0, \text{ if } 2k-3 = 4l+3 \iff k \text{ is odd} \end{cases} \tag{21}$$

$$b_{2k-2,2k-1} = \begin{cases} 1, \text{ if } 2k-2 = 4l+2 \iff k \text{ is even} \\ 0, \text{ if } 2k-2 = 4l+4 \iff k \text{ is odd} \end{cases} \tag{22}$$

$$b_{2k,2k-1} = b_{2k-1,2k} = \begin{cases} 1, \text{ if } k \text{ is even} \\ 0, \text{ if } k \text{ is odd} \end{cases} \tag{23}$$

$$b_{2k+1,2k-1} = b_{2k-1,2k+1} = \begin{cases} 1, \text{ if } k \text{ is even} \\ 0, \text{ if } k \text{ is odd} \end{cases} \tag{24}$$

then

$$c_{2k,2k} = \begin{cases} 1+0 = 1, \text{ if } k \text{ is odd} \\ 0+1 = 1, \text{ if } k \text{ is even} \end{cases} \tag{25}$$

– Computation of $c_{2k,2k}$: since $a_{2k,1} = \ldots = a_{2k,2k-3} = 0$, $a_{2k,2k-2} = a_{2k,2k+1} = 0$, and $a_{2k,2k-1} = a_{2k,2k} = 1$, then

$$\begin{aligned} c_{2k,2k} &= \sum_{h=1}^{2k+1} a_{2k,h} b_{h,2k} \\ &= a_{2k,1} b_{1,2k} + \ldots + a_{2k,2k-3} b_{2k-3,2k} \\ &\quad + a_{2k,2k-2} b_{2k-2,2k} + a_{2k,2k-1} b_{2k-1,2k} \\ &\quad + a_{2k,2k} b_{2k,2k} + a_{2k,2k+1} b_{2k+1,2k} \\ &= b_{2k-1,2k} + b_{2k,2k}. \end{aligned} \tag{26}$$

As

$$b_{2k-1,2k} = \begin{cases} 1, \text{ if } 2k-1 = 4l+1 \iff k \text{ is odd} \\ 0, \text{ if } 2k-1 = 4l+3 \iff k \text{ is even} \end{cases} \tag{27}$$

$$b_{2k,2k} = \begin{cases} 1, \text{ if } 2k = 4l+4 \iff k \text{ is even} \\ 0, \text{ if } 2k = 4l+2 \iff k \text{ is odd} \end{cases} \tag{28}$$

then

$$c_{2k,2k} = \begin{cases} 1+0 = 1, \text{ if } k \text{ is odd} \\ 0+1 = 1, \text{ if } k \text{ is even} \end{cases} \tag{29}$$

– Computation of $c_{2k+1,2k+1}$: since $a_{2k+1,1} = \ldots = a_{2k+1,2k-2} = 0, a_{2k+1,2k} = 0$ and $a_{2k+1,2k-1} = a_{2k+1,2k+1} = 1$, then

$$\begin{aligned} c_{2k+1,2k+1} &= \sum_{h=1}^{2k+1} a_{2k+1,h} b_{h,2k+1} \\ &= a_{2k+1,1} b_{1,2k+1} + \ldots + a_{2k+1,2k-2} b_{2k-2,2k+1} \\ &\quad + a_{2k+1,2k-1} b_{2k-1,2k+1} + a_{2k+1,2k} b_{2k,2k+1} \\ &\quad + a_{2k+1,2k+1} b_{2k+1,2k+1} \\ &= b_{2k-1,2k+1} + b_{2k+1,2k+1}. \end{aligned} \tag{30}$$

As

$$b_{2k-1,2k+1} = \begin{cases} 1, \text{ if } 2k-1 = 4l+1 \iff k \text{ is odd} \\ 0, \text{ if } 2k-1 = 4l+3 \iff k \text{ is even} \end{cases} \tag{31}$$

$$b_{2k+1,2k+1} = \begin{cases} 1, \text{ if } 2k+1 = 4l+1 \iff k \text{ is even} \\ 0, \text{ if } 2k+1 = 4l+3 \iff k \text{ is odd} \end{cases} \tag{32}$$

then

$$c_{2k+1,2k+1} = \begin{cases} 1+0 = 1, \text{ if } k \text{ is odd} \\ 0+1 = 1, \text{ if } k \text{ is even} \end{cases} \tag{33}$$

(2) Now, we can distinguish six cases:

– Computation of c_{1j} with $j > 1$: as $a_{1h} = 0$ for $4 \leq h \leq 2k+1$ and $a_{11} = a_{12} = a_{13} = 1$ then

$$c_{1j} = \sum_{h=1}^{2k+1} a_{1h} b_{hj} = b_{1j} + b_{2j} + b_{3j} = 0. \tag{34}$$

– Computation of c_{2j} with $j > 2$: as $a_{2h} = 0$ for $h \geq 3$ and $a_{21} = a_{22} = 1$ then

$$c_{2j} = \sum_{h=1}^{2k+1} a_{2h} b_{hj} = b_{1j} + b_{2j} = 1 + 1 = 2 \equiv 0 (\mathrm{mod}\, 2). \tag{35}$$

– Computation of c_{3j} with $j > 3$: as $a_{3h} = 0$ for $h \geq 6$, $a_{31} = a_{33} = a_{34} = a_{35} = 1$, and $a_{32} = 0$ then

$$\begin{aligned} c_{3j} &= \sum_{h=1}^{2k+1} a_{3h} b_{hj} = b_{1j} + b_{3j} + b_{4j} + b_{5j} \\ &= 1 + 1 + 0 + 0 = 2 \equiv 0 (\mathrm{mod}\, 2). \end{aligned} \tag{36}$$

– Computation of c_{ij} with $4 < i < 2k$ and $j > i$: as $a_{i,j} = 0$ for $j < i-2$ and $j > i+2$ then

$$\begin{aligned} c_{ij} &= a_{i,i-2} b_{i-2,j} + a_{i,i-1} b_{i-1,j} + b_{ij} \\ &\quad + a_{i,i+1} b_{i+1,j} + a_{i,i+2} b_{i+2,j}. \end{aligned} \tag{37}$$

If i is even then $c_{ij} = b_{i-1,j} + b_{ij}$ since $a_{i,i-2} = a_{i,i+1} = a_{i,i+2} = 0$ and $a_{i,i-1} = 1$. Consequently:

$$\begin{aligned} c_{ij} &= b_{i-2,j} + b_{ij} + b_{i+1,j} + b_{i+2,j} \\ &= \begin{cases} 0+1+1+0 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } i = 4l+1 \\ 1+0+0+1 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } i = 4l+3 \end{cases} \end{aligned} \tag{38}$$

On the other hand, if i is odd then $c_{ij} = b_{i-2,j} + b_{ij} + b_{i+1,j} + b_{i+2,j}$ since $a_{i-1,i} = 0$ and $a_{i,i-2} = a_{i,i+1} = a_{i,i+2} = 1$. As a consequence

$$c_{ij} = b_{i-1,j} + b_{ij} = \begin{cases} 1+1 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } i = 4l+2 \\ 0+0 = 0, \qquad\qquad\quad \text{if } i = 4l+4 \end{cases} \tag{39}$$

– Computation of $c_{2k-1,j}$ with $j > 2k-1$, that is, $j = 2k, 2k+1$. For the sake of simplicity we will distinguish two subcases:
  - If $j = 2k$ then

$$\begin{aligned} c_{2k-1,2k} &= \sum_{h=1}^{2k+1} a_{2k-1,h} b_{h,2k} \\ &= b_{2k-3,2k} + b_{2k-1,2k} + b_{2k,2k} + b_{2k+1,2k} \\ &= \begin{cases} 1+0+1+0 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } k \text{ is even} \\ 0+1+0+1 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } k \text{ is odd} \end{cases} \end{aligned} \tag{40}$$

  - If $j = 2k+1$ then:

$$\begin{aligned} c_{2k-1,2k+1} &= \sum_{h=1}^{2k+1} a_{2k-1,h} b_{h,2k+1} \\ &= b_{2k-3,2k+1} + b_{2k-1,2k+1} + b_{2k,2k+1} + b_{2k+1,2k+1} \\ &= \begin{cases} 1+0+0+1 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } k \text{ is even} \\ 0+1+1+0 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } k \text{ is odd} \end{cases} \end{aligned} \tag{41}$$

– Computation of $c_{2k,j}$ with $j > 2k$, that is, $j = 2k+1$: since $a_{2k,h} = 0$ for $1 \leq h \leq 2k-2$, $a_{2k,2k-1} = a_{2k,2k} = 1$ and $a_{2k,2k+1} = 1$ then

$$c_{2k,2k+1} = \sum_{h=1}^{2k+1} a_{2k,h} b_{h,2k+1} = b_{2k-1,2k+1} + b_{2k,2k+1}. \tag{42}$$

As

$$b_{2k-1,2k+1} = \begin{cases} 1, \text{ if } 2k-1 = 4l+1 \iff k \text{ is odd} \\ 0, \text{ if } 2k-1 = 4l+3 \iff k \text{ is even} \end{cases} \tag{43}$$

$$b_{2k,2k+1} = \begin{cases} 1, \text{ if } 2k = 4l+2 \iff k \text{ is odd} \\ 0, \text{ if } 2k = 4l+4 \iff k \text{ is even} \end{cases} \tag{44}$$

then

$$c_{2k,2k+1} = \begin{cases} 1+1 = 2 \equiv 0 (\mathrm{mod}\, 2), \text{ if } k \text{ is odd} \\ 0+0 = 0, \qquad\qquad\quad \text{if } k \text{ is even} \end{cases} \tag{45}$$

Using a similar argument, we can also prove that $B \cdot A = Id$.

4 Conclusions and Further Work

In this work the notion of linear cellular automaton on full trees with loops is introduced and its reversibility problem is studied. Specifically, it is shown that some linear cellular automata of this type are reversible and the inverse cellular automaton is explicitly computed.

Future work aimed at solving the complete reversibility problem for all $\mathcal{A}_T$ with $T \in \mathcal{T}_k$, and studying the applications of $\mathcal{A}_T$ in different fields such as f-reversible processes on graphs or digital image processing.

Acknowledgements. This research has been partially supported by Ministerio de Ciencia, Innovación y Universidades (MCIU, Spain), Agencia Estatal de Investigación (AEI, Spain), and Fondo Europeo de Desarrollo Regional (FEDER, UE) under projects with references TIN2017-84844-C2-2-R (MAGERAN) and MTM2017-86042-P, and the project with reference SA054G18 supported by Consejería de Educación (Junta de Castilla y León, Spain).

References

1. Augustynowicz, A., Baetens, J.M., De Baets, B., et al.: A note on the reversibility of 2D cellular automata on hexagonal grids. J. Cell. Autom. **13**, 521–526 (2018)
2. Bhattacharjee, K., Das, S.: Reversibility of d-state finite cellular automata. J. Cell. Autom. **11**, 213–245 (2016)
3. Chang, C.H., Chang, H.: On the Bernoulli automorphism of reversible linear cellular automata. Inf. Sci. **345**(1), 217–225 (2016)
4. Chang, C.H., Su, J.Y., Akin, H., et al.: Reversibility problem of multidimensional finite cellular automata. J. Stat. Phys. **168**(1), 208–231 (2017)
5. Di Gregorio, S., Trautteur, G.: On reversibility in cellular automata. J. Comput. Syst. Sci. **11**(3), 382–391 (1975)
6. Hoekstra, A.G., Kroc, J., Sloot, P.M.A.: Simulating Complex Systems by Cellular Automata. Springer, Berlin (2010)
7. Morita, K.: Reversible computing and cellular automata-a survey. Theor. Comput. Sci. **395**, 101–131 (2008)
8. Richardson, D.: Tessellations with local transformations. J. Comput. Syst. Sci. **6**, 373–388 (1972)
9. Sarkar, P.: A brief history of cellular automata. ACM Comput. Surv. **32**, 80–107 (2000)
10. Sarkar, P., Barua, R.: The set of reversible 90/150 cellular automata is regular. Discret. Appl. Math. **84**, 199–213 (1998)
11. Seck-Tuoh-Mora, J.C., Martínez, G.J., Alonso-Sanz, R., Hernández-Romero, N.: Invertible behavior in elementary cellular automata with memory. Inf. Sci. **199**, 125–132 (2012)
12. Toffoli, T., Margolus, N.H.: Invertible cellular automata: a review. Physica D **45**, 229–253 (1990)
13. Wang, X., Luan, D.: A novel image encryption algorithm using chaos and reversible cellular automata. Commun. Nonlinear Sci Numer. Simul. **18**, 3075–3085 (2013)