Journal of Network and Information Security
Volume 5 Issue 2 2017
ISSN.: 2321-6859

A Survey on Cellular Automata with the Application in Pseudo Random Number Generation

I. Gethzi Ahila Poornima[1], B. Paramasivan[2], K. Mohaideen Pitchai[3], M. Bhuvaeswari[4]
[1]Department of Computer Science and Engineering, National Engineering College, Anna University, Tamil Nadu, India. Email:gethzi.akila@gmail.com
[2]Department of Computer Science and Engineering, National Engineering College, Anna University, Tamil Nadu, India. Email: bparamasivan@yahoo.co.in
[3]Department of Computer Science and Engineering, National Engineering College, Anna University, Tamil Nadu, India. Email: mohaideen1981@gmail.com
[4]Department of Computer Science and Engineering, National Engineering College, Anna University, Tamil Nadu, India. Email: itsbhuvana@gmail.com

Abstract: The Cellular Automata (CA) were invented in the late 1940 by Stanislaw Ulam and John Von Neumann. CA are simple models of computation in which the components act together and exhibit complex behavior. Initially CA are represented as model of self-reproducing organisms. Later they are applied in various areas like Physics, biology and other applications. The self-reproducing behavior is then utilized to construct Universal Turing Machine. This Survey is about the applications of CA closer to Computer Science especially designing Pseudo Random Number Generator.

Keywords: Cellular Automata, CA, Applications of CA , Pseudo random number generator, PRNG, 1D CA rules.

I. Introduction

Cellular Automata consist of unlimited lattice of cells of d dimensions. One dimensional CA consist of a row of cells and set of rules. Two dimensional CA consist of a table or matrix of cells and a collection of rules. At time t each cell will be in any one of the permissible states. At time 't' the rules are applied to a set of cells to generate a new generation of CA [1]. The rules involve the states of the neighbor cells. The neighborhood was defined by various authors. Alvy Ray Smith III [2] defined the neighborhood template. A cellular space is labeled by pair (T,r) where T is the neighborhood template and r is the number of permissible states for cells. Some of the most popular neighborhood templates are Von Neumann neighborhood (Orthogonal) and Moore (Unit Cube) neighborhood.

A. Classification of Cellular Automata

Wolfram classified CA into four classes based on typical behavior [3]. These classes are Class I-Evolution of CA which leads to trivial Configurations, Class II- Evolution of CA which leads to periodic Configuration, Class III-Evolution of CA which leads to chaotic and Class IV-Evolution of CA which leads to complicated and persistent structure. Wentian Li et al [4] classified CA into six classes based on the differences between their statistical measures. They are spatially homogenous fixed points, spatially inhomogeneous fixed points, periodic behavior, locally chaotic behavior, chaotic behavior and complex behavior. Based on some static characteristics, CA can be categorized into different types. Based on the dimensions of CA, they are classified as one dimensional, two dimensional and N-dimensional CA. Based on the capabilities of changing cell status (current state), CA is of two types. They are Programmable CA (PCA) and Controllable CA (CCA). In PCA, the action of some cells can be controlled via some Rule Control Signal (RCS). The RCS will decide the rule to be applied to that particular cell. In CCA, the actions of some cells are controlled via Cell Control Signal (CCS) in addition to RCS. The classification of Cellular Automata is given in the Fig. 1

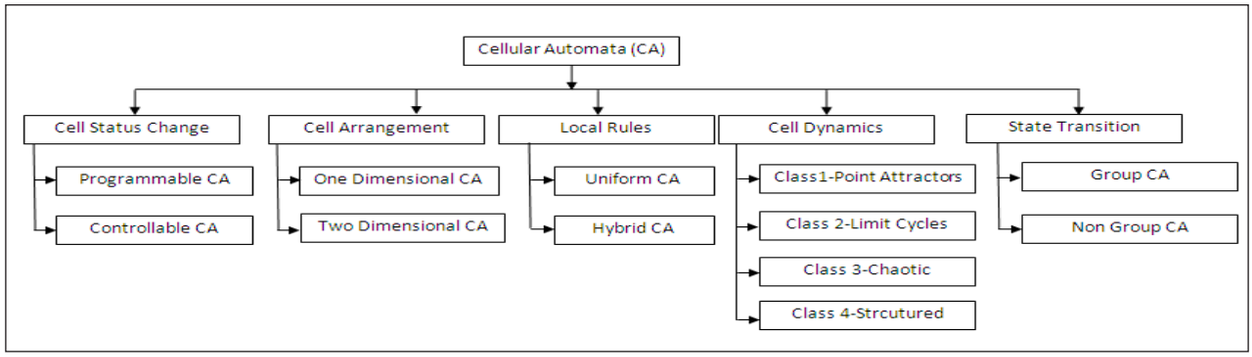


Fig. 1: Classification of Cellular Automata

B. Rule Space for One Dimensional CA

The elementary CA is a Cellular automaton with three neighbor including cell itself and of length 8 bit. The Elementary Cellular Automata rule is represented by a Look up Table which is a list of state values to which block configurations (000,001,010,011,100,101,110,111) are mapped. This list is called Rule table. The possible no of different neighborhood combination will be 2^3=8 and the number of rules that can be applied is 2^8=256. The Rule can be represented as $x_i^{t+1} = f(x_{i+1}^t, x_i^t, x_{i-1}^t)$, where the superscript t represents time and subscript represents spatial. So the rule table is in the form: (t_7t_6 t_5t_4 t_3t_2 t_1t_0). The block configuration 111 maps to t_7, block configuration 110 maps to t_6 etc.. Wolfram classified the rules into four distinct classes similar to the CA classification. The class I rules are called Null Rules. The class II rules are fixed point rules or periodic rules. The class III rules are chaotic and class IV rules have long transients and cannot be included to any of these classes. Wentian Li et al [4] presented the investigation of the structure of the elementary Cellular Automata rule space. They made some refinements in Wolfram's classification of rule space and classified the rule space into six classes. Also they viewed the rule structure as a hypercube of size m where m is the number of permissible states for each cell.

In [5], it has been reported that there are two primary regimes of rule space. They are Periodic and Chaotic separated by a transition regime. This distinction is done based on the control parameter λ which is defined as the percentage of all the entries in a rule table which map to non-zero states. The rule spaces for cellular Automata with two permissible states per cell have symmetry with respect to a point λ=0.5. The Fig. 2 shows a typical structure of Rule Space of Cellular Automata with two possible states for each cell. The authors in [4] classified the rules based on their behavior as Null Rules (homogenous fixed point rules N), Fixed Point Rules (inhomogeneous fixed point rules F), Periodic Rules (P), Locally chaotic Rules (Chaotic dynamics confined by the domain wall L), and Global chaotic Rules (Rules with random-looking spatial-temporal patterns or with exponentially divergent cycle lengths as lattice length is increased C).

Also they compared these classes with the classification done by Stephen Wolfram and reported Class I - Null rules, Class II - Fixed point and periodic rules, Class III - Global Chaotic rules and Class IV - cannot be fitted into any of these classes (belong to chaotic rules).

So all the 256 rule are under one of the above said five classes. Rules 0, 8, 32, 40, 128, 136, 160,168 belong to Null class. Rules 2, 4, 10, 12, 13, 24, 34, 36, 42, 44, 46, 56, 57, 58, 72, 76, 77, 78, 104, 130, 132, 138, 140, 152, 162, 164, 170, 172, 184, 200, 204, 232 belong to Fixed point class. 1,3, 5, 6,7,9,11 ,1 4,15, 19,23, 25, 27,28, 29,33, 35,37, 38,41,43, 50, 51,74, 108, 131, 133, 134, 142, 156, 178 belong to Periodic class. Locally chaotic rules are rules 26, 73,154 and chaotic rules are 18, 22, 30, 45, 54, 60, 90,105, 106, 129, 137, 146, 150 and 161.

Stephen Wolfram [6] showed the pattern generated for all the 256 rules (named as Wolfram's Rule) of the elementary CA. Pattern of some of the rules are as shown in Table I. The cell with state 0 will be a white dot and the cell with the state 1 will be black dot in the pattern. The authors in [7] partitioned the elementary CA rules into three classes on the basis of the growth of initial configuration with finite pattern in quiescent backgrounds. First class contains the rules which generate pattern length of size zero (i.e. Zero pattern) starting form an initial configuration with finite number of active sites (i.e. non-quiescent). Second class contains the rules which generate a constant number of configurations (i.e. constant no of configurations). Third class contains the CA rules which generate growing pattern. The rules in various classes will exhibit different dynamic behavior. They reported that rules in the first class exhibit very simple dynamics. Some rules inside the second class exhibit shift-like dynamics. Also some of the rules in the third class grow indefinitely.

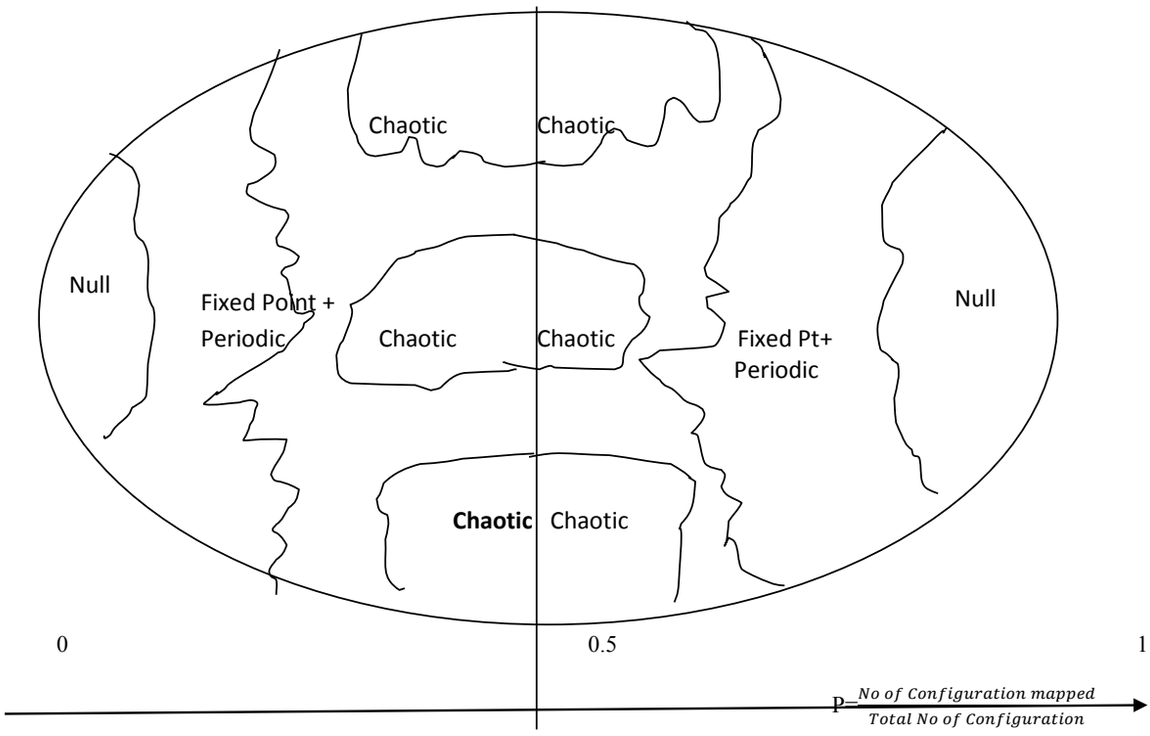


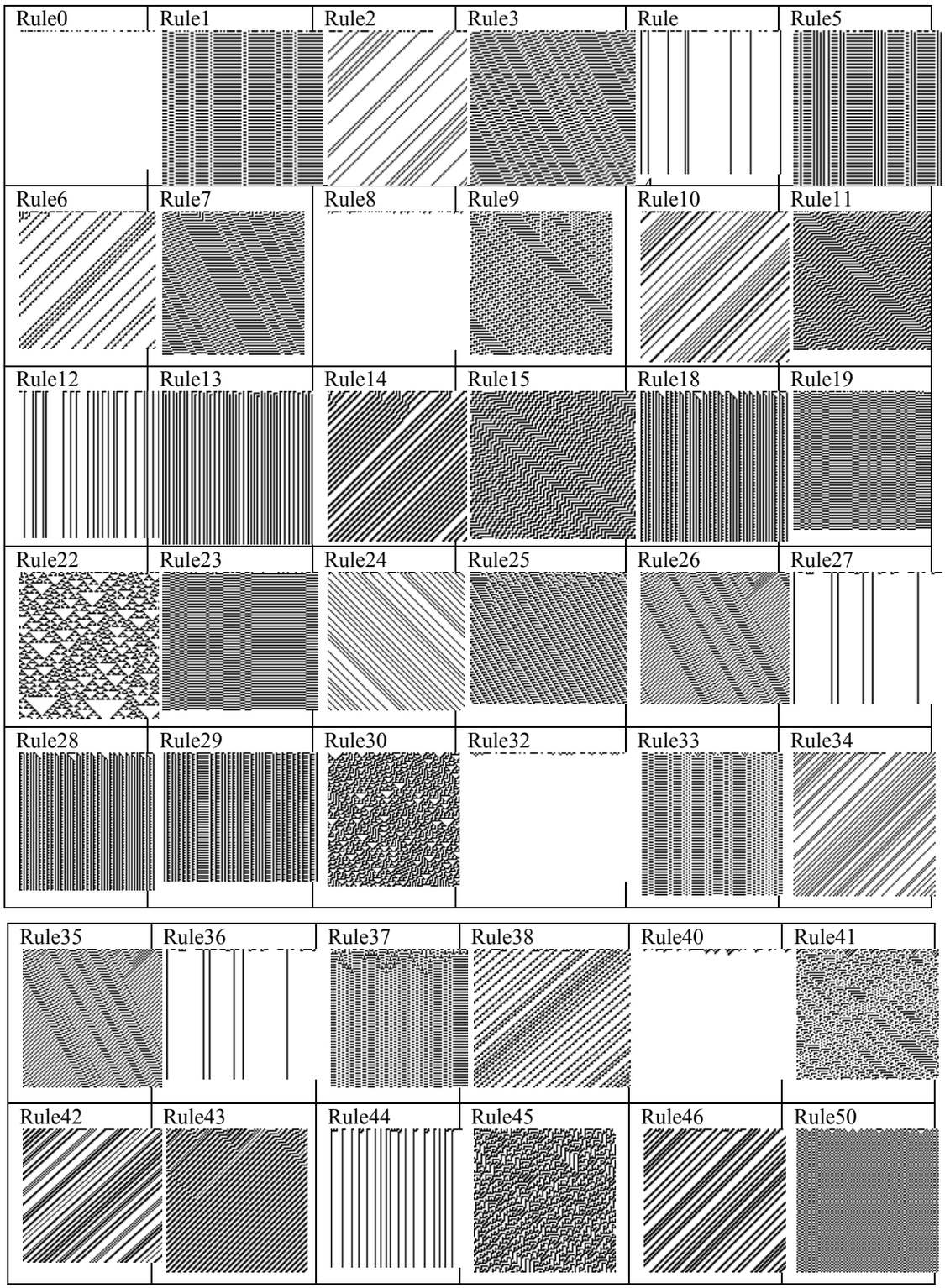
Fig. 2: Structure of Cellular Automata Rule Space

C. Characteristics of Cellular Automata Rules

Some properties of Cryptographically Strong CA Rules presented in various sequels have been discussed by the authors in [9] as follows

- Non-Linearity: Non Linearity of a Boolean function (Rule table) f of 'n' variables is the distance from f to set of affine functions with 'n' variables. Non linearity is needed for the cryptographic applications.
- Algebraic Degree: The maximum number of literals in any conjunction of ANF (Algebraic Normal Form - Any Boolean function expressed as XOR of conjunctions and a Boolean constant) of a Boolean function is called its degree. For example f(x1, x2) = x1 • x2 ⊕ x2 has algebraic degree 2. Linear functions have algebraic degree 1 since all the linear function has only XOR logic in its ANF. So Linear Rules have algebraic degree 1.
- Balancedness: A Boolean function of n variables is said to be balanced if for exactly 2n-1 assignments the function will evaluate to 0 and for exactly 2n-1 assignments the function f will evaluate to 1.

TABLE I: PATTERN GENERATED BY SOME OF THE RULES IN THE SET OF 256



- Correlation Immune: A function in n variables is correlation immune of order k, $1 \leq k \leq n$, if and only if all of the Walsh transforms $1 \leq wt(w) \leq k$; are equal to zero.
- Resiliency: A Boolean function which is balanced and correlation immune of order k is said to be a k-resilient function.

- D-Monomial Test: It is a statistical test for pseudo randomness. If a Boolean function of n variables is a good pseudorandom sequence generator, then it will have ½ d- degree monomials.

D. 2D Cellular Automata Rule Space

Two Dimensional CA is an array of 1D CA. There are many possible lattices and neighborhood structures for 2D CA. Most of the researches used square lattices with two kinds of neighborhood structure mentioned in Fig 1.1 and Fig 1.2.

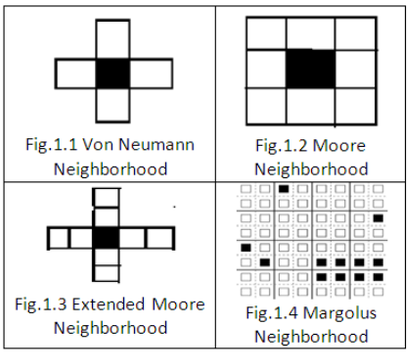


Fig.1.1 Von Neumann Neighborhood

Fig.1.2 Moore Neighborhood

Fig.1.3 Extended Moore Neighborhood

Fig.1.4 Margolus Neighborhood

Stephen Wolfram in [10] tabulated the number of possible rules of various kinds for cellular Automata with two permissible states and neighborhood as Von Neumann Neighborhood and Moore neighborhood as follows.

Rule Type	5-Neighbor Square	9-Neighbor
General	2^{32}	2^{512}
Rotationally symmetric	2^{12}	2^{140}
Reflection Symmetric	2^{24}	2^{288}
Completely symmetric	2^{12}	2^{102}
Outer Totalistic	2^{10}	2^{18}
Totalistic	2^{5}	2^{9}

E. Game of Life

A 2-dimensional CA was proposed by John Conway, called "Game of Life", which is based on biological Model. Each cell can have two possible states i.e. 1 (alive) or 0 (dead). The state of a cell may depend only on the states of the neighboring cells in the previous time step or it may depend on states of neighbor over several previous times (Memory CA). The local rule is described as follows:

- Survival

  If a cell is in state 1 (alive) and has 2 or 3 neighbors in state 1, then the cell survives, i.e., remains in state 1.

- Birth

  If a cell is in state 0 and has exactly 3 neighbors in state 1, then in the next time step the cell goes to state 1.

- Deaths

  A cell in state 1 dies (goes to state 0) of loneliness if it has 0 or 1 alive neighbor. Also, it dies because of overcrowding if it has 4 or more alive neighbors.

David Epstein [11] followed a standard convention for naming these Cellular automata in which the update rule is represented as 'Rule String'. The format of this string is "B_{xxx}/S_{yyy}"

Where xxx and yyy are the sub set of string {0, 1…8}. The string xxx represents numbers of alive neighbor cells for a dead cell to be born in the next time step. It causes the event "Birth". The string yyy represents the numbers of alive neighbors to be survived in the next time steps. It causes the event "Survive" i.e. 1 remains 1 in the next time step. The aforementioned Conway's Game of life is represented by the rule string B_3/S_{23}.

II. Applications of Cellular Automata

A. Cellular Automata Games

- *Game of Life* [12] - In late 1960s British Mathematician John Conway invented a virtual mathematical machine that operates on a two dimensional array of square cell. Each cell takes one of two states. i.e. dead and alive. Based on some local rules the alive may survive or dead and the dead cells can be born or die in the next generation. This can be modeled by a two dimensional CA with two possible states 0- dead, 1-alive.
- *Firing Squad* [13] - The goal of the real world firing squad problems is to design a system of rules according to which an officer can command an execution detail to fire that its members fire their rifles simultaneously. This problem can be modeled by Cellular automata since the CA has an array of cells whose transition of states depends on the local transition rule or function.
- *Firing mob* [14] is the generalization of firing squad problem.
- *Queen bee* [15] is a kind of reverse of the firing squad Synchronization Problem (FSSP). In FSSP initially a single cell is designated and the at the final state, all the cells have to be in the same state (Active). i.e. all the soldiers at final should shoot their rifles. But here in the Queen Bee problem, initially all the cells are in same state, and at the final stage, only one cell has to be designated. This kind of task can be termed as Leader Selection.
- *Iterated Prisoners Dilemma* [16] - In this game player can adopt either one of the two strategies: Co-operate (C) or Defect (D). Cooperation results in a benefit b to the opposing player but incurs a cost of c to the cooperator. Defection has no costs or benefits. So Prisoner's Dilemma game played on a square grid. Each cell is occupied by one player. Each player plays one round of the Prisoner's Dilemma game against his/her eight nearest neighbors.

The sum of payoffs from these eight games is the payoff of each player. After every iteration, each player looks at his/her neighbors and switches his/her strategy to the strategy that has obtained the highest score. As a result of the repeated games only the best strategies survive – the ones which give the greatest payoffs. There are some characteristic states of the configuration. Usually strategies group in clusters. This situation can be modeled using one dimensional Cellular Automata.

a. CA as Parallel computing machine [17],[18]-Two dimensional CA is used for image processing and pattern recognition. CA can be used for building parallel multipliers, prime number sieves, and parallel processing computers and also for sorting machines.
b. CA for modeling Physical and biological Systems [17], [18] - CA models for various forms of regular and random growth based on two dimensional CA. Pattern formation in reaction-diffusion systems has been achieved by the Cellular automata. Hydro dynamical system has been modeled using CA. The immune system was modeled by using CA [19],[20].
c. In Social Sciences [17]-Peter S Albin was the first to use CA for the checkerboard models in his book "The Analysis of Complex Socioeconomic Systems". Also CA have been used to model traffic flow and a design tool for urban development.
d. In VLSI applications [17], [18]- B.K. Sikdar et al [21] proposed a fault diagnosis scheme for VLSI circuits. A special class of one dimensional CA called MACA (Multiple Attractor CA) has been used for this design. In [22] the authors have used two dimensional CA for testing VLSI circuits. Cellular Automata have been used for test pattern generation in [23]. They proposed Cellular Automata as a framework for Built in Self-Test (BIST) structures. Chowdhury et al. introduced the CA based error correcting codes (CAECC) [24]. The encoder/decoder circuit complexity for CAECC has been shown to be low. They proposed CA based single byte error correcting and double byte error detecting code. K Paul et al enhanced this method by introducing the concept of extension field.
e. Cryptography [2]- S. Wolfram [25] reported that Cellular Automata provides the randomness in the physical systems. Since randomness is the most desirable property in cryptography, Cellular automata have been widely applied in cryptography. He presented a stream cipher based on a simple one dimensional Cellular automaton in 1985 [32].
f. Particle Simulations [26], [27], [28] [29]-C Burstedde et al [27] simulated pedestrian traffic using 2-dimensional Cellular Automata. Each pedestrian was considered as a particle thereby performing particle simulation using CA. The authors in [28] studied evacuation process using stochastic cellular automaton for pedestrian dynamics.
g. Exploring Ancient Architectural Designs [29][31]-Hokky Situngkir in [31] reported the utilization of three dimensional forms (buildings, arts, etc..) emerged by 2D CA employing local rules for the construction of next generation.

III. Pseudo Random Number Generators Using CA

A. Conventional Pseudo Random Number Generators

Pseudo Random Number Generators (PRNG) refers to an algorithm that uses mathematical formulae to produce sequence of random numbers. PRNGs generate a sequence of numbers approximating the properties of random numbers. They are fundamental to the use of cryptographic and key generation as they ensure message uniqueness.

a. Linear Congruential Generator (LCG)

Caroline Fontaine [32] presented a detailed study of Linear Congruential PRNG. He described the LCG as a Pseudo Random Number Generator that produces a sequence of numbers x1, x2, x3 … according to the recurrence relation $x_t = ax_{t-1}+b \pmod n$. This generator passes all the statistical tests such as frequency test, poker test, run test and auto correlation test. The major disadvantage with this method is predictability i.e. given a piece of the sequence; it is easy to reconstruct all the rest of it.

b. Multiplicative Linear Congruential Generator (MLCG)

It is one of the oldest and the best simple generator used by D.H Lemer in 1948. Here each successive number in the sequence is obtained by multiplying the previous one by a multiplier and optionally adding another constant and throwing away the most significant digits of the result: $s_{i+1} \equiv (as_i+c) \bmod m$ where a is the multiplier chosen, m is either the largest integer or slightly smaller than the largest integer that can be represented in one computer word.

c. Fibonacci Pseudo Random Number Generators (FPRNG)

Fibonacci Pseudo Random Number Generator generates the sequence in which each number is obtained by applying some operations on the preceding two numbers. The lagged Fibonacci generators are similar to the FPRNG except that the operations are performed on any two numbers in the sequence not necessarily the last preceding two. This will increase the unpredictability. FPRNG has been used in [33, 34].

d. Linear Feedback Shift Registers PRNG

Linear Feedback Shift Registers provide a simple means for generating non sequential list of numbers quickly on microcontrollers. This Random number generation process involves a right shift operation and an XOR operation.

M.Sahithi et al [36] presented a 8 bit random number generator using Linear feedback shift register.

e. PRNG using Cellular Automata

The CA based PRNGs are applied in many environment such as stream ciphers, constructing collision free hash function. The stream cipher is more secure because of the difficulty of retrieving the initial seed from the obtained temporal sequence. So it is very hard for the cryptanalyst to break the key. All the existing Pseudo Random Number Generators are linear. Wolfram [37] was the first to explore the cellular automata evolution in the pseudo Random number generation. He used a very simple rule for the evolution of Random numbers. The rule is as follows,

$a'_i = a_{i-1}$ *XOR* $(a_i$ OR $a_{i+1}) \Rightarrow a'_i = (a_{i-1}$ OR a_i OR a_{i+1} OR *ai* $a_{i+1})$ mod 2

The above Rule is Rule No 30. This rule is nonlinear. But the dependence of the rule on is linear. He studied the behavior of the CA starting from all initial states. Lyapunov exponents which measure the rate of information transmission in CA have been calculated.

The authors in [38] designed a symmetric key cryptography system based on Vernam Cipher. The keys required for the encryption process are generated by PRNG using CA. The quality of the PRNG depends on the set of rules applied. These rules are presented in this paper by an evolutionary algorithm called Cellular Programming. Entropy has been used as the fitness function. The Rule set contains 8 rules of radius r=1 and nearly 39 rules of radius r=2. The Rule set thus generated are tested and the resulting key sequence are highly resistant to attempts of breaking the key.

M. Szaban [39] et al explored genetic algorithm to find suitable rules from the set of 47 rules obtained in [38]. 10 subsets of good quality are rules are finally selected. Thus a high quality PRNS are obtained by using non- Uniform CA. The works in [37, 38] used Non-Uniform CA. But the implementation in hardware is too difficult and it needs large memory to store all the rules than Uniform CA.

Ping Ping [40] et al generated pseudo Random Number sequence by employing Uniform CA with the next nearest neighborhood. NIST statistic test and time spacing sampling techniques are used to find the suitable rules. They proved that the PRNG using Uniform CA is as good as PRNG using Non Uniform CA. At the same time the PRNG using uniform CA is better than PRNG using non uniform CA with respect to hardware implementation

Peter [41] et al used Rule 30 CA for pseudorandom Number generation which in turn is used for Built in Self test. They analyzed the characteristic of Rule 30 one dimensional CA and they observed that the immediately neighboring sites have a correlation of 0.5 which means that the words thus formed by considering all the sites are not independent. But the correlation dies out when it goes 3 or 4 sites away. So they introduced the "site spacing". So the correlation between adjacent sites can be reduced. They reported another way of removing correlation i.e. time spacing. They proved that as time spacing factor (Beta) increases the auto correlation between the adjacent sites decreases. Also they used Rule 45 CA for PRNG and applied time and site spacing. It has been reported that Rule 45 CA Pseudorandom Number generator's performance is not so good as Rule 30 CA.

The authors [42] employed two dimensional CA for pseudo Random Number Generation. The evolutionary cellular Programming has been used for generating rules for evolving 2D CA. They reported that there is no need of using time spacing for reducing the correlation. The lack of time spacing may facilitate hardware implementation. Without time spacing, the 2D CA PRNG can produce a good quality random sequence same as one dimensional CA with time spacing. It has been reported that the 2D CA generates sequence of higher cycle length i.e. e average cycle length for an n-cell CA increases exponentially and is on the order of 2^{n-3}.

In [43] a new class of Cellular Automata called Self-Programmable CA (SPCA) has been proposed specifically for the application of Pseudorandom Number Generation. The state transition rules to be applied at each iteration (evolution) are programmed and it keeps on changing so that it could create dynamic behavior. The dynamic rule selection process has been executed via two Rule vectors which controls the selection of rules. The authors have used 2 Self Programmable Cellular Automata (SPCA) with the rule combinations SPCA 90/165 and SPCA 150/105. It has been reported that the random sequence generated by Uniform CA 90 and the Uniform CA 150 fails maximum number of tests in DIEHARD and ENT when used separately. But it passes some of the tests in DIEHARD and ENT when Hybrid 90/150 CA. But the results are from the SPCA 90.165 and SPCA 150/105.

S.No	Author	Methodology	Pros	Cons
	Wolfram et al [37]	Non linear Rule is used for the pseudo-random number generation. (Rule 30) Total number of sequences generated is computed by changing different initial configuration.	Even though the rule is simple, it could produce highly random sequence. Even a small perturbation in the initial state will lead to a large difference in the sequence generated.	Some sort of regularity occurs in the sequences which affect the randomness.

S.No	Author	Methodology	Pros	Cons
	Peter D.Hortensius, Robert D Mcleod, Werner Pries[41]	CA is employed in pseudorandom sequence generator which in turn used in built-in-self test. Rule 30 and Rule 45 are used.	Implementation advantage over conventional linear generator is that it can be designed to require only adjacent neighbor communication and they are cascadable. It is comparatively more random than the conventional Linear Feedback Shift Register –based pseudorandom generators. Cross correlation is reduced.	Maximum cycle length for CA based PRNG is low compared to LFSR generator. 2. Rule applied here is more dependent on the start values.
	Franciszek Seredynski et al [38]	CAs are applied to generate pseudo random number sequence which has been used in encryption process of Vernam cipher. Non uniform CAs are used and the cellular programming technique is used to find rules for non uniform CAs.	The random sequence used as keys have robustness and it is highly resistant for breaking cryptography key.	The neighborhood radius is taken as only 1.
	Ping Ping, Feng Xu and Zhi-Jian Wang [40]	Uniform CA are employed to generate PNS. An algorithm has been proposed to select three rules from a set of 10 rules	It's performance could be equal to the Non uniform CA. Hardware implementation is very easy because the same rule is applied to all the cells in the CA configuration.	It cannot outperform the Non uniform CA.
5.	M. Szaban et al [39]	A searching mechanism has been presented to find a set of rules to produce a high PNS. Genetic algorithm is used and the entropy value is taken as fitness value. The resulting PNS is applied in the Vernam cipher.	The set of rules selected by this procedure can improve the quality of cryptographic module. It results in large key space.	-
6.	Marco Tomassini, Moshe Sipper, and Mathieu Perrenoud[42].	Cellular Programming evolutionary algorithm is employed to generate two dimensional CA pseudo random number generations. The Evolutionary algorithm has been used to evolve best rules. (Non uniform CAs are used.) It has been proved that it can evolve high quality random number sequence.	The average cycle length is high (in the order of 2^{n-3}) where n is the number of cells. Time spacing is not required and thereby it facilitates hardware implementation.	It is not sure that it could generate an optimal solution because evolutionary algorithms are stochastic heuristic search techniques.
7	Sheng-Uei Guan and Syn Kiat Tan[43].	Self Programmable CA has a rule selection neighborhood for selecting rules in addition to a localized state transition neighborhood. The rule selection is done among the rule set <90/165> and <150/105>.	Output Throughput is high.	For SPCA 90/165, diehard test has not been passed for all length 18 to 24 cells.
8	Sheng-Uei Guan, Shu Zhang[46]	Two models of Controllable Cellular Automata (CCA0, CCA1 and CCA2) are proposed. Random sequences generated from these CCAs are tested against DIEHARD. The results are compared with Programmable Cellular automata.	CCA0/CCA1 types improve the overall randomness of cells. CCA2 improves the randomness of both types (Basic and controllable) cells.	The controllable cells in CCA0 and CCA1 do not improve the randomness value of controllable cells.

Comparison of the Characteristics of the Conventional PRNG and the PRNG Using Cellular Automata.

The ultimate objective of pseudorandom number generator is to generate a sequence of random numbers which cannot be predicted or any of the bits cannot be recomputed. These sequences are applied in various areas. One among them is cryptography, i.e. key generation in encryption and decryption algorithm. The Conventional PRNG involves basically linear functions. Usually linear functions can generate a very long sequence which avoids the repeatability thereby it reduces the predictability. But the sequences generated by linear functions are not so secure. On the other hand non linear functions increase the security of the cipher designs. So if the sequence has to be applied in the cryptography, the non linearity has to be applied. The PRNG using Cellular Automata involves non linear functions. So it is required to provide non linearity (to increase security) and maximum length sequence (to reduce the repetition of random numbers in the generated sequence). Dipanwita Roy Chowdhury [44] et al proposed an algorithm to synthesis maximum length non linear Cellular Automata to increase the security and reduce the repetition.

Also the non linear Cellular Automata shows high correlation between the input to the automaton and its generated sequence. For cryptographic applications, PRNG need to be non linear and also satisfy additional properties. Dipanwita Roy Chowdhury [44] et al analyzed the non linear CA for a new property called "d- monomial characteristics addition" which will form a cryptographically suitable CA applied in PRNG. Some of the important cryptographically suitable functions are discussed by Carole J Etherington in his thesis [45] are Rotation symmetric Functions, Balanced function, and Non linearity. Rotation symmetric Functions are functions whose values remain unchanged when the variables in the functions are rotated circularly to each of the possible positions. A balanced function has an equal number of 1s and 0s in its truth table values. Balancedness is an important cryptographic criteria for designing the combiner function in order to prevent the system from leaking statistical information on the plaintext when given the cipher text. The nonlinearity of a function is the minimum Hamming distance between f and all affine functions. Functions with the highest nonlinearity are called bent functions. A crypto primitive should have high algebraic degree and non linearity. But to increase randomness, it should have higher order of correlation immunity. But these behaviors are contracting with each other. The linear rules can yield good randomness and cannot give security. The non linear rules can provide higher algebraic degree and also non linearity. But it can be easily cryptanalized because of the absence of correlation immunity. Dipanwita Roy Chowdhury[9] et al proposed an algorithm to produce CA with hybrid rule set which are very well suited for cryptographic primitives. Hybrid means that the rule set contains both the linear and non linear rules.

S.No	Characteristics	Applicability in Cryptography	Conventional Pseudorandom Number Generator	Cellular Automata based Pseudorandom Number generator.
1.	Linearity	Non linearity is required in cryptography.	Presence of linearity may produce a long sequence without repletion but it is not so secure.	The lack of linearity results in more security.
2.	Length of the Sequence	Maximum length of the random sequence should be high to be applied in cryptography.	The maximum length of the random number sequence is high.	The maximum length of the random number sequence is low.
3.	Randomness	Randomness should be more to be applied in cryptography.	Better Randomness	Good Randomness
4.	Correlation immunity	Should be highly resistive against the correlation between bits.	High Correlation immunity	Low Correlation Immunity (because of non linearity).

IV. Conclusion

In this article, a survey of cellular automata applied in pseudorandom number generator has been presented. The cellular automata has been analyzed and reported that it is suitable for the pseudorandom generation process especially for the cryptography. Thus the cellular Automaton is the only model which can provide all the significant features suitable for cryptography.

References

[1] P. Sarkar, "A brief history of cellular automata," *ACM Computing Surveys (CSUR),* vol. 32, pp 80-107, 2000.

[2] ALVY RAY SMITH III, "Cellular automata complexity trade-offs," *Information and Control,* vol. 18, pp. 466-482, 1971.

[3] K. Sutner, "Classification of cellular automata," *Encyclopedia of Complexity and Systems Science,* Part 2009.

[4] W. Li, and N. Packard, "The structure of the elementary cellular automata rule space," *Complex Systems,* vol. 4, pp. 281-297, 1990.

[5] W. Li, N. H. Packarda, and C. Langton, "Transition phenomena in cellular automata rule space," *Physica D,* vol. 45, pp. 77-94, 1990.

[6] Appendix of Theory and Applications of Cellular Automata, S. Wolfram, ed. (World Scientific, 1986).

[7] G. Braga, G. Cattaneo, P. Flocchini, and C. Q. Vogliotti, "Fundamental study: Pattern growth in Elementary Cellular Automata," *Theoretical Computer Science,* vol. 145, pp. 1-26, 1995.

[8] E. Filiol, "A new statistical testing for symmetric ciphers and hash functions," *International Conference on Information and Communications Security,* Springer Link 2002.

[9] K. Chakraborty, and D. R. Chowdhury, "CSHR: Selection of cryptographically suitable hybrid cellular automata rule," *International Conference on Cellular Automata for Research and Industry,* ACRI, Springer, pp. 591-600, 2012.

[10] N. H. Packard, and S. Wolfram, "Two-dimensional cellular automata," *Journal of Statistical Physics,* vol. 38, pp. 5-6, 1985.

[11] D. Eppstein, "Growth and decay in life-like cellular automata," *Game of Life Cellular Automata,* Springer Link, 2010.

[12] A. Adamatzky, *Game of Life Cellular Automata,* Springer, 2010.

[13] H. Umeo, M. Hisaoka, and T. Sogabe, "A survey on optimum-time firing squad synchronization algorithms for one-dimensional cellular automata," *International Journal of Unconventional Computing,* vol. 1, pp. 403-426.

[14] K. Culik II, and S. Dube, "An efficient solution of the firing mob problem," *Theoretical Computer Science,* vol. 91, pp. 57-69, 1991.

[15] A. Beckers, and T. Worsch "A perimeter-time CA for the queen bee problem," *Parallel Computing,* vol. 27, pp. 15-25, 2001.

[16] Katarzyna Zbieć, "The prisoner's dilemma and the game of life," *Studies in Logic, Grammar and Rhetoric*, vol. 19, pp. 95-100, 2003.

[17] N. Ganguly, B. K. Sikdar, A. Deutsch, G. Canright, and P. P. Chaudhuri "A survey on cellular automata," 2003.

[18] V. Sharma, A. Dev, and S. Rai, "A comprehensive study of cellular automata", *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 2, pp. 340-344, 2012.

[19] R. J. De Boer, and P. Hogeweg, "Growth and Recruitment in the Immune Network," In A. F. Perelson and G. Weisbuch, editors, *Theoretical and Experimental Insights into Immunology,* vol. 66, pp. 223-247, 1992.

[20] F. Celada, and P. E. Seiden, "A computer model of cellular interactions in the immune system," *Immunology Today,* vol. 13, pp. 56-62, 1992.

[21] B. K. Sikdar, and N. Ganguly, and P. P. Chaudhuri, "Fault diagnosis of VLSI circuits with cellular automata based pattern classifier," *IEEE transaction on Computer-Aided design of Integrated Circuits and systems,* vol. 24, pp. 1115-1131, 2005.

[22] A. R. Khan, "VLSI architecture of a cellular automata machine," *Journal of Computer and Mathematics with applications,* vol. 33, 79-94,1997.

[23] A. K. Das, "Additive cellular automata: Theory and application as a built-in self-test structure," PhD thesis, I.I.T. Kharagpur, India, 1990.

[24] D. R. Chowdhury, S. Basu, I. S. Gupta, and P. Pal Chaudhuri, "Design of CAECC Cellular Automata based Error Correcting Code," *IEEE Transaction on Computers,* vol. 43, pp. 759-764, 1994.

[25] S. Wolfran, "Origin of randomness in physical systems," *Physical Review Letter,* vol. 55, p. 449, 1985.

[26] S. Wolfran , "Cellular automata as models of complexity," *Nature,* vol. 311, pp. 419-424, 1984.

[27] H. Niesche, "Introduction to Cellular Automata – Seminar," *Organic Computing,* SS2006, 2006.

[28] C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz, "Simulation of pedestrian dynamics using a two-dimensional cellular automaton," *Physica A: Statistical Mechanics and Its Applications,* vol. 295, pp 507-525, 2001.

[29] A. Kirchner, and A. Schadschneider, "Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics," *Physica A: Statistical Mechanics and Its Applications,* vol. 312, pp. 260-276, 2002.

[30] M. Rickert, K. Nagel, and M. Schreckenberg, "Two lane traffic simulations using cellular automata," *Physica A: Statistical Mechanics and its Applications,* vol. 231, pp. 534-550, 1996.

[31] H. Situngkir, "Exploring ancient architectural designs with cellular automata," SSRN, 2010.

[32] S. Wolfram, "Theory and applications of cellular automata," *World Scientific,* Singapore, 1986. ISBN 9971-50-124-4pbk.

[33] C. Fontaine, "Linear congruential generator," *Encyclopedia of Cryptography and Security,* vol. 1, pp. 721-721, 2011.

[34] D. E. Knuth, *The Art of Computer Programming: Semi Numerical Algorithms*, vol. 2, Adison Wesley, 1981(Book).

[35] G. Marsaglia, & L. H. Tsay, "Matrices and structure of random number sequences," *Linear Algebra and Its Application,* vol. 67, pp. 147-156, 1985.

[36] M. Sahithi, B. MuraliKrishna, M. Jyothi, K. Purnima, A. Jhansi Rani, N. Naga Sudha, "Implementation of random number generator using LFSR for high secured multi purpose applications," *International Journal of Computer Science and Information Technologies,* vol. 3, pp. 3287-3290, 2012.

[37] S. Wolfram, "Random sequence generation by cellular automata," *Advances in Applied Mathematics,* vol. 7, pp. 123-169, 1986.

[38] F. Seredynski, P. Bouvry, and A. Y. Zomaya "Cellular automata computations and secret key cryptography," *Parallel Computing,* vol. 30, pp. 753-766, 2004.

[39] M. Szaban, F. Seredynski, and P. Bouvry, "Collective behavior of rules for cellular automata-based stream ciphers," IEEE Conference on Evolutionary Computation, pp. 179-183, 2006.

[40] P. Ping, F. Xu, and X.-J. Wang, "Generating high-quality random numbers by next nearest-neighbor cellular automata," *Advanced material Research,* vol. 765, pp. 1200-1204, 2013.

[41] P. D. Hortensius, R. D. Mcleod, W. Pries, D Michael Miller and Howard C. Card, "Cellular Automata- Based Pseudorandom Number Generators for Built-In self-Test," IEEE transactions on Computer-Aided Design, vol. 8, pp 842-859,1989.

[42] M. Tomassini, M. Sipper, and M. Perrenoud, "On the Generation of High-Quality Random Numbers by Two-Dimensional Cellular Automata," *IEEE Transactions on Computers,* vol. 49, pp. 1146-1151, 2000.

[43] S. U. Guan, and S. K. Tan, "Pseudorandom number generation with self-programmable cellular automata," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 23, pp. 1095-1101, 2004.

[44] S. Ghosh, A. Sengupta, D. Saha, and D. R. Chowdhury, "A scalable method for constructing non-linear cellular automata with period 2n – 1," *International Conference on Cellular Automata,* pp. 65-79, 2014.

[45] C. J. Etherington, "An Anaysis of Cryptographically Significant Boolean Functions with High Correlation Immunity by Reconfigurable Computer," 2010 Thesis.

[46] S.-U. Guan, and S. Zhang, "Pseudorandom number generation based on controllable cellular automata," *Future Generation Computer Systems,* vol. 20, pp. 627-641, 2004.