Available online at www.sciencedirect.com
ScienceDirect
Mathematics and Computers in Simulation 168 (2020) 111–121
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Original articles

Application of the graph cellular automaton in generating languages

B. Praba*, R. Saranya

SSN College of Engineering, Chennai, Tamil Nadu, India

Received 13 November 2018; received in revised form 9 April 2019; accepted 16 July 2019
Available online 17 August 2019

Abstract

The purpose of this research is to strengthen the core of Cellular Automaton using the concept of Graph theory and Automata theory. In this paper we introduce a new approach of defining the Graph Cellular Automaton (GCA), (V, I, δ) consisting of set of vertices and input symbols along with the time evolution function δ. Using the Basic Linear Rules, we define the kth generation GCA. This method enables us to capture the complete characterization of the future generations. When the defined GCA is taken as a graph corresponding to a given finite automaton, then the language generated by the kth generation GCA is predicted corresponding to the fundamental basic linear rules. This enables us to predict the language of the kth generation automaton using all the 512 rules. All explained concepts are illustrated with suitable examples.

Keywords: Cellular automaton; Complete graph; Graph cellular automaton; Basic linear rules; Finite automaton

1. Introduction

Throughout the 20th century, researchers' views of applications on CellularAutomaton changed vigorously [7,15]. The works on Graph Cellular Automaton development were started in the year 2002. David O'Sullivan [10] introduced a dynamic structural model and briefly discussed the relationship between other models and Graph Cellular Automaton. Many researchers made an attempt to give an application of CA in land-use [6]. Niandry, Fangwang used CA model to overcome the sensitivity of the neighbourhood [8]. Daniel, Suzana [13] discussed the model by which the stakeholders can evaluate to make subdivision designs to increase the grow rates of plants and to satisfy the buyer's expectation. Some of the researchers used GCA in Traffic simulation [2,5,16], Edge detection [4,12,15] and wireless sensor networks [14]. Ada Yuen and Robin Key [16] explained the application of Cellular Automaton briefly. They explained the traffic modelling by dividing it into three groups namely traffic flow theory, car flow theory, the particle hopping traffic model and also they explained structural design and musical composition. Birendra Kumar, Nayak [1] studied the Graph theoretic properties of Cellular Automaton and simplified all the Basic Linear Rules in comparison with other ways of its study. In graph theory, [3,4] Jia-Bao Lio, Xiang-feng-Pan characterized the graph which is having the minimum Kirchhoff index value among graphs with fixed number n of vertices and also their result implies that the Laplacian-energy-like per vertex of other lattices is independent of the boundary conditions. Praba defined the Rough Finite State Automaton and studies its properties of the Automaton [11].

* Corresponding author.
E-mail addresses: prabab@ssn.edu.in (B. Praba), saranyar@ssn.edu.in (R. Saranya).

https://doi.org/10.1016/j.matcom.2019.07.010

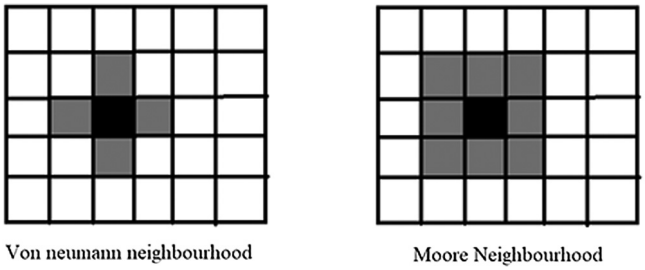

Fig. 1. The Von Neumann neighbourhood and The Moore Neighbourhood.

The researchers studied Cellular Automaton's future prediction using Basic Linear Rules. In this study they have assumed that each grid will be in any one of the states on or off. In this work, we extended this idea and we consider the grids will be in any one of the states from a finite set I. The basic linear rules are used in shifting the state of the grid. The main advantage of this approach is to view a 2 dimensional grid in which each cell will be in any one of the states from a finite set I as a GCA. Hence the language generated by this GCA can be predicted and the appropriate use of the basic linear rules repeatedly will yield the next generation Automaton. Also we have predicted the language generated by the kth generation GCA. In [9] a kind of Cellular Automaton Called Partitioned Cellular Automaton was defined to generate the array languages. But they have not used basic linear rules to generate the arrays. Another advantage of our work is that once the languages using the basic linear rules are predicted then the language generated by the rest of the rules out of 512 rules can also be predicted. This research article methodizes with the aim of developing the Next Generation GCA with the special focus on Language of the Automaton. This paper helps in modelling the next generation language with the help of GCA and the machine's life time can also be calculated by the discussed concept of GCA.

Section 1 deals with the basic preliminaries which is required to study the rest of the paper, the definition of GCA is defined and we characterize the Basic Linear Rule matrix in Section 2, In Section 3 kth Generation of the Graph Cellular Automaton is defined and the Generalization of the Next Generation Automaton is illustrated by theorems with suitable examples and Sections 4 and 5 describe the language of the corresponding automaton, generalization of some of the rules and examples followed by a conclusion.

2. Preliminaries

In this section some of the basic definitions are described.

2.1. Two dimensional cellular automaton

Two dimensional Cellular Automaton is set of cells known as grids black and white in colour in the way of chess board but it will not follow the rule of chess board it has its own rules to generate the CA's. The colour indicates the life of each cell. Based on the neighbourhood cell, a cell's life will be assigned at time $t+1$. Typically there were two neighbourhoods, Von Neumann neighbourhood and Moore neighbourhood (Fig. 1). Based on these two neighbourhoods selection the set of rules is assigned to generate the periodic life of the Cellular Automaton. Conway's Game of life is an interesting and very important factor to generate the next generation of the Cellular Automaton. (1) Alive — Any live cell with two or three neighbouring live cells then it will be alive in the next generation. (2) Dead — (i) Any live cell with more than three neighbouring live cells will die in the next generation called overpopulation. (ii) Any live cell with fewer than two neighbouring live cells will die in the next generation called loneliness. (3) Birth — Any dead cell with exactly three neighbouring live cells will become live cell in the next generation called reproduction.

2.2. Characterization of 2 dimensional CA

The study of CA is made easier through its matrix representation. If a grid is in of dimension $m \times n$ then the rule matrix of the grid is $mn \times mn$. The $m \times n$ matrix is also known as a problem matrix or an information matrix. Consider an information matrix of order 2×2 then the rule matrix will be of order of 4×4. The matrix corresponding to Rule n is denoted by M_n.

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad M_2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad M_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$M_8 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad M_{16} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Each row of the Rule matrix will explain the dependency of each element in the problem matrix. For example 1st row of the Rule matrix will represent the behaviour of the 1st element in the problem matrix. 2nd row of the Rule matrix will represent the dependency of the 2nd element in the problem matrix and so on.

2.3. Basic linear rules

Basically for all binary information matrix ($m \times n$), there exists a rule matrix ($mn \times mn$), this is realized as matrix ($adjacency\ matrix$), then there exists a graph with vertices, edges.

Rule 1, Rule 2, Rule 4, Rule 8 and Rule 16 are the fundamental basic rules. All the other rules can be obtained by using these 5 rules.

Rule 32, Rule 64, Rule 128 and Rule 256 are also known as fundamental basic rules which were obtained with the transpose operation of 1, 2, 4, 8 and 16 respectively.

2.4. Graph theory

Definition 2.4.1 (*Graph*)**.** A graph is an ordered pair $G = (V, E)$
V — The vertex set consisting of the nodes of the graph can also be written as $V(G)$.
E — The edge set consisting of the connections between the vertices also be written as $E(G)$.

Definition 2.4.2 (*Adjacency Matrix*)**.** An Adjacency matrix is a square matrix which represents a graph. In a matrix the place of the element indicates the connections of the graph. For every simple graph there exists an $(0-1)$ adjacency matrix. For a weighted graph the corresponding adjacency matrix will have the weights as its entries.

In this paper we use the matrix representation to define the GCA. Using the Basic Linear Rules we predict the Next generation Graph Cellular Automaton.

Definition 2.4.3 (*Automaton*)**.** An Automaton is represented by 5 tuples $\left(Q, \sum, \delta, q_0, F\right)$, where Q is a non empty set of States $\sum$ is a finite set of symbols, called the alphabet of the automaton δ is the transition function q_0 is the initial state from where any input is processed ($q_0 \in Q$) and F is the set of final state/states of Q ($F \subseteq Q$).

Definition 2.4.4 (*Language*)**.** $L \subseteq \sum^*$ recognized by an automaton is the set of all the words that are accepted by the Automaton.

2.5. Generalization of basic linear rule matrix using graph theory concept

Rule 1 For every $m \times n$ problem matrix, the graph of Rule 1 has mn self loops in it which means that each entry of the problem matrix will depend on itself.

Rule 2 The graph of Rule 2, connects the vertex v_i to v_{i+1} except $i = kn$, $k = 1, 2, \ldots$.

Rule 4 In a graph of rule 4, the vertex v_i connects with the vertex v_{n+i+1} for $i = 1, 2, \ldots, mn-n-1$ except $i = n$ and $mn - (n-1)$.

Rule 8 v_i connects to v_{n+i} in the graph of rule 8.

Rule 16 The graph of rule 16, connects (v_i, v_{n+i-1}) the vertices v_1 and v_{mn} are isolated.

These 5 rules are called as the Basic Linear Rules.

3. *K*th generation graph cellular automaton using basic linear rules

Definition 3.1. Graph Cellular Automaton is a 2 dimensional CA, $G = \{V, I, \delta\}$ consisting of a set of vertices V, I is a nonempty set of input symbols and δ is a time evolution function $\delta : V \times I \to V$. Using those 5 basic rule matrices in Cellular Automaton, the Next Generation Automaton is obtained by multiplying the rule matrix to the row major order of the information matrix. Hence we have the following definition.

Definition 3.2. Kth Generation Graph Cellular Automaton is a 2-dimensional CA defined as an ordered pair $G_k = (G_{k-1}, R)$ and $k \geq 1$, where G_{k-1} is the GCA corresponding to the $(k-1)$th Generation.

R is the rule matrix.

Note that kth generation Automaton can be viewed as a weighted graph for $k \geq 0$.

$A(G)_k = R \times A(G)_{k-1}$, where $A(G)_k$ is the adjacency matrix corresponding to GCA G_k.

Remark

(1) In real time problems the dependency of each cell of a Graph will vary with respect to the problem under consideration. Accordingly one had to use the appropriate rule matrix corresponding to the given Graph at each generation.

3.1. A generalization of the next generation GCA

In the following discussion $A(G)_k$ represents the adjacency matrix corresponding to the kth generation GCA of order $n \times n$ then the following lemma has been proved. This result explains the nature of the future generation GCA of Rules 1, 2, 4, 8 and 16.

Lemma 1. *The generation of the GCA remains the same on applying Rule 1.*

Proof. As Rule 1 is an identity matrix it is obvious that the generation of the GCA remains the same.

Lemma 2. *The kth generation of the GCA using Rule 2 will vanish.*

Proof. Using Rule 2 we note that the edge (V_i, V_j) transforms to (V_i, V_{j-1}) and also that in a $m \times n$ matrix, the nth column of the matrix will be zero in the first generation itself, from this one can conclude that the complete $m \times n$ matrix will vanish in the kth generation. Also, it is important to note that, the final state cannot be reached in the first generation itself because of the 0's in the nth column.

Example 1. The following problem matrix is the corresponding adjacency matrix for the above graph (Fig. 2) $A(G_0) = \begin{bmatrix} 0 & a & 0 \\ 0 & b & a \\ 0 & b & 0 \end{bmatrix}$. By the definition $M_2 \times A(G_0) = A(G_1)$.

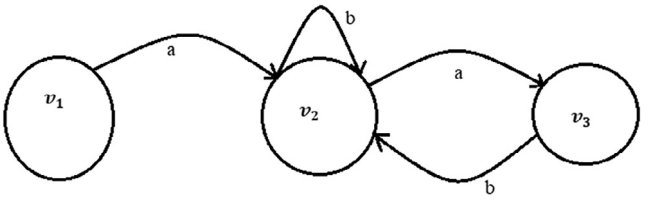

Fig. 2. GCA G_0 with $I = \{a, b\}$.

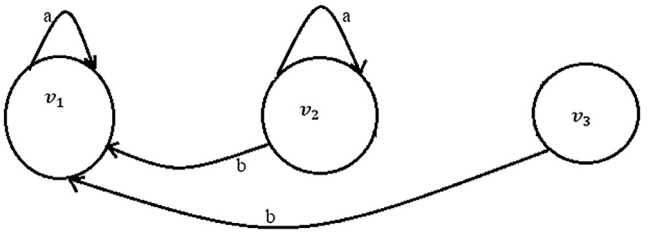

Fig. 3. First Generation GCA G_1 of G_0 on applying Rule 2.

As the problem matrix is of order 3×3, the corresponding rule matrix is of order 9×9.
Using Rule 2, we have a 9×9 matrix,

$$\begin{bmatrix} 0&1&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0 \end{bmatrix} \begin{bmatrix} 0\\ a\\ 0\\ 0\\ b\\ a\\ 0\\ b\\ 0 \end{bmatrix} = \begin{bmatrix} a\\ 0\\ 0\\ b\\ a\\ 0\\ b\\ 0\\ 0 \end{bmatrix}$$

Hence $A(G_1) = \begin{bmatrix} a&0&0\\ b&a&0\\ b&0&0 \end{bmatrix}$. The First Generation GCA G_1 is shown below (Fig. 3)

Similarly $M_2 \times A(G_1) = A(G_2)$.

$$\begin{bmatrix} 0&1&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0 \end{bmatrix} \begin{bmatrix} a\\ 0\\ 0\\ b\\ a\\ 0\\ b\\ 0\\ 0 \end{bmatrix} = \begin{bmatrix} 0\\ 0\\ 0\\ a\\ 0\\ 0\\ 0\\ 0\\ 0 \end{bmatrix}$$

$$A(G_2) = \begin{bmatrix} 0&0&0\\ a&0&0\\ 0&0&0 \end{bmatrix}, A(G_3) = \begin{bmatrix} 0&0&0\\ 0&0&0\\ 0&0&0 \end{bmatrix}$$

By the example it is clear that the information matrix is of order 3×3, so that the 3rd Generation of the Graph Cellular Automaton G_3 does not contain any transition.

Lemma 3. *The kth generation of the GCA using Rule 4 will vanish*

Proof. On applying Rule 4, in the next generation the edge (V_i, V_j) is shifted to (V_{i-1}, V_{j-1}). There will be no incoming edges and outgoing edges for nth state after applying Rule 4 to the matrix. By this one can conclude the kth generation of the GCA will vanish.

Example 2. Consider the Graph G (Fig. 2), then the information matrix of the corresponding graph is

$$A(G_0) = \begin{bmatrix} 0 & a & 0 \\ 0 & b & a \\ 0 & b & 0 \end{bmatrix}$$

To obtain $A(G_1)$,
$M_4 \times A(G_0) = A(G_1)$.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ 0 \\ 0 \\ b \\ a \\ 0 \\ b \\ 0 \end{bmatrix} = \begin{bmatrix} b \\ a \\ 0 \\ b \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Hence $A(G_1) = \begin{bmatrix} b & a & 0 \\ b & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$. Proceeding in a similar manner $A(G_3)$ will be zero matrix.

Lemma 4. *In the next generation GCA using Rule 8, the nth row will be all zero.*

Proof. In the generation of the GCA using Rule 8 the edge (V_i, V_j) is transformed to (V_{i-1}, V_j). When we note that the Rule matrix of Rule 8 after the $m.n$ row, all the entries will be zero. This indicates that there will be no outgoing edge from the final state. Hence the life of the GCA generations will be completely dissolved after the mth generation.

Example 3. Consider the same information matrix

$$A(G_0) = \begin{bmatrix} 0 & a & 0 \\ 0 & b & a \\ 0 & b & 0 \end{bmatrix}$$

To obtain the first generation of the Graph using Rule 8. $M_8 \times A(G_0) = A(G_1)$.

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ 0 \\ 0 \\ b \\ a \\ 0 \\ b \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ b \\ a \\ 0 \\ b \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Hence $A(G_1) = \begin{bmatrix} 0 & b & a \\ 0 & b & 0 \\ 0 & 0 & 0 \end{bmatrix}$.

Lemma 5. *In the next generation of GCA using Rule 16*
(*i*) *The edge* (V_i, V_j) *is shifted to* (V_{i-1}, V_{j+1}) *if* $i \geq j$.
(*ii*) *The edge moves on to* (V_i, V_{j-2}) *if* $i < j$.

Proof. If the information matrix $A(G_0)$ is of order $n \times n$ then M_{16} is of order $n^2 \times n^2$ in which $(2, n+1^{th})$ *entry* $(3, n+2^{th}), \ldots$ will be equal to one and the rest of the entries are zero. Hence $M_{16} \times A(G_{k-1}) = A(G_k)$ implies that if the (i, j)th entry $i \geq j$ in $A(G_{k-1})$ is one then $(i-1, j+1)$th entry will be 1 in $A(G_k)$ and if (i, j)th entry is 1 for $i < j$ in $A(G_{k-1})$ then $(i, j-2)$th entry will be 1 in $A(G_k)$

Remark On use of Rule 16, there will be no loop in the 1st vertex.

Example 4. Consider the same information matrix

$$A(G_0) = \begin{bmatrix} 0 & a & 0 \\ 0 & b & a \\ 0 & b & 0 \end{bmatrix}$$

To obtain the first generation of the GCA (G_1) using Rule 16. $M_{16} \times A(G_0) = A(G_1)$.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ 0 \\ 0 \\ b \\ a \\ 0 \\ b \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b \\ a \\ 0 \\ b \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Hence $A(G_1) = \begin{bmatrix} 0 & 0 & b \\ a & 0 & b \\ 0 & 0 & 0 \end{bmatrix}$.

In the following section, we consider a FA which can also be viewed as a GCA. In real time problems the dependency of the state will vary in each generation which will in turn affect the language generated by the next generation automaton.

4. Language generated by the GCA using the basic linear rules

In the following discussion, we consider a finite automaton $M = (Q, \sum, \delta, q_0, F)$, note that this can be viewed as a 2 dimensional initial information graph GCA (G_0) denoted by $M(G) = (V, I, \delta)$, where $V = Q$ is the set of vertices, $I = \sum$ the set of input symbols and δ is as in M. In real time problems when the transition between the states depends on its neighbourhood, δ becomes time evolution function. Hence we will be able to define the next generation automaton by using the appropriate rule matrix. If we assume that all possible transitions exist in G_0, that means the underlying graph is complete graph then the language generated by the next generation Automaton is defined as $L_J(M(G_i))$ where $J = 2, 4, 8$ and 16 , $i = 1, 2, 3, \ldots, n$ here J denoted the rule and i denotes the generation of the GCA. As our assumption is that all possible transitions exist in G_0 the language generated by any other Automaton M' with the same input symbol I denoted $L_J(M'(G_i)) \subseteq L_J(M(G_i))$. If the underlying graph G_0 is a complete graph with 'n' vertices, then the language generated by the first generation automaton using Rule 2 and Rule 4 will be $L_2(M(G_1)) = m(x_1m)^*(x_2m)^* \cdots (x_{n-2}m)^*$ and $L_4(M(G_1)) = m(x_1m)^*(x_2m)^* \cdots (x_{n-2}m)^*$ respectively.
where,
m — a string with one (*or*) more Input Symbols.
$x_i's$ — a string with input symbol/symbols.

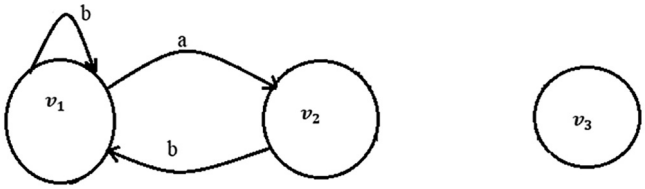

Fig. 4. First Generation GCA G_1 of G_0 Fig. 2 on applying Rule 4.

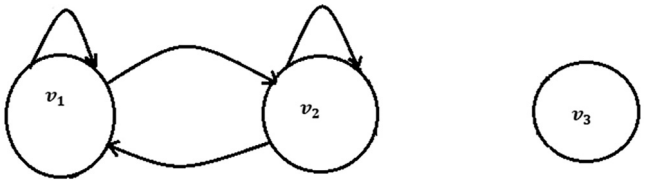

Fig. 5. First Generation GCA of the Complete Graph with 3 vertices using Rule 2.

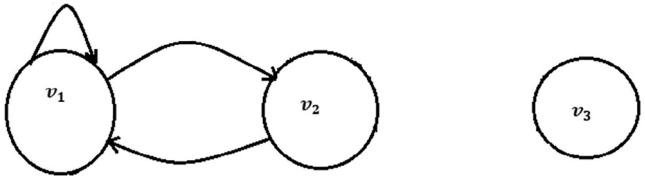

Fig. 6. First Generation GCA of the Complete Graph with 3 vertices using Rule 4.

Example 5. In the following example G_0 is taken as the complete graph with 3 vertices and (Figs. 5 and 6) represents the first generation GCA G_1 using Rule 2 and Rule 4 respectively then the languages are

$$L_2\left(M\left(G_1\right)\right) = m\left(x_1 m\right)^* \text{ and } L_4\left(M\left(G_1\right)\right) = m\left(x_1 m\right)^*$$

Where,
m — a string with one (or) more input Symbols.
x_1 — a string with one input symbol/symbols.
note that for Rule 2 and Rule 4 as already defined $(n-1)$th state is the final state.

Example 6. For the GCA defined in Figs. 3 and 4, assuming v_2 as a final state the language generated by the GCA using Rule 2 and Rule 4 are $L_2\left(M\left(G_1\right)\right) = a^*\lambda^*\left(b^n\left(a^*\lambda a^*\right)^n\right)$ and $L_4\left(M\left(G_1\right)\right) = b^*a\left(b^n\left(b^*a\right)^n\right)^*$ respectively.

In the same manner if we assume that G_0 is the complete graph with 'n' vertices, then the language generated by the first generation automaton using Rule 8 and Rule 16 will be $L_8\left(M\left(G_1\right)\right) = m\left(x_1 m\right)^*\left(x_2 m\right)^* \cdots \left(x_{n-2} m\right)^*$ and $L_4\left(M\left(G_1\right)\right) = m\left(x_1 m\right)^*\left(x_2 m\right)^* \cdots \left(x_{n-2} m\right)^*$ respectively.
where,
m — a string with one (or) more Input Symbols.
$x_i's$ — a string with input symbol/symbols.

Example 7. By assuming G_0 is the complete graph with 3 vertices, then the language generated by the first generation GCA using Rule 8 (Fig. 7) and Rule 16 (Fig. 8) are $L_8\left(M\left(G_1\right)\right) = x_1\left[m\left(x_2 m\right)^* x_3\right]^*$
$L_{16}\left(M\left(G_1\right)\right) = x_1\left[\left\{m\left(x_2 m\right)^* x_3\right\}\left\{x_4\left(m\left\{x_2 m\right\}^* x_5\right)\right\}\right]^*$
where, m — String with one or more Input Symbol.
x_1, x_2, x_3, x_4, x_5 — String with only one Input Symbol.

Example 8. The language generated by the first Generation GCA using Rule 8 and Rule 16 which is defined in Examples 3 and 4 is $L_8\left(M\left(G_1\right)\right) = a$ and $L_{16}\left(M\left(G_1\right)\right) = a^*bb^*$ by considering v_2 as a final state.

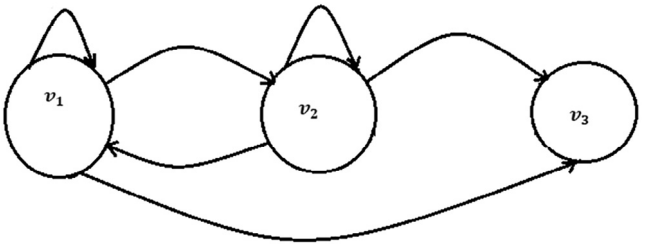

Fig. 7. First Generation GCA of the complete graph with 3 vertices using Rule 8.

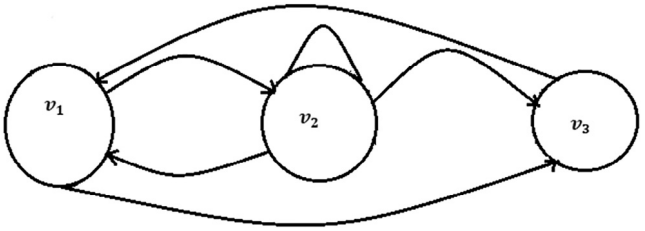

Fig. 8. First Generation GCA of the complete graph with 3 vertices using Rule 16.

In general the language generated by the GCA will follow the relation,

$$L_2\left(M\left(G_k\right)\right) \subseteq L_4\left(M\left(G_k\right)\right) \subseteq L_8\left(M\left(G_k\right)\right) \subseteq L_{16}\left(M\left(G_k\right)\right).$$

Remark 1. The language of the kth generation is predicted under the assumption the underlying GCA is a complete graph the language generated by the kth generation GCA will be a subset of the corresponding language of the kth generation automaton by taking G_0 a complete graph with the assumption that the same rule is applied in both the cases.

2. In Example 1, the first generation GCA G_1, the final state q_3 cannot be reached then it will not generate any language. In this case $(n-1)$th state will be the final state (ie)q_2 as a final state, the GCA will not generate any language. This is because of the place a_{13} of the information matrix. (Since a_{13} is shifted to a_{12} in the first generation). In Example 1, in $A\left(G_0\right)$, a_{13}^0 is 0. So, there is no transition from q_0 to q_2 in G_0. Hence q_1 will be the final state and also that there is no transition from q_0 to q_1 in G_1. Hence the language generated by G_1 is empty language is the limitation that should be explored in some other way. Our future work explores in this direction.

The kth generation of the problem matrix for some of the rules is shown below.

In the following discussion we assume that $A\left(G_0\right) = \left[a_{ij}\right]$. In general we assume that $A\left(G_k\right) = \left[a_{ij}^k\right]$.

Now out of 512 rules, we have chosen some of the most frequently used rules and the entries of $A\left(G_k\right)$ using these rules are explained.

5. Illustration

In this illustration, a simpler version of obtaining the information matrix in the next generation using Rule 3 and Rule 25 were given. This helps us in finding $A\left(G_1\right)$ and the corresponding finite automaton.

Rule 3 Using Rule 3, the (i, j)th entry of $A\left(G_k\right)$ namely $\left[a_{ij}\right]^k$ is,

$a_{ij}^n = a_{ij}^{n-1} + a_{i(j+1)}^{n-1}$, $1 \le i, j \le m$.

From the above equation we note that, the steady state will be reached in the $(n-1)$th generation and there after the language generated by the GCA remains the same for all the rules.

Example 9.

$$A\left(G_0\right) = \begin{bmatrix} 0 & a & 0 \\ 0 & b & a \\ 0 & b & 0 \end{bmatrix}$$

Using Rule 3, we have a 9×9 matrix,

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ 0 \\ 0 \\ b \\ a \\ 0 \\ b \\ 0 \end{bmatrix} = \begin{bmatrix} 0+a \\ a+0 \\ 0 \\ 0+b \\ b+a \\ a \\ 0+b \\ b+0 \\ 0 \end{bmatrix} = \begin{bmatrix} a & a & 0 \\ b & a,b & a \\ b & b & 0 \end{bmatrix}.$$

Rule 5 The next generation of the rule matrix will follow the rule, $a_{ij}^n = a_{ij}^{n-1} + a_{(i+1)(j+1)}^{n-1}$, $1 \le i, j \le m$.

Rule 6 Entries of the next generation will follow the rule $a_{ij}^n = a_{i(j+1)}^{n-1} + a_{(i+1)(j+1)}^{n-1}$, $1 \le i, j \le m$.

Rule 9 Using Rule 9 the next generation of the rule matrix will follow the rule $a_{ij}^n = a_{ij}^{n-1} + a_{(i+1)j}^{n-1}$, $1 \le i, j \le m$.

Rule 10 The rule matrix of the next generation GCA will follow the rule $a_{ij}^n = a_{i(j+1)}^{n-1} + a_{(i+1)j}^{n-1}$, $1 \le i, j \le m$.

Rule 11 The next generation GCA' rule matrix will follow the rule $a_{ij}^n = a_{ij}^{n-1} + a_{i(j+1)}^{n-1} + a_{(i+1)j}^{n-1}$, $1 \le i, j \le m$. (ie) $a_{ij}^n = a_{ij}^{n-1} + Rule10$

Rule 12 The next generation GCA' rule matrix will follow the rule $a_{ij}^n = a_{(i+1)j}^{n-1} + a_{(i+1)(j+1)}^{n-1}$, $1 \le i, j \le m$.

Rule 13 The next generation GCA' rule matrix will follow the rule $a_{ij}^n = a_{ij}^{n-1} + a_{(i+1)j}^{n-1} + a_{(i+1)(j+1)}^{n-1}$, $1 \le i, j \le m$. (ie) $a_{ij}^n = a_{ij}^{n-1} + Rule12$

Rule 14 The next generation GCA' rule matrix will follow the rule $a_{ij}^n = a_{i(j+1)}^{n-1} + a_{(i+1)j}^{n-1} + a_{(i+1)(j+1)}^{n-1}$, $1 \le i, j \le m$. (ie) $a_{ij}^n = a_{i(j+1)}^{n-1} + Rule12$

Rule 15 The rule matrix of the next generation GCA will follow the rule $a_{ij}^n = a_{ij}^{n-1} + a_{i(j+1)}^{n-1} + a_{(i+1)j}^{n-1} + a_{(i+1)(j+1)}^{n-1}$. (ie) $a_{ij}^n = a_{ij}^{n-1} + Rule14$. **Rule 21** The next generation GCA' rule matrix will follow the rule $a_{ij}^n = a_{ij}^{n-1} + a_{(i+1)(j+1)} + a_{(i+1)(j-1)}^{n-1}$

Rule 22 The next generation GCA' rule matrix will follow the rule $a_{ij}^n = a_{i(j+1)}^{n-1} + a_{(i+1)(j+1)} + a_{(i+1)(j-1)}^{n-1}$

Rule 25 The rule matrix of the next generation GCA will follow the rule $a_{ij}^n = a_{ij}^{n-1} + a_{(i+1)j}^{n-1} + a_{(i+1)(j-1)}^{n-1}$, $1 \le i, j \le m$.

Example 10.

$$A(G_0) = \begin{bmatrix} 0 & a & 0 \\ 0 & b & a \\ 0 & b & 0 \end{bmatrix}$$

Using Rule 25,

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ 0 \\ 0 \\ b \\ a \\ 0 \\ b \\ 0 \end{bmatrix} = \begin{bmatrix} 0+0 \\ a+0+b \\ 0+b+a \\ 0+a+0 \\ b+0+b \\ a+b+0 \\ 0+0+0 \\ b+0+0 \\ 0+0+0 \end{bmatrix} = \begin{bmatrix} a & a,b & a,b \\ a & b & a,b \\ 0 & b & 0 \end{bmatrix}.$$

5.1. Representation of GCA in computer

Representing the GCA in computer includes information matrix and rule matrix. The information matrix is a square matrix of order $n \times n$, which represents vertices of the graph and the entries represent the input symbols corresponding to the vertices. A matrix with n states/rows is fixed and the amounts to $O(1)$. The class of memory occupancy class is $O(n^2)$ to represent a GCA in computer and for information matrix multiplication the memory occupancy class will be less than $O(n^3)$.

6. Conclusion

The aim of this research is to extend the concept of Graph Cellular Automaton to define the k_{th} generation GCA using basic linear rules. On viewing a finite automaton as a GCA the language generated by the finite automaton at each of the generation was predicted. The main advantage of this work is as it is defined for the basic linear rules the language generated by the k_{th} generation FA by any of the 512 rules can also be predicted. All defined concepts are illustrated through examples. Depending upon the system behaviour at each generation the rule can be changed and on having the pattern of the change of rules at each generation, the language generated by kth generation can also be predicted. The future work in this direction is to explore more applications of the k_{th} generation GCA.

7. Acknowledgement

The authors thank The Management, SSN Institutions and The Principal for providing encouragement and support for the successful completion of this research.

References

[1] Nayak K. Colour, Sudhakar sahoo, Sushant Kumar Rout, Colour graph: An efficient model for 2 dimensional Cellular Automata, Orissa Mathematical Society Conference, 2008, 0802.3626.
[2] Yasser Hassan, Eiichiro Tazaki, Cellular automata using rough set approach to the traffic decision system, in: The 16th Annual Conference of Japanese Society for Artificial Intelligence 2002, 2002, 3F2-02.
[3] J.B. Liu, X.F. Pan, Minimizing Kirchhoffindex among graphs with a given vertex bipartiteness, Appl. Math. Comput. 291 (2016) 84–88.
[4] J.B. Liu, X.F. Pan, F.F. Hu, Asymptotic Laplacian-energy-like invariant of lattices, Appl. Math. Comput. 253 (2015) 205–214.
[5] Krzysztof Malecki, Graph cellular automata with relation-based neighbourhoods of cells for complex systems modelling: A case of traffic simulation, Symmetry 9 (2017) 322.
[6] Krzysztof Malecki, Jankowski Jaroslaw, Mateusz Rokita, Application of graph cellular automata in social network based recommended system, in: International Conference on Computational Collective Intelligence ICCCI 2013: Computational Collective Intelligence. Technologies and Applications, 2013, pp. 21–29..
[7] Carteson Marr, MOrc-Thorsten Hiutt, Cellular automata on graphs: Topological properties of ER graphs evolved towards low- entropy dynamics, Entropy 14 (2012) 993–1010.
[8] N. Moreno, F. Wang, D.J. Marceau, An object- based land-use cellular automata model to overcome cell size and neighbourhood sensitivy, in: Proceedings of GEOBIA 2008- Pixels, Objects, Intelligence GEOgraphic Object Based Image Analysis for the 21st Century, 2008, pp. 5–8..
[9] Kenchi Morita, Satoshi Ueno, Parallel generation and parsing of array languages using reversible cellular automata, Int. J. Pattern Recogn. Artif. Intel. 08 (02) (1994) 543–561.
[10] David O'Sullivan, Graph cellular automata a generalised discrete urban and region model, Environ. Plan. B: Plann. Des. 28 (2001) 687–705.
[11] B. Praba, R. Saranya, Information system with decision variables and rough finite state automaton, J. Adv. Res. Dyn. Control Syst. 01 (2018) 364–373.
[12] Paul L. Rosin, Xiangfang Sun, Detection using cellular automata, in: Cellular Automata in Image Processing and Geometry, in: Emergence, Complexity and Computation, vol. 10, Springer International Publishing-Switzerland, 2014.
[13] D. Stevens, A GIS based irregular cellular automata model of land-use change, Environ. Plan. B: Plann. Des. 34 (2007) 708–724.
[14] Antomia Tretyakova, Franciszek Seredynski, Pascal Bouvry, Graph based cellular automata approach to maximum lifetime coverage problem in wireless sensor networks, in: 2014- IEEE International Parallel & Distributed Processing Symposium Workshop, 04, Dec 2014, 978-1-4799-4116-2114.
[15] Lourens J. Waldorp, Jolanda J. Kossakowski, Mean field dynamics of graphs I: evolution of probabilistic cellular automata for random and small world graphs, in: International Conference on Computational Collective Intelligence ICCCI 2013: Computational Collective Intelligence. Technologies and Applications, 2018, arxiv:1610.05105.
[16] Ada H. Y. Yuen, Robin H. Kay, Applications of Cellular Automata, Conference Proceedings, 2010.