

RESEARCH ARTICLE

Classification of 2D Cellular Automata Nongroup Rules

NORAH H. ALANAZI  **AND ABDULRAOUF KHAN** , (Member, IEEE)

Department of Computer Sciences, College of Computer Sciences and Information Technology, King Faisal University, Al-Ahsa 31982, Saudi Arabia

Corresponding authors: Norah H. Alanazi (219002389@student.kfu.edu.sa) and AbdulRaouf Khan (raoufkh@kfu.edu.sa)

This work was supported by the Deanship of Scientific Research, King Faisal University, under Grant 4732.

ABSTRACT Cellular automata (CA) are self-organizing lattices consisting of cell grids, and the state of each cell can be updated according to the neighboring cell states. The updating rules depend on the neighbors involved in updating a particular cell. In a two-dimensional CA (2DCA) structure, each cell has nine neighbors, including the cell itself. The CA characteristics may change depending on the nature of the operations and the set of neighbors used. Accordingly, researchers have used the 2DCA group rules as a tool to model complex systems in various applications, including artificial intelligence, genomics, computer architecture, graphics, image processing, engineering, and physics. However, nongroup 2DCA rules contain a large number of rules, and the general behavior of all these rules has not yet been studied. In this study, the behavior of 2DCA nongroup rules with null boundary conditions was investigated, and the rules were classified into three categories based on their cycle length and depth. In addition, we applied these classifications to propose a cellular automata-based pseudo-random number generator (PRNG). Statistical tests were performed to validate the suitability of the nongroup CA rules for PRNGs.

INDEX TERMS Cellular automata, cellular automata classification, nongroup two-dimensional CA, random number generation.

I. INTRODUCTION


Cellular automata (CA) are discrete, spatially structured models that evolve over time through a series of discrete steps, where each cell's state is determined by a set of local rules, showcasing emergent and often complex patterns [1]. Cellular refers to the separation of space into separate portions, called cells, and an automaton is a machine that conducts calculations. Examples include machines, computer simulations, and mathematical abstraction. The rules governing automata define how a system changes over time. These rules specify how to determine the state of the CA during the next time step, based on the present state. Cellular automata (CAs) are composed of homogeneous cell arrangements. The cell updates its state during several discrete time steps according to the specific rules and prior states of the immediate neighboring cells.

At the beginning of cellular automaton, it was simply a one-dimensional array, and it was introduced in early 1950 by Von Neumann and Stan Ulam as a model of self-reproducing

biological systems. In 1966, Neumann presented it as a formal model of self-replicating biological systems [2]. Von Neumann's fundamental goal was to apply axiomatic and deductive analysis to the study of complex natural systems.

Recently, CAs have become a well-known area of research and have been widely used by researchers in different domains. Based on the initial state of a CA, it can be either a group CA, meaning that the initial given state is restored, or a nongroup CA, meaning that the initial given state is not restored. CAs are used to model various systems in physics, biology, engineering, and sociology because they provide a simple means for a comprehensive mathematical analysis [3]. In addition, CAs have been proposed for complex systems, such as cryptosystems [4], pseudo-random number generators [5], text compression [6], and image processing [7].

The focus of the research is to develop the theoretical concepts of Cellular automata, so that it can be further used as a tool to simulate various scientific problems and experiments since simple computational rules of 2D non-Group CA can produce complex and unpredictable behavior. There are many natural processes that can be understood through

The associate editor coordinating the review of this manuscript and approving it for publication was Yiming Tang .

computational processes like Chaotic phenomenon, fractals, and neural networks, rather than traditional mathematical relations. And therefore, new computational methods are needed to understand the complexity of the natural world. The structure of the remaining content in this paper is outlined as follows. Section II describes earlier studies contributed by various researchers regarding the characterization and applications of CAs. Section III presents the fundamentals of 2DCAs and the mathematical model for classifying nongroup rules. Section IV presents the results of the classification of the nongroup rules with the proposed 2DCA nongroup algorithm. Section V discusses the applications of 2DCA in the area of pseudo-random number generators (PRNGs) and presents the results of two statistical tests. Section VI. Conclusion of the paper. Section VII. Acknowledgments

II. LITERATURE REVIEW

Over the last 60 years, researchers have investigated CA rules, behaviors, and properties to develop applications in various fields. This section discusses the contributions of researchers to cellular automata in various fields to CA.

Von Neumann described a two-dimensional automaton as an endless array of homogeneous cells in which each cell is linked to its four surrounding neighbors [8]. A two-dimensional cellular automaton (2DCA) structure containing nine cells at each step, including the cell itself, is called the Moore neighborhood [9]. The updating of a cell includes cells in all the directions. The most commonly used neighborhood structures are the five-neighborhood and nine-neighborhood structures. In the nine-neighborhood CA, there are 2^{29} rules.

The “Game of Life” was proposed by Conway [10] in 1960 and is the most well-known CA model. The idea was to study the macroscopic behavior of a population using a simple set of rules [8]. Conway developed a virtual mathematical machine featuring a two-dimensional infinite array of cells, each in either a zero (dead) or a one (alive) state and the update of the cell states is governed by two local rules: survival and birth.

The 1980s was a remarkable time for CA, during which Wolfram conducted many experiments to analyze CA growth patterns [11]. His approach considers cellular automata as models of complex systems, meaning that simple cellular automata rules can lead to very complicated patterns. However, the most significant advantage is that their mathematical simplicity can be extremely helpful in modeling systems. A related phenomenon used in cellular automata evolution is self-organization, rather than using systems of differential equations. Using cellular automata helps reduce the entropy states of random unordered configurations. The structure of the elementary cellular automata rule space was investigated in 1990. The total number of rule tables in the elementary CA rule space is 256 [12]. The authors were concerned with organizing 256 rules in the rule space and the probability of two nearby rules having a likelihood behavior, particularly focusing on the likelihood of neighboring rules exhibiting analogous dynamical behavior. The structure is represented

by the original eight-dimensional rule space and four-dimensional mean-field cluster space. They determined the probabilities of connecting a rule to other intraclass rules using interclass rules.

1992, an elegant tool was introduced by [13] for analyzing various properties of hybrid linear cellular automates and additive cellular automates. The aim of the elegant tool was to achieve a suitable characterization of CAs for two-valued logic circuits. For the sake of simplicity, they specified the type of CA as finite CAs with two states per cell and considered the 3-neighborhood CAs only.

Khan et al. [14] proposed a fixed positional weight CA model in 1997 to understand the behavior of a vast number of rules. It is a mathematical model for representing a 2D uniform CA using a two-dimensional CA with nine neighbors and provides a number called the positional weight for each cell in a grid. These numbers are 1, 2, 4, 8, 16, 32, 64, 128, and 256 in the clockwise direction, and are called primary rules. According to Khan, the 2DCA with nine neighbors has 512 uniform rules, meaning that the same rule applies to all cells simultaneously. An application-specific integrated circuit (ASIC) based on cellular automata for date-authentication was proposed in 2001 [15]. The ASIC is constructed based on the concepts and state transitions of nongroup CAs.

In 2007, researchers employed cellular automata to devise a computational simulation model that elucidated the progression of Ductal Carcinoma In Situ (DCIS), a prevalent type of breast cancer. This model harnessed a simplified genetic regulatory network simulation to govern cell behavior and anticipate the origins of cancer through mutable genes. Their study delved into understanding how somatic mutations contribute to the development of DCIS [16]. Using computational simulations, a genetic model based on cellular automata (GCA) provided related mutable genes to model how somatic mutations lead to DCIS.

Researchers are currently focusing on characterizing 2DCA with multiple single-length cycle attractors to obtain cost-effective solutions in real-life applications. The cycle length provides an interesting classification for understanding the behavior of 2DCA uniform group rules with null boundary conditions based on the characteristic matrix. Khan [17] classified large 2DCA sets into groups based on a T matrix. In 2012, Choudhury et al. [18] focused on 2DCA to model wireless sensor networks (WSNs). They used CA algorithms with radius-of-2 neighborhoods to enhance the coverage and lifetime of a WSN.

In 2019, researchers provided a new CA classification based on the cycle length [19]. They worked on two-dimensional CA group rules that employ uniform XOR rules only and found that the CA cycle lengths rely on the number of columns and/or rows. Their experiment was limited to 65 group rules with null boundary conditions, and they classified 65 2DCA rules as group rules into five classes. However, no study has examined the classification of nongroup 2DCA rules, except a couple of rules in [21]

this study focused on classifying nongroup uniform rules with periodic boundary conditions only.

III. FUNDAMENTALS OF 2DCA

The 2DCA is a parallel processing machine that comprises many cell-holding values (called states) and updating rules. The cells were arranged in an $m \times n$ array (m rows \times n columns). The state of CAs can be represented by an $m \times n$ binary matrix. During their evolution, a cell within a CA modifies its state based on both its present state and the states of its eight neighboring cells. For precision in state updating, a cell follows a set of rules known as the next state rules, often called local rules, whose arguments are the present states of the cell's neighbors [20]. The next state, X , of the $(i, j)^{th}$ cell can be represented mathematically as

$$X_{ij}(t + 1) = f \left[\begin{matrix} X_{(i-1,j-1)}(t), X_{(i-1,j)}(t), X_{(i-1,j+1)}(t), \\ X_{(i,j-1)}(t), X_{(i,j)}(t), X_{(i,j+1)}(t), \\ X_{(i+1,j-1)}(t), X_{(i+1,j)}(t), X_{(i+1,j+1)}(t) \end{matrix} \right] \quad (1)$$

The Boolean function for the nine variables is represented as f . To indicate the 2DCA transition rule, Khan [17] proposed a specific rule convention, as shown in Fig. 1.

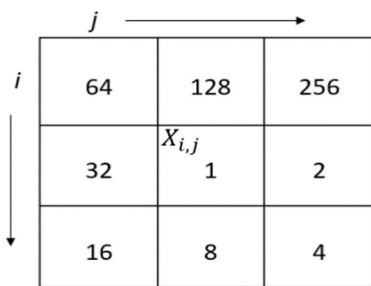


FIGURE 1. Khan model.

If the central cell is considered to be the current cell, then all other cells around it are considered to be its eight nearest neighbors. The numbers in the model signify the rule number associated with the neighbor of the current cell. For instance, if the next state of the current cell depends on its present state, it is called Rule 1. If the next state of the current cell depends on its right neighbor's state, it is called Rule 2. Similarly, if the next state of the current cell depends on the state of its top-right neighbor, it is called rule 256, and this pattern continues accordingly. However, if the next state of the cell is influenced by multiple cells, by convention, the rule number is determined by summing the numerical representations of the respective neighbors. For instance, if the subsequent state relies on the present state of the cell (cell in the center), the right neighbor, and the left neighbor, the rule is computed as $1+2+32$, resulting in 35 rules.

Further, a rule is said to be a group rule if the transition from one state to another returns to the initial state after 'L' steps, called the cycle length. In this case, L can take any value between 1 m and $2^{m \times n}$ for a 2DCA with m rows and n columns. However, if no initial state is returned to, the rule is considered a nongroup rule. Thus, the states are not in a

single cycle. In [21], it was reported that nongroup CA can exhibit three different behavior types:

First type: Irrespective of the number of rows and columns of an $m \times n$ CA, the final state of the CA containing all zero elements is called the graveyard state.

Second type: Irrespective of the number of rows and columns of an $m \times n$ CA, certain states lead to a cycle.

Third type: States lie in a cycle for certain rules with a particular set of rows and columns. This class of nongroup CAs is very exciting because they behave like group rules for a certain number of rows and columns.

This type of 2DCA categorization is in vogue because researchers do not know which rules lead to graveyard states, which leads to a particular cycle. Accordingly, developing CA applications becomes difficult without fully understanding the rule behavior. There are 446 nongroup uniform rules, and this study aims to categorize the behavior of this vast number of rules and classify them into common behaviors or conditions. The classification of these rules is discussed in the next section.

IV. CLASSIFICATION OF NONGROUP RULES

Experiments were conducted to classify 2DCA nongroup rules. The 2DCA structures were randomly selected with seeds having a value of either 0 or 1. The CA sizes, also known as matrices for the experiments, were selected with dimensions $(m \times n) \in \{1, 2, 3, \dots, 256\}$. All nongroup uniform rules [6] were applied to these various 2DCA structures, leading to the separation of nongroup CAs into the three classes described above. The experiments were performed with Anaconda Navigator as a desktop graphical user interface (GUI) using Python.

A. CLASSIFICATION ALGORITHM OF THE NONGROUP RULES

The classification algorithm is essentially a cellular automaton, where each cell in the matrix evolves according to a set of rules. These rules are based on the states of the neighboring cells. To classify the 2DCA nongroup rules, the proposed algorithm determines whether the cycle exists or not and determines the index of the cycle with which matrix it exists. First, the algorithm begins by taking the number of the rows (n) and the columns (m). The values should be integers and in the range of 1 to 256. Next, an empty list for the matrix is initialized to store the generated matrix. A nested loop populates the matrix with random values (0s and 1s). Next, define the cell positions for the eight neighbors surrounding a cell in the matrix. Each input as a nongroup rule is subsequently converted to binary representation. A list is created to store each iteration of the matrix.

The main loop is executed until a termination condition is met. A new matrix is initialized within each iteration to represent the next matrix state. A bitwise XOR operation is performed between the current cell and its relative neighbor, and all the boundaries of the matrix are treated as 0s.

The algorithm checks: if the new matrix is entirely composed of 0s, it declares that the matrix has no cycle, and

Algorithm 1 2DCA Nongroup Rules Algorithm

```

Input: m, n, r
Output: It forms a cycle, or it does not form a cycle
Initialize: matrix0 ← 0s and 1s
Print (matrix0)
cell pos ← [(0, 0), (0, 1), (1, 1), (1, 0), (1, -1), (0, -1),
(-1, -1), (-1, 0), (-1, +1)]
rule ← ⌊log2(r)⌋
← r
rules ← [
while True do
    rule ← ⌊log2(R)⌋, Calculate next rule r
    rules.append(rule)
    R ← R - 2rule
    if R = 0 then
        end if
    end while
all matrix ← [Copy_Matrix(matrix)]
matrix count ← 1
while True do
    new matrix ← Initialize empty matrix
    for k in Range (Length(rules)) do
        Nrule ← rules[k]
        for i in Range(n) do
            for j in Range(m) do
                if is valid cell position (i, j, n, m, cell pos
                [Nrule]) then
                    if k > 0 then
                        new mat[i][j] ← new mat[i][j] ⊕
                        matrix [i + cell pos [Nrule][0]][j
                        + cell pos[Nrule][1]
                    else
                        new mat[i][j] ← matrix [i + cell
                        pos [Nrule][0]][j + cell
                        pos[Nrule][1]
                    end if
                else
                    if k = 0 then
                        new matrix[i][j] ← 0
                    end if
                end if
            end for
        end for
    end for
    flag0 ← Check zero matrix (new matrix)
    if flag0 then
        Print (No cycle)
    end if
    flagC, cycle num ← Check Cycle (all matrix, new
    matrix)
    if flagC then
        Print (Cycle found in matrix No.)
    end if
    all matrix.append(Copy_Matrix(new matrix))
    matrix ← Copy_Matrix (new matrix)
end while

```

ends the loop. If a match is found with the previous matrices, it declares that the matrix has a cycle and the index of the occurrence of the cycle then ends the loop.

If neither all the values are 0s nor the cycle is detected, the new matrix is added to the list of the matrices, replacing the old matrix, and proceeds to the next iteration. The process continues until the termination condition is met.

The 2DCA Nongroup Rules algorithm process is shown in Algorithm 1. The list of symbols used in Algorithm 1 are summarized in Table 1.

TABLE 1. Reference table.

Symbol	Description
m	Number of rows from 1 to 256
n	Number of columns from 1 to 256
r	Rule number (nongroup rules total of 446 rule)
matrix ₀	Initialize Random Matrix of 0s and 1s
cell pos	Cell positions as the relative positions of 8 neighbors
Nrule	Nongroup Rule number
flag ₀	All values in the matrix is zeros
flag _C	The matrix forms a cycle either with initial matrix or after some depths.

B. NONGROUP RULES CLASS 1

Definition: Irrespective of the number of rows and columns (m × n) of an initial 2DCA, the final state of the CA with all zero elements is called the graveyard state, as depicted in Fig.2.

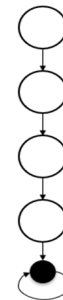


FIGURE 2. Nongroup class one.

Proposition 1: The following rule numbers are class 1 nongroup 2DCA rules with null boundary conditions: 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 36, 40, 44, 48, 52, 56, 60, 64, 66, 72, 80, 88, 96, 104, 112, 120, 128, 130, 132, 134, 144, 160, 176, 192, 194, 208, 224, 240, 256, 258, 260, 262, 264, 266, 268, 270, 288, 320, 322, 352, 384, 386, 388, 390, 416, 448, 450, 480.

For example, Rule 4, in which the next cell state depends on the present state of the right lower corner cell, is mathematically represented as

$$[X_{i,j}]_{t+1} = [X_{i+1,j+1}]_t \tag{2}$$

This result leads to a zero-graveyard state for a matrix size 3 × 3, as shown in Fig.3.

Rule 6 = Rule 2 + Rule 4, in which the next cell state depends on the present states of the right cell and the right lower corner cell, is mathematically represented as

$$[X_{i,j}]_{t+1} = [X_{i,j+1} \oplus X_{i+1,j+1}]_t \tag{3}$$

This result also leads to a zero graveyard for a matrix size 3 × 3, as shown in Fig.4.

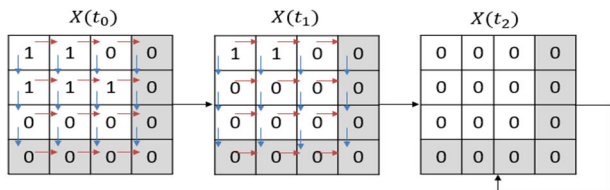


FIGURE 3. Nongroup class one: rule 4 leads to a zero graveyard.

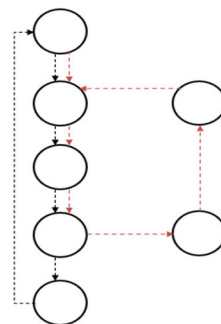


FIGURE 5. Nongroup class two.

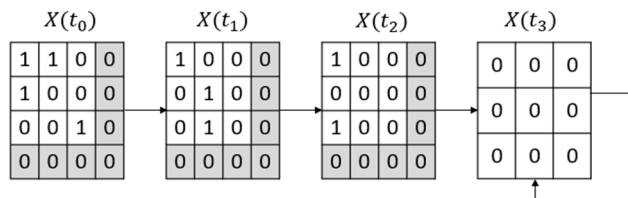


FIGURE 4. Nongroup class one: rule 6 leads to a zero graveyard.

The result leads to the initial matrix for a matrix size of 2×4 , as shown in Fig. 6.

C. NONGROUP RULES CLASS 2

Definition: Given a particular set of rows and columns ($m \times n$) of an initial 2DCA, the states lie on a cycle, as shown in Fig. 5.

Proposition 2: The following rule numbers are class 2 nongroup 2DCA rules with null boundary conditions: 34, 35, 38, 39, 42, 43, 47, 50, 51, 54, 55, 58, 59, 62, 63, 69, 71, 74, 75, 77, 78, 79, 82, 83, 84, 85, 86, 87, 90, 91, 92, 93, 94, 95, 98, 99, 101, 102, 103, 106, 107, 109, 110, 111, 114, 115, 117, 118, 119, 122, 123, 125, 126, 127, 136, 137, 138, 139, 140, 141, 142, 143, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 197, 199, 200, 201, 202, 203, 204, 205, 206, 207, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 273, 275, 276, 277, 279, 281, 283, 284, 285, 287, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 325, 327, 328, 329, 330, 331, 332, 333, 334, 335, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 392, 393, 394, 395, 396, 397, 398, 399, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 452, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 482, 483, 484, 486, 485, 487, 488, 489, 490, 491, 492, 493, 494, 495, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511.

For example, for Rule 98 = Rule 2 + Rule 32 + Rule 64, is mathematically represented as

$$[X_{i,j}]_{t+1} = [X_{i,j+1} \oplus X_{i,j-1} \oplus X_{i-1,j-1}]_t \quad (4)$$

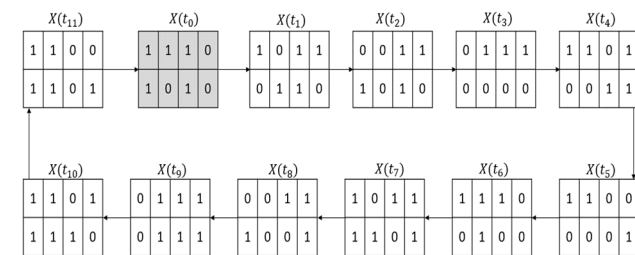


FIGURE 6. Nongroup class two: rule 98 leads to the initial matrix.

Another example is rule 141 = Rule 1 + Rule 4 + Rule 8 + Rule 128, is mathematically represented as

$$[X_{i,j}]_{t+1} = [X_{i,j} \oplus X_{i+1,j+1} \oplus X_{i+1,j} \oplus X_{i-1,j}]_t \quad (5)$$

in which the result leads to the initial matrix for a matrix size 3×6 , as shown in Fig. 7.

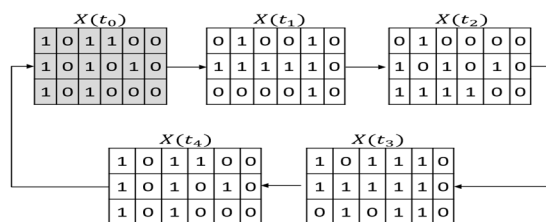


FIGURE 7. Nongroup class two: rule 141 leads to the initial matrix.

D. NONGROUP RULES CLASS 3

Definition: Irrespective of the number of rows and columns ($m \times n$) of 2DCA, some states lead to a cycle. This implies that some states have two predecessors, as shown in Fig. 8.

Proposition 3: The following rule numbers are class 3 nongroup 2DCA rules with null boundary conditions: 68, 70, 76, 100, 108, 116, 124, 196, 198, 272, 274, 278, 280, 282, 286, 304, 324, 326, 336, 368, 400, 432, 453, 454, 465, 466, 496.

For example, for class three is Rule 465 = Rule 1 + Rule 4 + Rule 64 + Rule 128 + Rule 256, mathematically represented as:

$$[X_{i,j}]_{t+1} = [X_{i,j}]_t \oplus [X_{i+1,j+1}]_t \oplus [X_{i-1,j-1}]_t \oplus [X_{i-1,j}]_t \oplus [X_{i-1,j+1}]_t \quad (6)$$

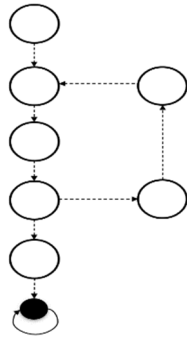


FIGURE 8. Nongroup class three.

When the result leads into zero graveyard for matrix size 2×4 , as shown in Fig.9.

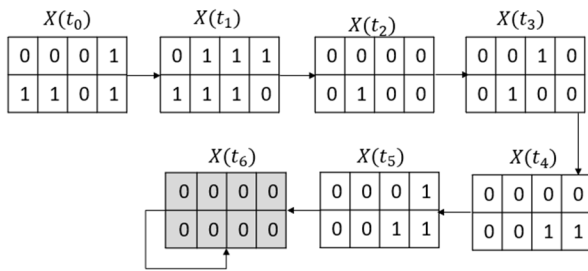


FIGURE 9. Nongroup class three: rule 465 leads to a zero graveyard.

When the result leads to a cycle with a matrix size greater than 0 for a matrix size 3×4 , as shown in Fig.10.

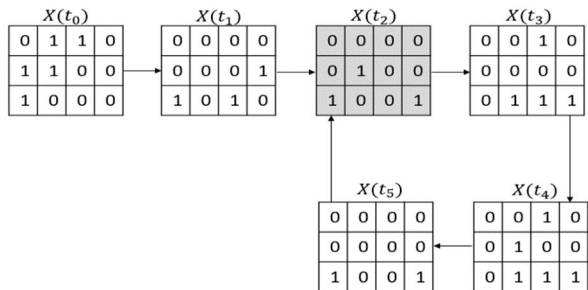


FIGURE 10. Nongroup class three: rule 465 leads to a cycle found with a matrix greater than 0.

The following two lemmas prove class 2 rules namely 34 and 136 (as an example). These rules can either form a cycle or lead to a state with all zero elements. To prove the lemmas T Matrices as mentioned in [20].

Lemma 1: In a 2D CA, with rule 34 = Rule 2 + Rule 32, the next state transition can be represented as

$$[X_{i,j}]_{t+1} = [X_{i,j+1}]_t \oplus [X_{X_{i,j-1}}]_t \tag{7}$$

If the number of columns are even, the CA remains in a cycle, and if number of columns are odd, the CA leads to all zero state.

Proof: The characterization matrix [20] of Rule 34 is given by

$$[T_{34}] = \begin{bmatrix} S_{n \times n} & 0 & 0 & 0 \\ 0 & S_{n \times n} & 0 & \dots & 0 \\ 0 & 0 & S_{n \times n} & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & S_{n \times n} \end{bmatrix} \tag{8}$$

or in Kronecker product form can be written as

$$T_{34} = (I_m) \otimes (S_n) \tag{9}$$

Therefore,

$$\begin{aligned} \text{rank of } T_{34} &= \text{rank of } (I_m) \cdot \text{rank of } (S_n) \\ \text{or } \rho T_{34} &= \rho (I_m) \cdot \rho (S_n) \\ &= m \times \rho(S_n) \end{aligned}$$

$$\begin{aligned} \text{since } \rho (S_n) &= \begin{cases} n & \text{if } n \text{ is even} \\ n-1 & \text{if } n \text{ is odd} \end{cases} \\ \text{or } \rho (T_{34}) &= \begin{cases} mn & \text{if } n \text{ is even} \\ m(n-1) & \text{if } n \text{ is odd} \end{cases} \\ \text{hence, dimension of kernel of } (T_{34}) &= \begin{cases} 0 & \text{if } n \text{ is even} \\ m & \text{if } n \text{ is odd} \end{cases} \end{aligned} \tag{10}$$

This result implies that if n is taken even, the number of predecessors of a state is $2^d = 2^0 = 1$ (where d is the dimension of the kernel), hence lies on a cycle.

Lemma 2: In a 2D CA, with rule 136 = Rule 8+Rule 128, the next state transition can be represented as

$$[X_{i,j}]_{t+1} = [X_{i+1,j}]_t \oplus [X_{i-1,j}]_t \tag{11}$$

If the number of rows are even, the CA remains in a cycle, and if number of rows are odd, the CA leads to all zero state.

Proof: The characterization matrix [20] of Rule 136 is given by

$$[T_{136}] = \begin{bmatrix} 0 & (I)_{n \times n} & 0 & 0 \\ (I)_{n \times n} & 0 & (I)_{n \times n} & \dots & 0 \\ 0 & (I)_{n \times n} & 0 & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix} \tag{12}$$

or in Kronecker product form

$$T_{136} = (S_m) \otimes (I_n) \tag{13}$$

Therefore,

$$\begin{aligned} \text{rank of } T_{136} &= \text{rank of } (S_m) \cdot \text{rank of } (I_n) \\ \text{or } \rho T_{136} &= \rho (S_m) \cdot \rho (I_n) \\ &= \rho (S_m) \times n \end{aligned}$$

$$\text{since } \rho (S_m) = \begin{cases} m & \text{if } m \text{ is even} \\ m-1 & \text{if } m \text{ is odd} \end{cases}$$

$$\text{or } \rho (T_{136}) = \begin{cases} mn & \text{if } m \text{ is even} \\ (m-1)n & \text{if } m \text{ is odd} \end{cases}$$

hence, dimension of kernel of (T_{136})

$$= \begin{cases} 0 & \text{if } m \text{ is even} \\ n & \text{if } m \text{ is odd} \end{cases} \quad (14)$$

This result implies that if m is taken even, the number of predecessors of a state is $2^d = 2^0 = 1$ (where d is the dimension of the kernel), hence lies on a cycle.

V. PSEUDO-RANDOM NUMBER GENERATOR BASED ON TWO-DIMENSIONAL CELLULAR AUTOMATA

This section highlights the application of 2DCAs to pseudo-random number generators (PRNGs). Because 2DCA has more neighboring cells than 1DCA, there are more possible states, and therefore, better randomness properties. PRNGs are algorithms that use statistical methods to generate random numbers [22]. The randomness properties include the prediction of bits in a random number. PRNGs are primarily used in cryptography, such as key generation in encryption and decryption algorithms [23]. Based on the characterization of the nongroup CA, a sequence of random numbers (0s and 1s) could be generated. However, it is obvious from the classification of non-group rules, that Class 1, cannot be used to generate Pseudorandom number, since the final state of the CA will be all zero's and accordingly, the patterns cannot be repeated. The following rules 34, 35, 188, 191, 290, 70, 108, 124, 198, 286, 69, 85, 117, 125, and 197, from Class 2 and Class 3 were used to generate Pseudorandom Numbers with different matrix sizes. The nongroup rules were selected randomly and tested for random number generation.

Several properties have been proposed to validate randomness. Two basic statistical tests were conducted in this study. These tests are commonly used to determine whether a binary sequence has certain features that characterize a truly random sequence.

A. FREQUENCY TEST

A frequency test is a statistical test that can assess the randomness and security of stream ciphers based on 2D cellular automata [24]. The frequency test for 2DCAs involves analyzing the density of 1s or 0s in the matrix over time. The idea is that a random distribution of values should have roughly equal densities of 1s and 0s, whereas non-random behavior (such as predictable repeating patterns) can skew the densities. The statistic used is

$$x1 = \frac{(n_0 - n_1)^2}{n} \quad (15)$$

Let n_0 and n_1 denote the number of 1s and 0s in sequence s . This statistic follows χ^2 distribution with one degree of freedom [24].

B. POKER TEST

The poker test is a statistical test used to analyze the randomness of a 2DCA [24]. In this test, a grid of cells was randomly initialized with two possible states: 0s and 1s. The automaton is then executed for a specified number of steps, and the state of each cell is observed at the end of each step. The poker test involves obtaining a sample of cells from the

grid and converting their states into binary numbers. These binary numbers are then grouped into sets. Each set was compared with a reference table of possible poker hands, and the frequency of each poker hand was recorded. This process was repeated for multiple samples, and the results were tabulated.

TABLE 2. Frequency and poker test results.

Matrix size	Rule	Frequency Test	Poker Test	Results
4 × 4	34	0.75	0.878	PASS
4 × 4	35	0.75	0.88	PASS
4 × 4	188	0.831	0.863	PASS
4 × 4	191	0.4666	0.878	PASS
4 × 4	290	0.4	0.7692	PASS
10 × 10	70	0.89	0.63	PASS
10 × 10	108	0.83	0.63	PASS
10 × 10	124	0.99	0.63	PASS
10 × 10	198	0.97	0.63	PASS
10 × 10	286	0.89	0.63	PASS
16 × 8	69	0.35	0.77	PASS
16 × 8	85	0.68	0.71	PASS
16 × 8	117	0.95	0.74	PASS
16 × 8	125	0.95	0.73	PASS
16 × 8	197	0.99	0.71	PASS

The poker test aims to determine whether the distribution of poker hands in the sample is statistically consistent with a random distribution. If so, the automaton is said to produce “random-like” behavior. The poker test counts the number of times each potential subsequence of length m occurs approximately the same number of times in sequence s such that $\lfloor n/m \rfloor \geq 5 \cdot (2^m)$. Defining $k = \lfloor n/m \rfloor$, sequence s is divided into k non-overlapping parts of length m . Let n_i be the number of occurrences of the i^{th} type of sequence of length m , where $1 \leq i \leq 2^m$. The statistic used is

$$x3 = \frac{2^m}{k} \left(\sum_{i=1}^{2^m} n_i^2 \right) - k \quad (16)$$

and roughly follows a χ^2 distribution with $(2^m - 1)$ degrees of freedom [25].

Table 2 presents the results of both tests against various 2DCA nongroup rules. A sequence of random numbers (0s and 1s) was generated with different matrix sizes using rules 34, 35, 188, 191, 290, 70, 108, 124, 198, 286, 69, 85, 117, 125, and 197.

VI. CONCLUSION

This paper presents experimental results investigating two-dimensional cellular automata (2DCA) nongroup rules

leading to the discovery of new usages and developments in CA. In addition, 2DCA nongroup rules with null boundary conditions have been classified into three types with distinct behaviors. This classification is expected to be important for understanding and developing new 2DCA application areas for nongroup rules. This paper also demonstrates and validates the applicability of nongroup 2DCAs to pseudo-random number generators (PRNGs).

Throughout our studies and experiments of 2DCA nongroup rules we found that it generates behaviors that are difficult to interpret which limit their effectiveness in pattern recognition applications compared to group CA rules. Also, some of 2DCA non-group rules may lead to computationally complex behavior, making it challenging to analyze or simulate their dynamics efficiently.

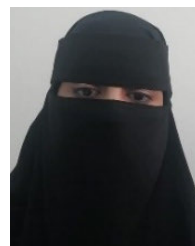
The future works will include developing a general mathematical framework to calculate the cycle lengths of all the non-group CA rules and explore new application areas in the fields of artificial intelligence and reinforcement learning.

ACKNOWLEDGMENT

The authors wish to express their gratitude and appreciation to the Deanship of Scientific Research, King Faisal University, for their valuable support and encouragement throughout this study.

REFERENCES

- [1] S. Wolfram, *A New Kind of Science*, vol. 5. Champaign, IL, USA: Wolfram Media, 2002, p. 130.
- [2] J. V. Neumann, *Theory of Self-Reproducing Automata*, A. W. Burks, Ed. Urbana, IL, USA: Univ. of Illinois Press, 1966.
- [3] S. Das, S. Mukherjee, N. Naskar, and B. K. Sikdar, "Characterization of single cycle CA and its application in pattern classification," *Electron. Notes Theor. Comput. Sci.*, vol. 252, pp. 181–203, Oct. 2009.
- [4] E. Corona-Bermúdez, J. C. Chimal-Eguía, and G. Téllez-Castillo, "Cryptographic services based on elementary and chaotic cellular automata," *Electronics*, vol. 11, no. 4, p. 613, Feb. 2022, doi: [10.3390/electronics11040613](https://doi.org/10.3390/electronics11040613).
- [5] A. Levina, D. Mukhamedjanov, D. Bogaevskiy, P. Lyakhov, M. Valueva, and D. Kaplun, "High performance parallel pseudorandom number generator on cellular automata," *Symmetry*, vol. 14, no. 9, p. 1869, Sep. 2022, doi: [10.3390/sym14091869](https://doi.org/10.3390/sym14091869).
- [6] A. R. Khan, P. P. Choudhury, K. Dihidar, and R. Verma, "Text compression using two-dimensional cellular automata," *Comput. Math. Appl.*, vol. 37, no. 6, pp. 115–127, Mar. 1999, doi: [10.1016/S0898-1221\(99\)00080-2](https://doi.org/10.1016/S0898-1221(99)00080-2).
- [7] S. Roy, M. Shrivastava, U. Rawat, C. V. Pandey, and S. K. Nayak, "IESCA: An efficient image encryption scheme using 2-D cellular automata," *J. Inf. Secur. Appl.*, vol. 61, Sep. 2021, Art. no. 102919, doi: [10.1016/j.jisa.2021.102919](https://doi.org/10.1016/j.jisa.2021.102919).
- [8] P. Sarkar, "A brief history of cellular automata," *ACM Comput. Surv.*, vol. 32, no. 1, pp. 80–107, Mar. 2000, doi: [10.1145/349194.349202](https://doi.org/10.1145/349194.349202).
- [9] J. Santoso, O. S. Santoso, and B. R. Trilaksono, "Matrix characteristics for two dimensional nongroup cellular automata," in *Proc. Int. Conf. Electr. Eng. Informat.*, Jul. 2011, pp. 1–4, doi: [10.1109/ICEEI.2011.6021567](https://doi.org/10.1109/ICEEI.2011.6021567).
- [10] G. J. Martínez, A. Adamatzky, K. Morita, and M. Margenstern, "Computation with competing patterns in life-like automaton," in *Game of Life Cellular Automata*. London, U.K.: Springer, 2010, pp. 547–572, doi: [10.1109/HPCS.2010.5547075](https://doi.org/10.1109/HPCS.2010.5547075).
- [11] S. Wolfram, "Statistical mechanics of cellular automata," *Rev. Mod. Phys.*, vol. 55, no. 3, pp. 601–644, Jul. 1983.
- [12] W. Li and N. Packard, "The structure of the elementary cellular automata rule space," *Complex Syst.*, vol. 4, no. 3, pp. 281–297, 1990.
- [13] A. K. Das, A. Sanyal, and P. Palchaudhuri, "On characterization of cellular automata with matrix algebra," *Inf. Sci.*, vol. 61, no. 3, pp. 251–277, Jun. 1992, doi: [10.1016/0020-0255\(92\)90053-b](https://doi.org/10.1016/0020-0255(92)90053-b).
- [14] A. R. Khan, P. P. Choudhury, K. Dihidar, S. Mitra, and P. Sarkar, "VLSI architecture of a cellular automata machine," *Comput. Math. Appl.*, vol. 33, no. 5, pp. 79–94, Mar. 1997, doi: [10.1016/S0898-1221\(97\)00021-7](https://doi.org/10.1016/S0898-1221(97)00021-7).
- [15] P. Dasgupta, S. Chattopadhyay, and I. Sengupta, "Theory and application of non-group cellular automata for message authentication," *J. Syst. Archit.*, vol. 47, no. 5, pp. 383–404, May 2001, doi: [10.1016/S1383-7621\(00\)00058-8](https://doi.org/10.1016/S1383-7621(00)00058-8).
- [16] A. Bankhead and R. B. Heckendorn, "Using evolvable genetic cellular automata to model breast cancer," *Genetic Program. Evolvable Mach.*, vol. 8, no. 4, pp. 381–393, Dec. 2007, doi: [10.1007/s10710-007-9042-x](https://doi.org/10.1007/s10710-007-9042-x).
- [17] A. R. Khan, "Classification of 2D cellular automata uniform group rules," *Eur. J. Sci. Res.*, vol. 64, no. 1, pp. 51–57, 2011.
- [18] S. Choudhury, K. Salomaa, and S. G. Akl, "A cellular automaton model for wireless sensor networks," *J. Cell. Automata*, vol. 7, no. 3, pp. 223–241, Aug. 2011, doi: [10.2316/P.2011.735-089](https://doi.org/10.2316/P.2011.735-089).
- [19] A. Alrumaih, "On cycle lengths of two-dimensional linear uniform cellular automata group rule," M.S. thesis, Dept. Comput. Sci., King Faisal Univ., Al Hofuf, Saudi Arabia, 2019.
- [20] A. R. Khan, "On two dimensional cellular automata and its VLSI applications," *Int. J. Electr. Comput. Sci.*, vol. 10, no. 6, pp. 124–129, 2010.
- [21] A. R. Khan, F. Al_Humaidan, and Mansoor-uz-Zafar, "Invertibility condition of few non group 2D cellular automata rules," *Eur. J. Sci. Res.*, vol. 67, pp. 240–247, Mar. 2012.
- [22] L. Pasqualini and M. Parton, "Pseudo random number generation: A reinforcement learning approach," *Proc. Comput. Sci.*, vol. 170, pp. 1122–1127, Jan. 2020, doi: [10.1016/j.procs.2020.03.057](https://doi.org/10.1016/j.procs.2020.03.057).
- [23] I. G. A. Poornima, B. Paramasivan, K. M. Pitchai, and M. Bhuvanawari, "A survey on cellular automata with the application in pseudo random number generation," *Netw. Inf.*, vol. 5, no. 2, pp. 12–22, 2017.
- [24] S. M. Hosseini, H. Karimi, and M. V. Jahan, "From complexity to random behaviors; Generate random numbers by confusion in cellular automata state's," in *Proc. Int. Conf. Sci. Comput. (CSC)*, 2011, p. 1.
- [25] A. S. Mohammed, "Applied poker test for general digital sequences," *IOSR J. Math.*, vol. 12, no. 1, pp. 17–23, 2016.



NORAH H. ALANAZI received the B.S. degree in computer science from King Faisal University, Saudi Arabia, in 2017, where she is currently pursuing the M.S. degree in computer sciences.

From 2019 to 2020, she was a Software Developer with Enseyab Information Technology, Saudi Arabia. In 2021, she joined Sendan International Company, Saudi Arabia, as a Software Developer. Her research interests include cellular automata and cryptography.

Ms. Norah maintained an academic excellence rate throughout her M.S. study program from 2019–2022.



ABDULRAOUF KHAN (Member, IEEE) received the M.Sc. and Ph.D. degrees in electronics and computer sciences from the University of Kashmir, Srinagar, India, in 1987 and 1999, respectively.

From 1988 to 2001, he was a Senior Lecturer with the Department of Electronics, University of Kashmir. In 2001, he joined the Al-Zaytoonah University of Jordan, Amman, Jordan, as an Assistant Professor with the Department of Computer Science. Since 2005, he has been an Associate Professor with the Department of Computer Science, King Faisal University, Saudi Arabia. He has published more than 50 articles in the reputed International Journals and Conferences. He has investigated several research projects funded by the governments of Saudi Arabia and Malaysia. His research interests include the theory and applications of cellular automata, computer architecture, computer security, and image processing.

Dr. Khan is also a member of several other professional societies.

• • •