Fractal Arrangement for 2D Cellular Automata and Its Implementation for Outer-Totalistic Rules

Yoshihiko Kayama(✉), Yuka Koda, and Ikumi Yazawa

Department of Media and Information, BAIKA Women's University,
2-19-5 Shukuno-sho, Osaka, Ibaraki 567-8578, Japan
y_kayama@ieee.org

Abstract. Cellular automata (CAs) have played a significant role in studies of complex systems. Recently, a recursive estimation of neighbors algorithm that distinguishes the perception area of each cell from the CA rule neighborhood was introduced to extend CA. This framework makes it possible to construct non-uniform CA models composed of cells with different sizes of the perception area, which can be interpreted as an individual attribute of each cell. For example, focusing primarily on one-dimensional (1D) elementary CA, fractal CAs composed of self-similarly arranged cells have been proposed and their characteristics have been investigated. In this paper, 2D fractal CAs are defined and implemented for outer-totalistic CA rules. Fractal CAs derived from a linear rule inherit that rule's features, including replicability and time reversibility, which indicate their applicability to various fields.

1 Introduction

Cellular automata (CAs), which were first introduced by von Neumann and Ulam to model biological self-reproduction [1], are discrete computational systems that have played a significant role in the study of complex systems. CAs comprise a set of cells arranged on a regular lattice where each cell in an initial state is taken from a finite set. The state is updated at each time step according to a local rule based on its own state and the states of a fixed set of neighboring cells. Such CAs are uniform and synchronous, i.e., all cells apply the same local rule and are updated synchronously, and are referred to as *standard* CAs. Various extended CA models that are of theoretical and practical interest have been investigated by relaxing the characteristics of standard CA. Recently, based on the *recursive estimation of neighbors* (REN) algorithm, a method to construct non-uniform CAs in which each cell is allowed to follow a different local rule has been proposed [2,3]. The REN algorithm, a framework inspired by that of Reynolds' Boids program [4], takes a standard CA rule with a unit rule radius and extends it to rules with larger radii other than the unit rule radius. The perception area of each cell, which is defined by the value of the extra radius, is no longer identical to the neighborhood specified by the standard CA rule.

G. Mauri et al. (Eds.): ACRI 2018, LNCS 11115, pp. 328–339, 2018.
https://doi.org/10.1007/978-3-319-99813-8_30

In the following, cells within the neighborhood of a cell defined by the standard CA rule are referred to as *neighbors*. The standard CA rule is used recursively to estimate the next states of the neighbors from the present states of cells within the perception area. Moreover, the extended rules form a sequence indexed by the value of the extra radius, which contains the standard CA rule, referred to as the *basic rule*, as its first term. Even though the rules in the sequence are obtained from the extension of a basic rule via REN, each extended rule corresponds to a standard CA rule with an equal value of the rule radius to its extra radius as a rule mapping cell configurations of the perception area of a target cell to its next state. In other words, extension using REN relates a standard CA rule with a unit radius to others with longer radius values.

A non-uniform CA can be constructed from cells that follow distinct extended rules that belong to the same sequence of extended rules. Among various possible cell arrangements, those with fractal geometries are particularly interesting because such geometries, such as Koch's curve and Sierpinski's gasket, have a property known as self-similarity. Fractal structures also play an important role in complex systems in nature, such as biological structures, Internet connections, and social networks. Such non-uniform CA that comprise fractally arranged cells, i.e., fractal CA (F-CA), have been proposed [5]. The attractive characteristics of basic rules, e.g., pattern replicability and reversibility in linear rules of 1D elementary CA (ECA), are carried over into their F-CA. Here we focus on F-CAs derived from 2D CA rules. A practical implementation is discussed in consideration of outer-totalistic rules. Similar to fractal ECA, some characteristics of linear basic rules in outer-totalistic CA are inherited by their F-CAs.

The remainder of this paper is organized as follows. Section 2 explains extension using REN. A practical extension of 2D outer-totalistic CA rules is described in Sect. 3. Section 4 describes the construction of 2D F-CA from the sequence of extended rules and implements the F-CA construction for outer-totalistic rules. In addition, as potential applications of F-CA derived from a multi-state linear outer-totalistic rule, a diffusion process of encryption systems and textile design samples are presented. Conclusions and suggestions for future work are given in Sect. 5.

2 Extension of CA Using REN

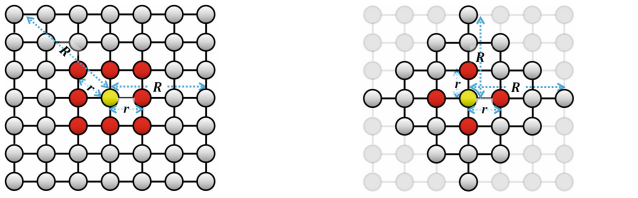
In case of Reynolds' Boids program, each boid acquires information regarding the positions and velocities of other boids within its perception area and determines its own movement to follow the representative values of the neighbors. The radius of the perception area can be treated as a parameter differentiating individual elements. To incorporate a similar scenario in a CA, the perception area of a cell should be separated from the neighborhood determined by the CA rule, so that the size of the area can be treated as an attribute of each cell. Under the standard CA framework, however, there is no scope for expanding the sensory area of a cell. For example, each cell of ECA acquires the states of the three cells within its radius-one neighborhood to determine its own state in the next

time step. Such separation can be possible if the update process of each cell has an intermediate process of estimating next states of neighboring cells, given as follows:

$$\text{Acquire information about neighbors} \Rightarrow \text{estimate their next states} \Rightarrow \text{determine its own nextstate} \quad (1)$$

Estimation and determination of states are assumed to be processed by only a basic CA rule because if other rules or mechanisms were introduced, the present framework would become complicated and finding a reasonable selection method would, therefore, be difficult. Moreover, it is assumed that all cells use the same update algorithm. Then, the basic rule will be used recursively as explained in the next subsection.

When a standard CA rule, i.e., the basic rule, is extended using REN, any extended rule is assumed to have a larger cell perception area than the neighborhood defined by the basic rule. The neighborhood and the perception area are parameterized by their respective radii, r and R, where r is the common radius of the neighborhoods of all cells defined by the basic rule, and R is the extra radius given by the perception area of each cell. As illustrated in Fig. 1, the neighbors in the neighborhood are included in the perception area. The value of the extra radius representing the size of the perception area of a target cell can now be recognized as its independent attribute.



(a) 8 neighbors (Moore) (b) 4 neighbors (von Neumann)

Fig. 1. Perception area of a target (yellow cell) including its neighbors (red cells) in 2D extended CAs with (a) Moore and (b) von Neumann neighborhoods. Each target cell has an extra radius $R = 3$ and the radius of the basic rule $r = 1$. (Color figure online)

The basic rule is used recursively in the REN process to estimate the *next states* of the neighbors of a target cell and to determine the next state of the target cell by applying it to the *estimated next states* of the neighbors and the *current state* of the target. The update process (1) is expressed by the following steps.

1. Perceive the current states of all cells within the perception area of a target.
2. Apply REN to estimate the next states of the neighbors.

3. Determine the next state of the target by applying the basic rule to the neighbors' estimated next states and to the target's current state.

In the second step, the next states of the target's neighbors are estimated as a set $\left\{\varphi_{neighbors}^{(t+1)}\right\}$ by applying REN to the information in the set of current states of the cells within the perception area $\left\{x_{p-area}^{(t)}\right\}$ (first step). Here, x and φ represent the actual and estimated states of a cell, respectively. { } indicates the set of states of cells within an area or a group. The next state of the target is estimated as $\varphi_{target}^{(t+1)}$ by applying the basic rule to $\left\{\varphi_{neighbors}^{(t+1)}\right\}$ together with its current state $x_{target}^{(t)}$. Finally, $\varphi_{target}^{(t+1)}$ is assigned to $x_{target}^{(t+1)}$, i.e., the next actual state of the target.

The estimated next states of the neighbors are not necessarily identical to their actual next states because the estimation requires information about the neighbors' extra radius values, and it is assumed that each cell cannot perceive such information. The REN algorithm includes an assumption about the estimation of the extra radius values of neighbors as mentioned in the next subsection. Given that we focus primarily on extending 2D eight-neighbor (or four-neighbor) CA rules, r is set to one in the following, where neighbors are adjacent to each cell. In the time evolution of a cell, only the basic rule is used recursively, regardless of the value of R. In that sense, R can be an attribute of each cell, thereby allowing the construction of non-uniform models containing cells with different R values. Note that this differs from standard CAs, which are always uniform.

2.1 Recursive Estimation of Neighbors

The recursive nature of REN comes from an assumption of self-similarity, i.e., the next state of each cell in a perception area is determined by the previously described three steps. A target cell's immediate neighbors estimate the states of their neighbors, which are denoted as $neighbors^{(1)}$. To describe REN more concretely, we set the radius R of the target to an integer k. As mentioned previously, the target can perceive the current states of its neighbors because they are all contained within its perception area (first step). However, the sizes of the neighbors' perception areas are assumed to be unperceivable by the target. Therefore, to estimate the neighbors' next states $\left\{\varphi_{neighbors}^{(t+1)}\right\}$, the target must evaluate their sizes. Here, we assume that the target cell estimates a perception area that is *as large as possible* for each neighbor within its own perception area. Thereafter, the radius value of the neighbor's perception area is assumed to be $k-1$. The target cell then attempts to estimate the next state of each of its neighbors by assuming that a neighbor applies the same steps, i.e., the neighbor can be considered the next target, $target^{(1)}$, and its next state will be estimated by the next states of its neighbors (i.e., the neighbors' neighbors: $neighbors^{(1)}$) and its present state using REN. At this time, the size of the perception area of each of the $neighbors^{(1)}$ will be evaluated as $k-2$. Next, if each of the $neighbors^{(1)}$ is considered the next target, i.e., $target^{(2)}$, the perception

area size of its neighbors ($neighbors^{(2)}$) will be evaluated as $k-3$. Similarly, each of the $neighbors^{(2)}$ can be considered the next target, i.e., $target^{(3)}$, and the perception area of its neighbors ($neighbors^{(3)}$) will be evaluated as $k-4$. Eventually, the chain of neighbors will reach the edge of the perception area of the original target. Such a boundary cell, i.e., $target^{(k)}$, will have no neighbors; therefore, the basic rule cannot be applied. Here, we add the further assumption that the next state of the boundary cell will be estimated as being the same as the current state. Note that this is the REN algorithm's *termination condition*:

$$\varphi^{(t+1)}_{neighbor^{(k-1)}} = \varphi^{(t+1)}_{target^{(k)}} = x^{(t)}_{target^{(k)}}. \tag{2}$$

where $\varphi^{(t+1)}_{neighbor^{(k-1)}} \in \left\{\varphi^{(t+1)}_{neighbors^{(k-1)}}\right\}$. Finally, the next states of all cells within the perception area are estimated recursively using the basic rule.

3 Extension of 2D Outer-Totalistic Rules

Outer-totalistic implies that the rule function depends on the sum of the states of the outer neighbors, i.e., all cells except the center cell within the neighborhood defined by a CA rule. When the state of the (i,j)-th cell at time step t and the CA rule function are denoted $x^{(t)}_{(i,j)}$ and f, respectively, the standard time evolution of the state is given as follows:

$$\text{Std.CA: } x^{(t+1)}_{(i,j)} = f(x^{(t)}_{(i,j)}, \sigma_8(i,j)),\ \sigma_8(i,j) = \sum_{nb(i,j)} x^{(t)}_{nb(i,j)} \tag{3}$$

where $\sigma_8(i,j)$ represents the sum of the states of the eight cells neighboring the target cell with the Moore neighborhood[1] (Fig. 1a), and $nb(i,j)$ represents each position of the target's immediate neighbors, such as the following.

$$\begin{aligned} nb(i,j) \in \{&(i-1,j-1),(i,j-1),(i+1,j-1),(i-1,j),\\ &(i+1,j),(i-1,j+1),(i,j+1),(i+1,j+1)\}. \end{aligned} \tag{4}$$

Next, we demonstrate the time evolution process in extended 2D outer-totalistic CA. Here, an extended CA is assumed to be uniform such that all cells have the same value of $R = k$. The time evolution of the (i,j)-th cell requires the sum of the *estimated* states of its neighbors at $t+1$ and its current state $x^{(t)}_{k,(i,j)}$, as mentioned in the third step discussed in the previous section. When the sum is denoted $\sigma_8(i,j;k-1)$, Eq. (3) becomes:

$$\varphi^{(t+1)}_{k,(i,j)} = f(x^{(t)}_{k,(i,j)}, \sigma_8(i,j;k-1)), \tag{5}$$

where $\varphi^{(t+1)}_{k,(i,j)}$ is the estimated state of the target, which is assigned as the actual next state $x^{(t+1)}_{k,(i,j)}$. The sum $\sigma_8(i,j;k-1)$ can be expressed as follows:

[1] CAs with the von Neumann neighborhood (Fig. 1b) can be extended through similar steps.

$$\sigma_8(i,j;k-1) = \sum_{nb(i,j)} \varphi^{(t+1)}_{k-1,nb(i,j)} \tag{6}$$

where $\varphi^{(t+1)}_{k-1,nb(i,j)}$ is the estimated state of each neighbor at $t+1$ with an *assumed* radius R of $k-1$. This value comes from the assumptions of the REN algorithm because $k-1$ is the maximum value of the perception area for the immediate neighbors within the perception area of the target with $R=k$. However, each $\varphi^{(t+1)}_{k-1,nb(i,j)}$ is not necessarily equal to its respective actual state $x^{(t+1)}_{k,nb(i,j)}$ because the true value of R of the neighbors is not $k-1$ but k in this uniform case.

Following the procedure mentioned in Sect. 2.1, the REN algorithm produces the following recursive expressions for the estimated states of the m-th immediate neighbors ($neighbors^{(m)}$):

$$\varphi^{(t+1)}_{k-m,nb^{(m)}(i,j)} = f(x^{(t)}_{k,nb^{(m)}(i,j)}, \sigma_8(nb^{(m)}(i,j);k-m-1)) \tag{7}$$

$$\sigma_8(nb^{(m)}(i,j);k-m-1) = \sum_{nb^{(m+1)}(i,j)} \varphi^{(t+1)}_{k-m-1,nb^{(m+1)}(i,j)}, \tag{8}$$

where $nb^{(m)}(i,j) = \overbrace{nb(\cdots(nb}^{m}(i,j))\cdots)$, $m=1,2,\cdots,k-1$. Given that $m=k$ implies that the estimated value of R will be equal to 0 ($<r=1$), the following termination condition ends the recursion.

$$\varphi^{(t+1)}_{0,nb^{(k)}(i,j)} = x^{(t)}_{k,nb^{(k)}(i,j)}. \tag{9}$$

Once the next states of the k-th neighbors are determined, we can go back to Eq. (5) by using the above recursive expressions.

As a concrete demonstration, we begin by considering the case where $k=r=1$. Equations (5) and (9) yield $x^{(t+1)}_{1,(i,j)} = f(x^{(t)}_{1,(i,j)}, \sigma_8(i,j;0))$ and $\varphi^{(t+1)}_{0,nb(i,j)} = x^{(t)}_{1,nb(i,j)} = x^{(t)}_{nb(i,j)}$ respectively. These mean that $\sigma_8(i,j;0) = \sigma_8(i,j)$ (Eq. (6)), such that the extended CA rule with $R=1$ is identical to the basic rule (Eq. (3)). In the next case, where $R=2$, Eq. (5) gives $x^{(t+1)}_{2,(i,j)} = f(x^{(t)}_{2,(i,j)}, \sigma_8(i,j;1))$ and the recursive expressions (Eqs. (7) and (8)) give the following:

$$\varphi^{(t+1)}_{1,nb(i,j)} = f(x^{(t)}_{nb(i,j)}, \sigma_8(nb(i,j);0)), \tag{10}$$

$$\sigma_8(nb(i,j);0) = \sum_{nb^{(2)}(i,j)} \varphi^{(t+1)}_{0,nb^{(2)}(i,j)}. \tag{11}$$

Owing to the termination condition $\varphi^{(t+1)}_{0,nb^{(2)}(i,j)} = x^{(t)}_{nb^{(2)}(i,j)}$ (Eq. (9)), the extended CA rule with $R=2$ is expressed as follows:

$$x^{(t+1)}_{(i,j)} = f(x^{(t)}_{(i,j)}, \sum_{nb(i,j)} f(x^{(t)}_{nb(i,j)}, \sum_{nb^{(2)}(i,j)} x^{(t)}_{nb^{(2)}(i,j)})), \tag{12}$$

which can be considered a standard CA rule with $r = 2$. Considering that cases in which R takes larger values can be derived in the same manner, each extended rule is one of the standard CA rules with such large rule radius $r = R$. Eventually, the extended rules form a sequence indexed by the value of R, and the first term is identical to the basic rule.

Furthermore, if the rule function f of a basic rule is independent of the state of the (i, j) cell, i.e.,

$$\text{Std.CA: } x_i^{(t+1)} = f(\sigma_8(i,j)) = f(\sum_{nb(i,j)} x_{nb(i,j)}^{(t)}), \tag{13}$$

Equation (12) becomes

$$x_{(i,j)}^{(t+1)} = f(\sum_{nb(i,j)} f(\sum_{nb^{(2)}(i,j)} x_{nb^{(2)}(i,j)}^{(t)})). \tag{14}$$

The right hand side of Eq. (14) is identical to

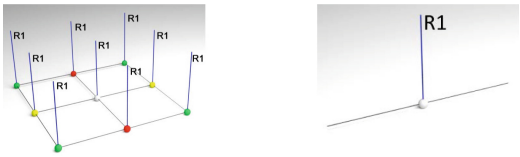
$$\text{Std.CA: } x_i^{(t+2)} = f(\sum_{nb(i,j)} x_{nb(i,j)}^{(t+1)}) = f^{(2)} \bullet x_i^{(t)}, \tag{15}$$

thus indicating that the extended rule with $R = 2$ is identical to two evolutions of the basic rule. According to the similar discussion of cases with larger values of R, the sequences of extended rules derived from a basic rule independent of the state of the center cell are identical to the time evolutions of the basic rule.

When the basic rule is assigned the code $\mathcal{N}$, the sequence formed by its extended rules is represented as $[\mathcal{N}]$. If each rule in the sequence is identified, it is denoted by the code of the basic rule followed by the letter R, indicating the extra radius and its value k. Therefore, $[\mathcal{N}]$ can be enumerated as $\{\mathcal{N}\text{R1}, \mathcal{N}\text{R2}, \mathcal{N}\text{R3}, \cdots\}$, where $\mathcal{N}$R1 is identical to the basic rule, as discussed above. In the following, a cell with the value k for its attribute R or a cell that follows the rule $\mathcal{N}\text{R}k$ is referred to as an Rk cell. For example, one of the most famous 2D CA, i.e., Conway's Game of Life (GoL) [6,7], can be specified as B3S23 in the Golly/RLE format [8,9]. The sequence of extended rules derived from the GoL rule is denoted [B3S23] = {B3S23R1, B3S23R2, B3S23R3, $\cdots$}, and the first term B3S23R1 is identical to the GoL rule.

4 2D Fractal CA

The extension of a basic rule enables the construction of non-uniform CAs in which cells take different values for the extra radius R or follow different extended rules that belong to the sequence originating from the basic rule. This allows CAs with self-similar fractal structures to be derived as a special arrangement of the cells using the classical initiator-generator method [10]. In Sect. 4.2, F-CAs for 2D outer-totalistic CA rules are discussed, whereas those for 1D elementary CA, or F-ECAs, have been studied in the literature [5].

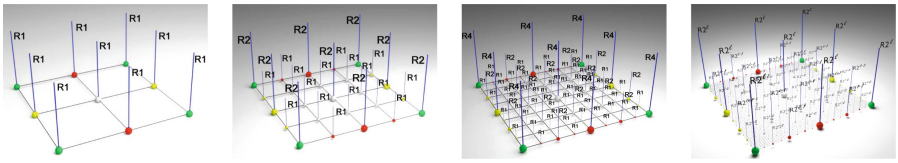


(a) Initiator (b) Generator

Fig. 2. Initiator and generator of 2D F-CA. The white circles are cells, and the black lines are the links connecting them. The blue lines with an R1 sign represent the values of the extra radius R of the cells. In Fig. 2a, the four green cells at the corners are identical, and the two pairs of side cells (front and back (red) and left and right (yellow)) are also identical according to the periodic boundary conditions. (Color figure online)

4.1 2D Fractal Arrangement

To construct a self-similar fractal arrangement of cells that follows the extended rules in a sequence, an initiator and generator set must be defined. Given that the following discussion applies a periodic boundary condition (i.e., a torus), a F-CA with a Moore neighborhood begins with a 2×2 regular lattice, where the four R1 cells (green) at the corners are identical, and the two pairs of R1 cells (front and back (red) and left and right (yellow)) are also identical (Fig. 2a). Note that a generator can be adopted as two links with an R1 cell (Fig. 2b). Figure 3 shows that the level zero F-CA is identical to the initiator and that the level l F-CA is generated by the generator by replacing all links of the level $(l-1)$ F-CA. Eventually, the total number of independent cells becomes $2^{2(l+1)}$ at level l because the number of $\mathrm{R}2^l$ cells is four and that of $\mathrm{R}2^{l-m}$ cells is 3×2^{2m} $(m = 1 \ldots l)$ due to the periodic boundary conditions (Fig. 3d).



(a) Level 0 (b) Level 1 (c) Level 2 (d) Level l

Fig. 3. 2D F-CA with Moore neighborhood. The four green cells at the corners are identical, and the two pairs of side cells (front and back (red) and left and right (yellow)) are also identical due to the periodic boundary conditions. The level l F-CA is generated by replacing all links of the level $(l-1)$ F-CA (Fig. 2b). (Color figure online)

4.2 2D Fractal Outer-Totalistic CA

If we restrict examples to life-like CAs, which are outer-totalistic *binary* CAs (including the GoL), each rule can be denoted $\mathrm{B}b_1b_2\cdots\mathrm{S}s_1s_2\cdots$ in the

Golly/RLE format, where B and S mean "Born" and "Survival," respectively, and

$$b_1, b_2, \cdots, s_1, s_2, \cdots \in \{0, 1, 2, 3, 4, 5, 6, 7, 8\},$$

2D fractal life-like CAs can be constructed by arranging the cells presented above. As noteworthy examples, "Replicators" (B1357S02468 and B1357S1357 [11]) are expressed as follows:

$$f_{B1357S02468}(x^{(t)}_{(i,j)}) = \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} \oplus x^{(t)}_{(k,l)}, \tag{16}$$

$$f_{B1357S1357}(x^{(t)}_{(i,j)}) = x^{(t)}_{(i,j)} \oplus \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} \oplus x^{(t)}_{(k,l)}, \tag{17}$$

where $\oplus$ represents the exclusive-OR (XOR) operation. Because the latter function $f_{B1357S1357}$ is substantially independent of the current state of the center cell $x^{(t)}_{(i,j)}$ by the XOR between $x^{(t)}_{(i,j)}$ and $\sum\sum \oplus x^{(t)}_{(k,l)}$, the sequence [B1357S1357] is identical to the time evolution of a standard CA B1357S1357, as remarked in Sect. 3. As shown in Fig. 4, its F-CA, i.e., F-CA[B1357S1357] exhibits an interesting feature that every group of cells separated by the R value maintains an independent lifetime each: a group of cells with $R = 2^n$ has a lifetime of 2^n time steps, which means that all cell states of the group become zero from almost initial configurations after the lifetime and that each group evolves independently. The cause of the phenomena can be attributed to the coefficients of the extended rules in [B1357S1357] which form two-dimensional Sierpinski's gasket in the same manner that those of the extended rules in [#90] of ECA form one-dimensional Sierpinski's gasket [5].

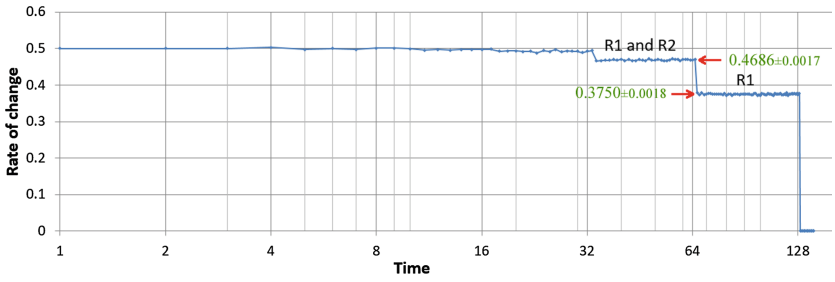


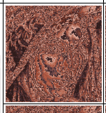
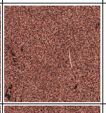
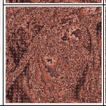
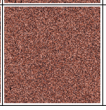
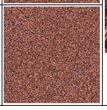
Fig. 4. A sample plot of rates of change of cell states in a level 7 F-CA[B1357S1357], starting from a pseudorandomly generated initial configuration. The red arrows indicate the averaged values for $time = 33 - 64$ and $65 - 128$, respectively. The values $0.4686 \pm 0.0017(std)$ and $0.3750 \pm 0.0018(std)$ correspond to the half values of the number rate of R1 and R2 cells $\frac{3\times 2^{14}}{2^{16}} + \frac{3\times 2^{12}}{2^{16}} = 0.9375$ and that of R1 cells $\frac{3\times 2^{14}}{2^{16}} = 0.75$. (Color figure online)

As an attractive example among fractal ECAs, the time reversibility of linear rule #150 is inherited by its F-CA, denoted F-ECA[#150] [5]. Another linear rule B1357S02468 (Eq. 16) is also time reversible and leads to reversible F-CA[B1357S02468]. Although time reversibility for any level has not been proven mathematically, no counterexamples have been found to date[2]. The fractalization of the CA presented here is independent of the number of cell states; thus, multi-state CA rules can be adopted for the basic rules. If we assume linear 2^n-state (or modulo-2^n) CA [12], the above total sum is expressed by the modulo-2^n operation:

$$f_{modulo-2^n}(x^{(t)}_{(i,j)}) = \Big(\sum_{k=i-1}^{i+1}\sum_{l=j-1}^{j+1} x^{(t)}_{(k,l)}\Big) \bmod 2^n,\ (n = 1, 2, \cdots). \tag{18}$$

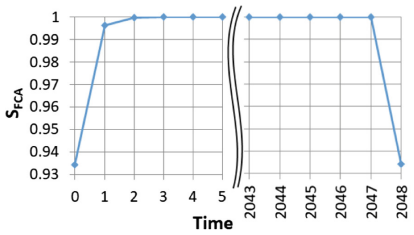
Note that the F-CA constructed from the above linear modulo-2^n rule also shows time reversibility. As a potential application, this may work as a diffusion algorithm for image encryption systems [13]. There are some possible advantages of such models, e.g., each cell can be used to handle an individual character or image pixel as is, and fewer time steps are required to fully scramble plain data. Specifically, from the above linear rule, the level l arrangement of the F-CA shows reversibility with a period 2^{l+n}. Table 1 illustrates the time reversibility of the uniform CA (R1 cells only) of the rule with $n = 4$ in Eq. (18) (16 colors) and the F-CA arrangement with level 7 (lattice size: 256×256). Figure 5a shows the changing averaged entropy of the cell-state frequencies of the process shown in Table 1. The rapid scrambling of the F-CA arrangement can be recognized by comparing the uniform CA and F-CA in Fig. 5b.

Table 1. Time reversibility of CA constructed from the 2D 16-state linear rule (Eq. (18)). The number of colors of the original Lenna image was reduced to $2^4 = 16$. The second and third rows show the time evolutions of the uniform CA (R1 cells only) and level 7 F-CA[modulo-2^4 linear], respectively. The period of the F-CA equals $2^{7+4} = 2048$ time steps.

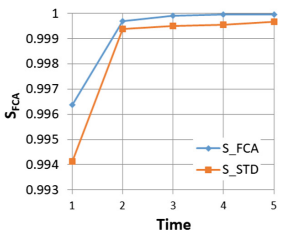
Time	0: initial	1	2	···	2047	2048
Uniform CA				···		
Fractal CA				···		

As another example, fractally symmetric patterns generated from time advances of the F-CA can be used to design textiles. Table 2 shows sample pat-

[2] The time reversibility of F-CA[B1357S02468] was proved until level 2 by a round-robin check of all configurations.



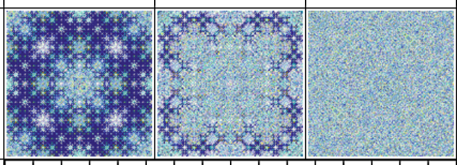
(a) F-CA $time = 0 \sim 5, 2043 \sim 2048$



(b) Initial scramblings

Fig. 5. Entropy changes of the cases in Table 1: (a) F-CA[modulo-2^4 linear] at $time = 0 \sim 5$ and $2043 \sim 2048$, and (b) comparison of initial scramblings between the uniform CA (S_{STD}) and F-CA[modulo-2^4 linear] (S_{FCA}).

Table 2. Sample patterns of textile design using the level 8 F-CA[modulo-2^4 linear]. Each pattern appears from the respective initial state after the time steps. A color pallet showing the correspondence of cell states and colors is common to these three cases.

Initial State	Two R=2^8 cells at center and top-left	R=2^8 cell at center	Two R=2^8 cells at center and top-left
Time	4	5	30
F-CA level 8 16 colors			
Pallet	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15		

terns generated from the F-CA of the linear modulo-2^n CA. Note that different initial configurations can reduce completely different patterns.

5 Conclusions and Discussion

The extension of standard CA rules using the REN algorithm allows the construction of non-uniform CA comprising cells with different sized perception areas. In this paper, we have proposed 2D F-CA by arranging such cells self-similarly and presented an implementation for outer-totalistic CA rules. By focusing on the extension of the linear CA rules, their features, such as replicability and reversibility, are carried over into their F-CA. In addition, image scrambling and textile design samples have been presented as specific application examples.

Note that the mathematical proof of the reversibility of F-CA[B1357S02468] should be provided, and its availability for encryption systems and the inde-

pendent lifetimes of the cell groups of F-CA[B1357S1357] requires additional detailed discussion. Survey of F-CAs other than F-CA[modulo-2^n linear] is also the focus of our future work.

References

1. von Neumann, J.: The theory of self-reproducing automata. In: Burks, A.W. (ed.) Essays on Cellular Automata. University of Illinois Press (1966)
2. Kayama, Y.: Extension of cellular automata by introducing an algorithm of recursive estimation of neighbors. In: Proceedings of the 21st International Symposium on Artificial Life and Robotics, pp. 73–77 (2016)
3. Kayama, Y.: Expansion of perception area in cellular automata using recursive algorithm. In: Proceedings of the Fifteenth International Conference on the Simulation and Synthesis of Living Systems, pp. 92–99 (2016)
4. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. ACM SIGGRAPH Comput. Graph. **21**(4), 25–34 (1987)
5. Kayama, Y.: Cellular automata in fractal arrangement. In: Proceedings of the 23rd International Symposium on Artificial Life and Robotics (2018)
6. Gardner, M.: Mathematical games. Sci. Am. **223**, 102–123 (1970)
7. Berlekamp, E.R., Conway, J.H., Guy, R.K.: Winning Ways for Your Mathematical Plays. Academic, New York (1982)
8. Adamatzky, A. (ed.): Game of Life Cellular Automata. Springer, London (2010). https://doi.org/10.1007/978-1-84996-217-9
9. Eppstein, D.: Growth and decay in life-like cellular automata. In: Adamatzky, A. (ed.) Game of Life Cellular Automata, pp. 71–98. Springer, London (2010). https://doi.org/10.1007/978-1-84996-217-9_6
10. Mandelbrot, B.B., Pignoni, R.: The Fractal Geometry of Nature, vol. 173. WH Freeman, New York (1983)
11. Fredkin, E.: An informational process based on reversible universal cellular automata. Phys. D: Nonlinear Phenom. **45**(1–3), 254–270 (1990)
12. Willson, S.J.: Calculating growth rates and moments for additive cellular automata. Discrete Appl. Math. **35**(1), 47–65 (1992)
13. Wang, X., Luan, D.: A novel image encryption algorithm using chaos and reversible cellular automata. Commun. Nonlinear Sci. Numer. Simul. **18**(11), 3075–3085 (2013)