

Cost Optimized Design Technique for Pseudo-Random Numbers in Cellular Automata

Arnab Mitra^{1,3} and Anirban Kundu^{2,3}

¹ Adamas Institute of Technology, West Bengal - 700126, India
mitra.arnab@gmail.com

² Kuang-Chi Institute of Advanced Technology, Shenzhen - 518057, China
anirban.kundu@kuang-chi.org

³ Innovation Research Lab (IRL), West Bengal - 711103, India
anik76in@gmail.com

ABSTRACT

In this research work, we have put an emphasis on the cost effective design approach for high quality pseudo-random numbers using one dimensional Cellular Automata (CA) over Maximum Length CA. This work focuses on different complexities e.g., space complexity, time complexity, design complexity and searching complexity for the generation of pseudo-random numbers in CA. The optimization procedure for these associated complexities is commonly referred as the cost effective generation approach for pseudo-random numbers. The mathematical approach for proposed methodology over the existing maximum length CA emphasizes on better flexibility to fault coverage. The randomness quality of the generated patterns for the proposed methodology has been verified using Diehard Tests which reflects that the randomness quality achieved for proposed methodology is equal to the quality of randomness of the patterns generated by the maximum length cellular automata. The cost effectiveness results a cheap hardware implementation for the concerned pseudo-random pattern generator. Short version of this paper has been published in [1].

KEYWORDS

Pseudo-Random Number Generator (PRNG), Prohibited Pattern Set (PPS) & Fault Coverage, Randomness Quality, Diehard Tests, Cellular Automata (CA)

1. INTRODUCTION

Production of random patterns, plays a fundamental role in the field of research work varying from Computer Science, Mathematics or Statistics to cutting edge VLSI Circuit testing. Random numbers are also used in cryptographic key generation and game playing. Mathematicians describe that this random numbers happen in a sequence where the values are homogeneously distributed over a well defined interval, and it is unfeasible to predict the next values based on its past or present ones.

In probabilistic approach, a random number [2] is defined as a number, chosen as if by chance from some precise distribution such that selection of a large set of these numbers reproduces the fundamental distribution. Those numbers are also required to be independent to maintain no correlations between successive numbers. In the generation process of random numbers over some specified boundary, it is often essential to normalize the distributions such that each differential area is equally populated. In order to generate a power-law distribution $P(x)$ from a uniform distribution $P(y)$, it is $P(x) = Cx^n$ for $x \in [x_0, x_1]$. Then normalization follows as Equation (1),

$$\int_{x_0}^{x_1} P(x)dx = C \frac{[x^{n+1}]_{x_0}^{x_1}}{n+1} = 1 \quad \dots\dots\dots (1)$$

From Equation (1), the value of C can be determined as followed in Equation 2.

We get,
$$C = \frac{n+1}{x_1^{n+1} - x_0^{n+1}} \quad \dots\dots\dots (2)$$

In practice, random number generator requires a specification of an initial number used as the starting point, which is known as a "seed". Random number generators are classified into several groups based on the difference in generation procedures of random numbers. Pseudo-random number and true-random number are most commonly used in scientific works. The classification of the random number is based upon the selection of seed, that describes how much randomized way has been adopted for the selection of that seed for generation of random pattern. The quality of any random number generator is being tested through a statistical test suit named Diehard Tests [3].

Random numbers or patterns [2], [4] obtained by means of executing a computer program, is based on implementing a particular recursive algorithm are commonly referred to as pseudo-random numbers. "Pseudo", emphasizes that the selection of seed is in deterministic way.

The quality of randomness generated by any random number generator is needed to be verified. For this purpose, the diehard tests are a battery of statistical tests for measuring the quality of a random number generator. This statistical test suit was developed by George Marsaglia over several years and first published in 1995 on a CD-ROM of random numbers [3]. The following tests are executed as enlisted below to measure the quality of the randomness of any particular random pattern generator:

- i) Birthday Spacings:** This name is based on the birthday paradox. Here a pair of random points on a large interval is chosen such that the spacing between the points should be asymptotically exponentially distributed.
- ii) Overlapping Permutations:** In this test, the sequences of five consecutive random numbers are analyzed to observe overlapping of the permutations; a total 120 possible orderings should occur with statistically equal probability.
- iii) Ranks of Matrices:** In this test, some number of bits are selected from some number of

random numbers to form a matrix over $\{0, 1\}$. Thereafter the rank is counted based on the determinant value of the matrices.

iv) Monkey tests: This name is based on the infinite monkey theorem. Here the sequences of some number of bits are assumed as "words". Thereafter the count is completed over the overlapping words in a stream. The number of "words" that does not appear should follow any known distribution.

v) Count the 1's: In this test, the 1 bit in each of either successive or chosen bytes is counted and then the counted value is converted into "letters" and finally following the counting of the occurrences of five-letter "words".

vi) Parking Lot Test: In this test, randomly placed unit circles in a 100×100 square are tested to find whether any of the circles, overlaps an existing one. After an iteration of 12000 tries, the number of successfully "parked" circles should follow a certain normal distribution.

vii) Minimum Distance Test: In this test, the minimum distance between the pairs of randomly placed 8,000 points in a $10,000 \times 10,000$ square is measured. The square of this distance is exponentially distributed with a certain mean.

viii) Random Spheres Test: In this test, randomly chosen 4000 points with the center a sphere on each point is placed in a cube of edge 1000 and the radius is calculated. The smallest sphere's volume should be exponentially distributed with a certain mean.

ix) The Squeeze Test: In this test a multiplication is performed with 2^{31} by random floats on $[0, 1)$ until 1 is achieved. The iteration of this multiplication occurs around 100000 times. The number of floats needed to reach 1 should follow a certain distribution.

x) Overlapping Sums Test: In this test a long sequence of random floats on $[0, 1)$ is generated. Thereafter an addition of sequences of 100 consecutive floats is performed. The sums should be normally distributed with characteristic mean and sigma.

xi) Runs Test: In this test a long sequence of random floats on $[0, 1)$ is generated to perform the ascending and descending runs tests. The counts should follow a certain distribution.

xii) The Craps Test: In this test the games of craps is played for 200000 times to obtain the wins and the number of throws per game. Each count should follow a certain distribution.

Cellular Automata [5, 6] is used to represent a dynamic mathematical model. A Cellular Automaton (pl. cellular automata, in short CA) is a discrete model studied in computability theory, mathematics, physics, complexity science, theoretical biology and microstructure modeling. It consists of a regular grid of cells, each in one of a finite number of states, such as "1" for ON and "0" for OFF. The framework might be in higher degree of dimensions. For each cell, a set of cells called its neighborhood (usually including the cell itself) is defined relative to the specified cell. For example, the neighborhood of a cell might be defined as the set of cells a distance of 2 or less from the cell. A primary state (time $t=0$) is selected by assigning a state for each cell. A new generation is created (advancing t by 1), according to some predetermined mathematical function that determines the new state of each cell in terms of the current state of the cell and the states of the cells in its neighborhood.

The simplest CA is one-dimensional with only two neighbors defined at the adjacent cells on either side of it. This combination forms a neighborhood of 3 cells, with $2^3=8$ possible patterns for this neighborhood. There are then $2^8=256$ possible rules. These 256 CAs are generally referred to by their Wolfram code. Wolfram code is a standard naming convention invented by Wolfram; it gives each rule a number from 0 to 255. A number of papers have analyzed and compared these 256 CAs, either individually or collectively.

CA has a significant role for computation as it is easy to implement through hardware components. Following Figure 1 shows a rough outline of CA implementation and Figure 2 describes the hardware implementation of CA through flip-flop.

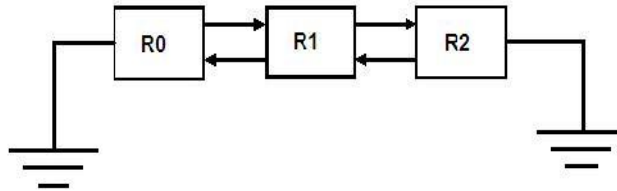


Figure 1. Implementation of 3-cell Null Boundary CA

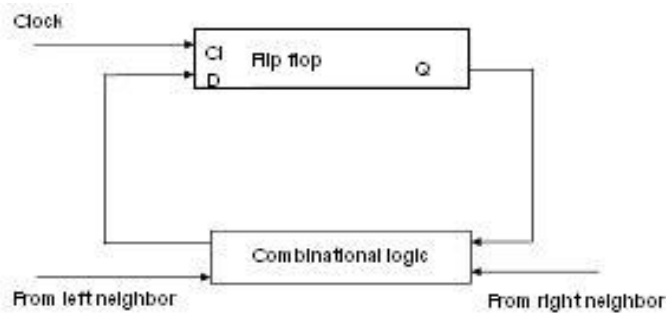


Figure 2. Hardware implementation of CA

Rest of the paper is organized as follows: Section 2 briefs about the related work; Section 3 describes proposed work; Experimental observations and result analysis are shown in Section 4 and Conclusion is reflected in Section 5.

2. RELATED WORK

Several methods have been implemented to generate good quality random numbers. Some novel efforts have been established to generate good quality random numbers [5], [7-19]. Some important efforts have also been made to generate pseudo-random numbers using CA. It has been already established that the pseudo-random numbers generated using maximum length CA exhibits a high degree of randomness.

The most common way to generate pseudo-random number is to use a combination of “randomize” and “rand” functions. Random patterns can be achieved based on the following recursive PRNG Equation 3.

$$X_{n+1} = P_1 X_n + P_2 (\text{Mod}N) \quad \text{.....(3)}$$

Here P_1, P_2 are prime numbers, N is range for random numbers. X_n is calculated recursively using the base value X_0 . X_0 is termed as seed and it is a prime number.

If X_0 (seed) is same all time or selected in any deterministic way, then it yields pseudo-random number [2].

It has been observed that in the generation process of pseudo-random pattern using Maximum Length CA, only a large cycle is responsible for yielding the pseudo random patterns. If a single cycle with larger numbers of states is used to generate the random patterns, the associated cost would be increased. All the cost associated with the generation process of random numbers; i.e., time, design and searching costs are having higher values as the complexity is directly proportional to the number of cell sizes used in the cycle. So, it is convenient to reduce the cycle without affecting the randomness quality of the generated random patterns for reducing these associated costs.

For the generation process of high quality of pseudo-random numbers, several works have been established with the uses of Cellular Automata. The research on Maximum Length CA reveals that with the increased number of cells, the resulting maximum length cycle includes the maximum degree of randomness. In Maximum Length CA, prohibited pattern set (PPS) is excluded from the cycle for achieving higher degree of randomness. In Maximum Length CA, the generation procedure of random patterns is complex and costly. In this methodology, it is mandatory to keep track of PPS in maximum length cycle and to exclude the portion in resulting maximum length cycle. Thus it makes the generation procedure complex. The associated complexity costs i.e., time complexity, design complexity and searching complexity are high along with this procedure. The quality of randomness in generated integer pattern is quite high [1]. Therefore, an alternative easy to implement generation methodology of random numbers would be much more beneficial which deal with the flaws of Maximum Length CA. In our proposed methodology, we have proposed a system that will be dealing with the flaws of the Maximum Length CA. Thus the proposed methodology should be more cost effective in terms of design complexity, time complexity and searching complexity though the generated random integer patterns are having the quality of randomness, is at least equal to the quality of randomness achieved by Maximum Length CA. The detailed discussion of the proposed methodology is following in Section 3.

3. PROPOSED WORK

A cost effective and simpler design methodology always should be beneficial for the generation of random integer patterns. The cost efficiency refers to the space, time, searching and design complexity of any involved algorithm. In the cost optimization generation methodology of the universal random pattern, a one dimensional Equal Length Cellular Automata, is proposed over the existing Maximum Length Cellular Automata for random pattern generation.

In the proposed methodology, a new approach has been proposed to achieve high degree of randomization in terms of better cost optimization with respect to all the concerning complexities mentioned earlier. The resulting methodology should also be very much convenient in hardware

implementation. The quality of randomness of the generated pattern by the proposed methodology is compared through Diehard Tests with Maximum Length CA random number generator. The conclusion is made about the quality of randomness of any data set based on the number of Diehard Tests passed. Several data set are analyzed through Diehard, is described in Table 4 to compare for randomness quality of different random number generators.

In the proposed PRNG, we propose an equal length one dimensional CA over the maximum length CA. In our work, we propose the decomposition of the larger cycle of maximum length CA, into more relevant sub-cycles such that our concerning complexities cost can be reduced as well as the fault coverage can be more flexible than of previously established maximum length CA. The proposed PRNG system is described in Figure 3 [1].

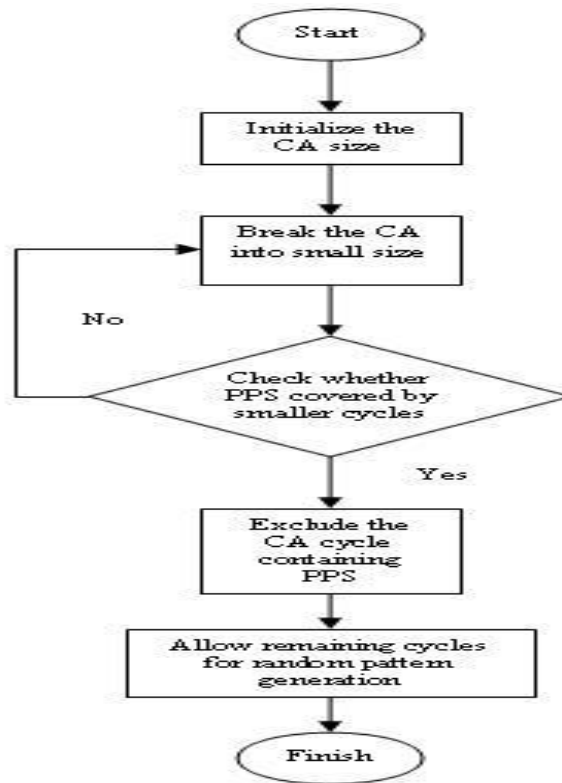


Figure 3. Flowchart of proposed system

A new mathematical approach has been proposed to achieve the same amount of randomization in less cost with respect to various complexities and hardware implementation, according to the flowchart of the proposed system as mentioned in Figure 3.

The resulting larger cycle of maximum length CA is divided into two or more equal sub-cycles instead of taking the full cycle of maximum length. These sub-cycles are capable to generate random patterns as equivalent to the randomness quality achieved from maximum length CA. The following Algorithm 1 has been used for the decomposition of an n-cell maximum length CA [1].

Algorithm 1: Cycle Decomposition

Input: CA size (n), PPS Set

Output: m-length cycles excluding PPS

Step 1: Start

Step 2: Initialize the number of n-cell CA to generate random patterns using n-cell CA

*Step 3: Decompose the cell number (n) into two equal numbers (m) such that $n=2*m$*

Step 4: Check each PPS whether it belongs to a single smaller cycle CA

Step 5: Repeat Step 3 and Step 4 until each PPS belongs to separate smaller cycles

Step 6: Allow m-length cycles of n-cell CA after excluding all the PPS containing cycles

Step 7: Stop

In the proposed methodology, the primary concern is to reduce the PPS. Therefore it has been taken care of that the occurrence of every PPS must be completed in some of the smaller sub-cycles, such that by eliminating all those smaller cycles, the remaining prohibited patterns free cycles can be allowed to generate random patterns. Thus, this methodology implies a better cost effectiveness approach. The proposed methodology thus simplifies the design complexity and empowers the searching complexity. The terminology design complexity refers to the implementation procedure for generation of random pattern and empowering searching complexity means the zero overhead for keeping track for PPS for random pattern generation. In comparison with an n-cell maximum length CA, more number of smaller cycles instead of one maximum length cycle should be used.

The maximum length CA cycle has been decomposed into smaller equal length cycles using Algorithm 1. The Algorithm 1 explains the decomposition procedure of maximum length CA.

Let us assume, for the n-length CA, the total number of states= 2^n .

We can generate this same amount of CA cells using Equation 4.

We have $2^n = 2*(2^m)$; i.e., at least 4 numbers of equal length cycles.

Thus 'm' is always less than 'n'.

Let one maximum length CA is divided into two sub-cycles.

Then, $2^n = (2^{m1} + 2^{m2})$

If $m1 = m2$

Then it becomes $2^n = 2 (2^{m1})$

Upon illustrating this model, let us assume the case where maximum length CA, $n=8$.

So, $2^8 = 2^7 + 2^7$ (i.e., two equal length of CA cycles)

$= 2^6 + 2^6 + 2^6 + 2^6$ (i.e., four equal length of CA cycles)

$= 2^5 + 2^5 + 2^5 + 2^5 + 2^5 + 2^5 + 2^5 + 2^5$ (i.e., eight equal length of CA cycles)

$= 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4 + 2^4$ (i.e., sixteen equal length of CA cycles)

The PPS is excluded from the cycle as per the procedure for generating the maximum random pattern in maximum length CA. In our procedure, the PPS can be totally removed as we are having more number of cycles for generation of random sequences. In proposed methodology, the cycles producing PPS can be removed from the generation of pseudo random numbers.

According to the proposed methodology, a CA size of $n=24$ might be decomposed into equal length smaller cycles instead of one maximum length circle. In this case it can be divided into 16 smaller cycles of length 20 or more. Let us assume that there exist nine number prohibited patterns as PPS. So in worst case, assuming every single prohibited pattern occurs in a single cycle, there exist $(16 - 10=6)$ number of 20 CA cycle to generate random patterns. The pattern generation on this scenario is followed as in Figure 4. Figure 4(a) shows one maximum length cycle with prohibited patters and on the hand Figure 4(b) shows 16 equal length smaller cycles where some the cycles contains prohibited set only. The following Figure 4 is based on Null Boundary 3 cell CA where the rules $\langle 90, 90, 150 \rangle$ have been applied. The PPS is denoted as $\{PS_0, PS_1, \dots, PS_9\}$.

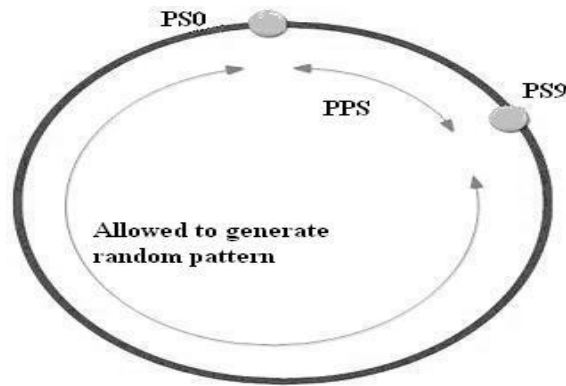


Figure 4(a).

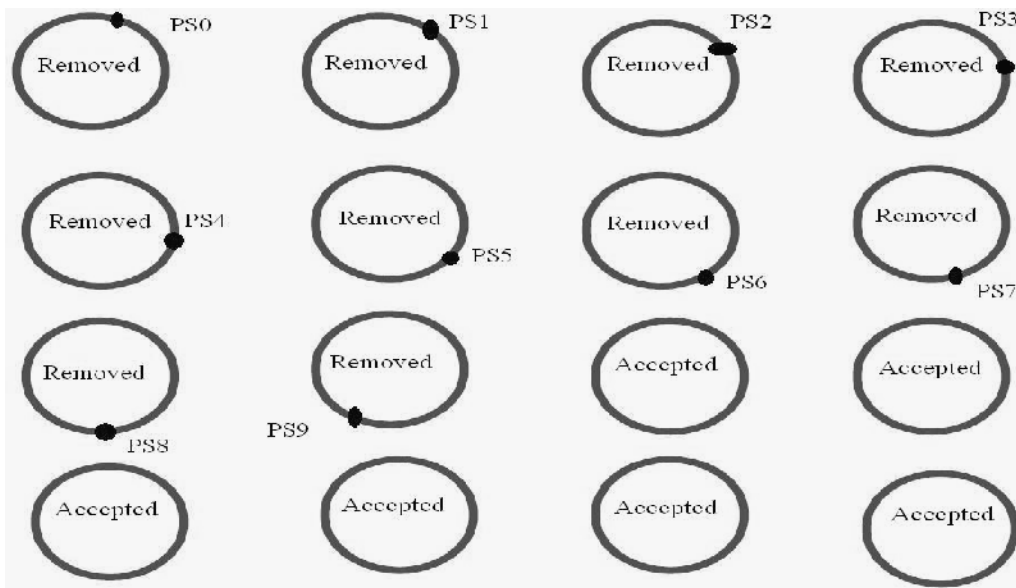


Figure 4(b).

Figure 4. Figure. 4(a). Maximum length CA Cycle for $n=24$ and Figure 4(b). Proposed equal length CA of smaller cycle size

Further a CA size of $n=64$ might be decomposed into equal length smaller cycles instead of one maximum length cycle. In this case it can be decomposed it into 128 smaller cycles of length 57 or more. Let us assume that there exist hundred number prohibited patterns as PPS. So in worst case, every single prohibited pattern will be in a single cycle, there exist $(128 - 100 = 28)$ number of cycles of length 57 to generate random patterns. The pattern generation is followed as in Figure 5. Figure 5(a) shows one maximum length cycle with prohibited patters and Figure 5(b) shows 100 equal length smaller cycles contains prohibited pattern only. Figure 5 is based on Null Boundary 3 cell CA where the rules $\langle 90, 90, 150 \rangle$ have been applied. The PPS is denoted as $\{PS_0, PS_1, \dots, PS_{99}\}$. Thus higher numbers of PPS are discarded from random pattern generating equal length CA cycles.

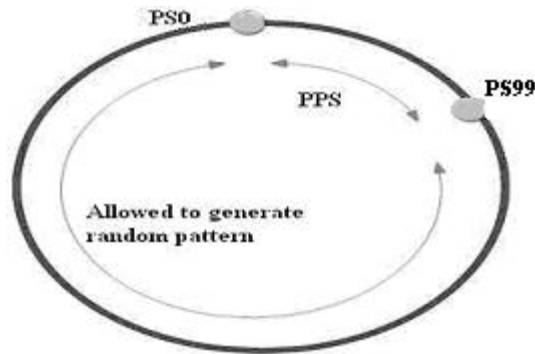


Figure 5(a).

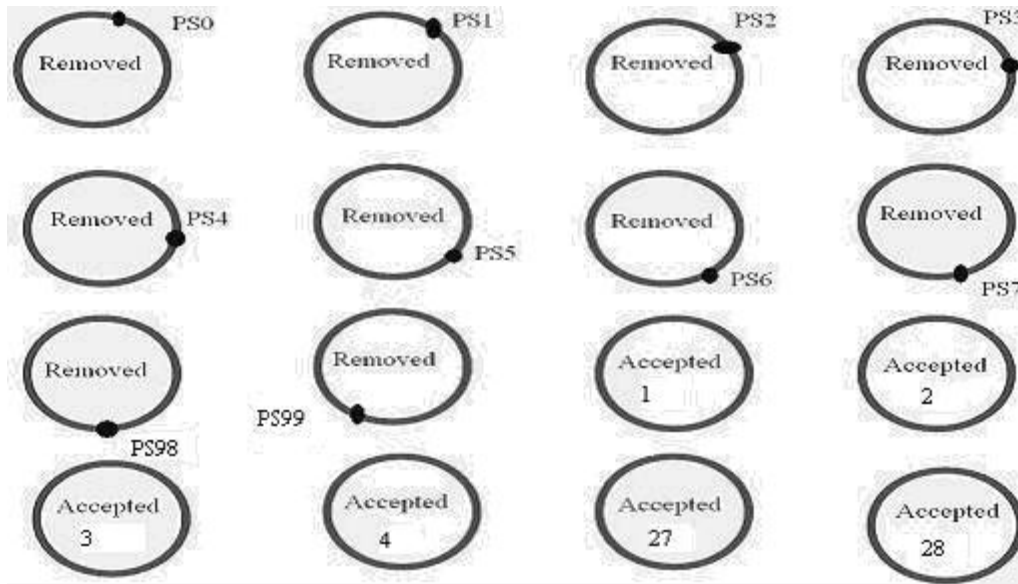


Figure 5(b).

Figure 5. Figure. 5(a). Maximum length CA Cycle for $n=64$ and Figure 5(b). Proposed equal length CA of smaller cycle size

In terms of fault coverage, the maximum length CA follows a procedure to handle the PPS if it exists in the larger cycle, which is responsible for generating random pattern. The fault coverage mechanism is as followed in following section.

In practice, the maximum length CA based random number generator produces the random patterns of integer using the cycle which might contain some of the prohibited patterns. In this scenario, the PPS are excluded form that maximum length CA cycle. Assume that there exist some n-numbers of PPS in the maximum length cycle. Let the PPS are $\{PS_0, \dots, PS_n\}$. For the exclusion purpose of these PPS from the cycle, the minimum length of arc (Arc_{min}) between the prohibited pattern PS_1 and PS_n should be measured so that we can utilize the remaining cycle arc i.e. effective arc (E_{arc}) for random number generation, which is typically free from PPS. This scenario can be efficiently visualized in Figure 6.

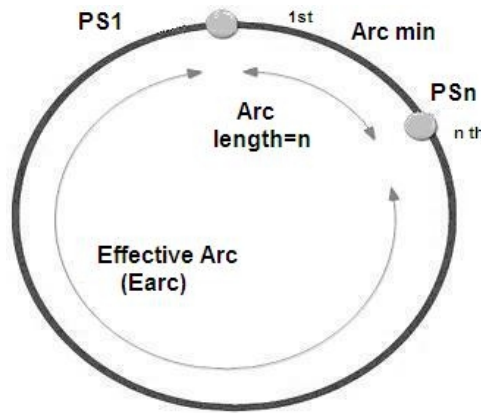


Figure 6. Typical Cycle structure of an n-cell Maximum Length CA

The Figure 6 is based on the facts that there exists total n-number of prohibited patterns with the following set: $PPS = \{PS_1, \dots, PS_n\}$. Figure 6 also explain the procedure to measure Arc_{min} and E_{arc} from an n-cell maximum length CA for fault coverage in random pattern generation.

Definition 1: *The minimum length of arc (Arc_{min}) is the minimum distance between the first and last prohibited pattern in an n-cell maximum length CA cycle.*

Definition 2: *The effective arc (E_{arc}) is the remaining arc length of an n-cell maximum length CA cycle which excludes Arc_{min} from the corresponding CA cycle of states and it is responsible for generating pseudo random patterns of integers.*

In our proposed methodology, we have discarded all the cycles containing any prohibited pattern. Some equal length cycles containing PPS are being discarded, yet we have enough cycles for generation of random patterns. The following Table-1 shows the comparison of fault coverage between maximum length CA and our proposed CA for CA cell size $n=9$. Let us assume that there exists 9 numbers of prohibited patterns. This procedure is described in Figure 7.

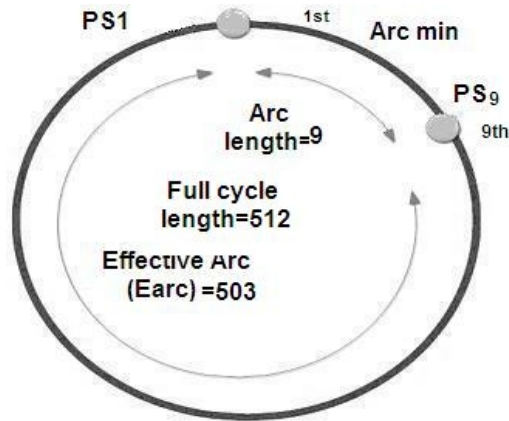


Figure 7. Typical cycle structure for maximum length CA cycle for n=9

4. EXPERIMENTAL OBSERVATIONS AND RESULT ANALYSIS

The PPS containing arc in random pattern generating cycle is excluded from the cycle as per the procedure for generating the maximum random pattern in maximum length CA. On the other hand, the PPS containing cycles are totally removed to generate the random sequences. There is no overhead to calculate E_{arc} and Arc_{min} in proposed equal length CA methodology. The following Table 1 compares the procedures of maximum length CA and equal length CA based random pattern generators.

Table 1: Comparison of fault coverage procedures

	Maximum length CA	Equal Length CA
PPS	9	9
E_{arc}	503	N/A
Arc_{min}	9	N/A

Table 1 reflects the advantages of our proposed methodology over maximum length CA for fault coverage in random pattern generation for CA size n=9. In proposed equal length CA, the cycles containing any prohibited pattern is excluded from generating random patterns.

The p-value analysis, generated for each sample data set by Diehard battery series test, helps to decide whether the test data set passes or fails the diehard test. Diehard returns the p-value, which should be uniform [0, 1) if the input file contains truly independent random bits. Those p-values are obtained by $p=F(x)$, where F is the assumed distribution of the sample random variable 'x', which is often normal. The value $p < 0.025$ or $p > 0.975$ means the RNG has "failed the test at the 0.05 level". Table 2 indicates the different tests of Diehard Test Suit.

Table 2: Diehard Tests

No.	Name of the test
1	Birthday Spacings
2	Overlapping Permutations
3	Ranks of 31x31 and 32x32 matrices
4	Ranks of 6x8 Matrices
5	The Bitstream Test
6	Monkey Tests OPSO,OQSO,DNA
7	Count the 1`s in a Stream of Bytes
8	Count the 1`s in Speci_c Bytes
9	Parking Lot Test
10	Minimum Distance Test
11	The 3DSpheres Test
12	The Squeeze Test
13	Overlapping Sums Test
14	Runs Test
15	The Craps Test

The comparison result in Table 3 shows the degree of randomness achieved by different random number generators in Diehard Tests.

Table 3: Performance result through Diehard for different n-values in maximum length CA-RNG

Diehard Test Number	Max-length CA n=8	Max-length CA n=10	Max-length CA n=23	Max-length CA n=64
1	Fail	Fail	Pass	Pass
2	Fail	Fail	Pass	Pass
3	Fail	Fail	Pass	Pass
4	Fail	Fail	Pass	Pass
5	Fail	Fail	Fail	Fail
6	Fail	Fail	Fail	Pass
7	Fail	Fail	Pass	Pass
8	Fail	Fail	Fail	Pass
9	Fail	Fail	Pass	Pass
10	Fail	Fail	Pass	Pass
11	Fail	Fail	Pass	Pass
12	Fail	Fail	Fail	Pass
13	Fail	Fail	Fail	Pass
14	Fail	Fail	Pass	Pass
15	Fail	Fail	Pass	Pass
Total Number of Diehard Test Passes	0	0	10	14

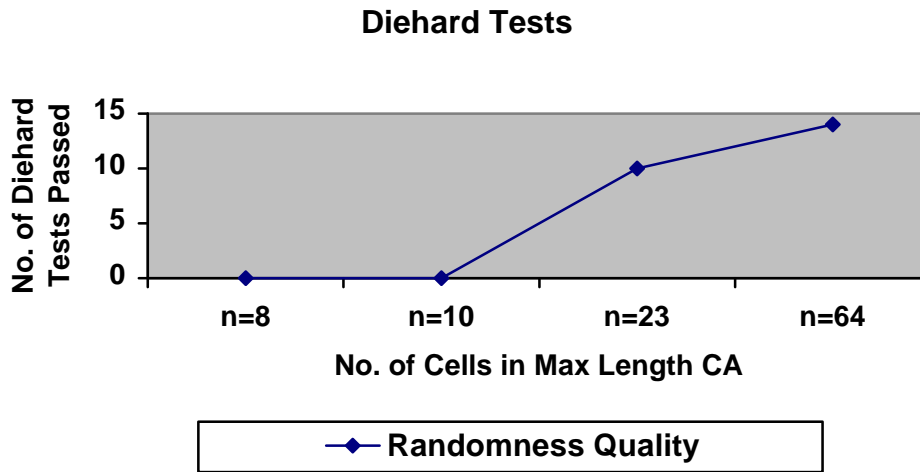


Figure 8. Randomness Quality for Maximum Length CA for different cell sizes (n)

Table 3 shows the degree of randomness achieved in Maximum Length CA for different cell sizes in terms of the total number of Diehard Tests passes. Figure 8 reflects this degree of randomness graphically.

Table 4: Performance result through Diehard for different CA random number generators

Diehard Test Number	Max-length CA		Equal length CA	
	n=23	n=64	n=23	n=64
1	Pass	Pass	Pass	Pass
2	Pass	Pass	Pass	Pass
3	Pass	Pass	Pass	Pass
4	Pass	Pass	Pass	Pass
5	Fail	Fail	Fail	Fail
6	Fail	Pass	Fail	Pass
7	Pass	Pass	Pass	Pass
8	Fail	Pass	Fail	Pass
9	Pass	Pass	Pass	Pass
10	Pass	Pass	Pass	Pass
11	Pass	Pass	Pass	Pass
12	Fail	Pass	Fail	Pass
13	Fail	Pass	Fail	Pass
14	Pass	Pass	Pass	Pass
15	Pass	Pass	Pass	Pass
Total Number of Diehard Test Passes	10	14	10	14

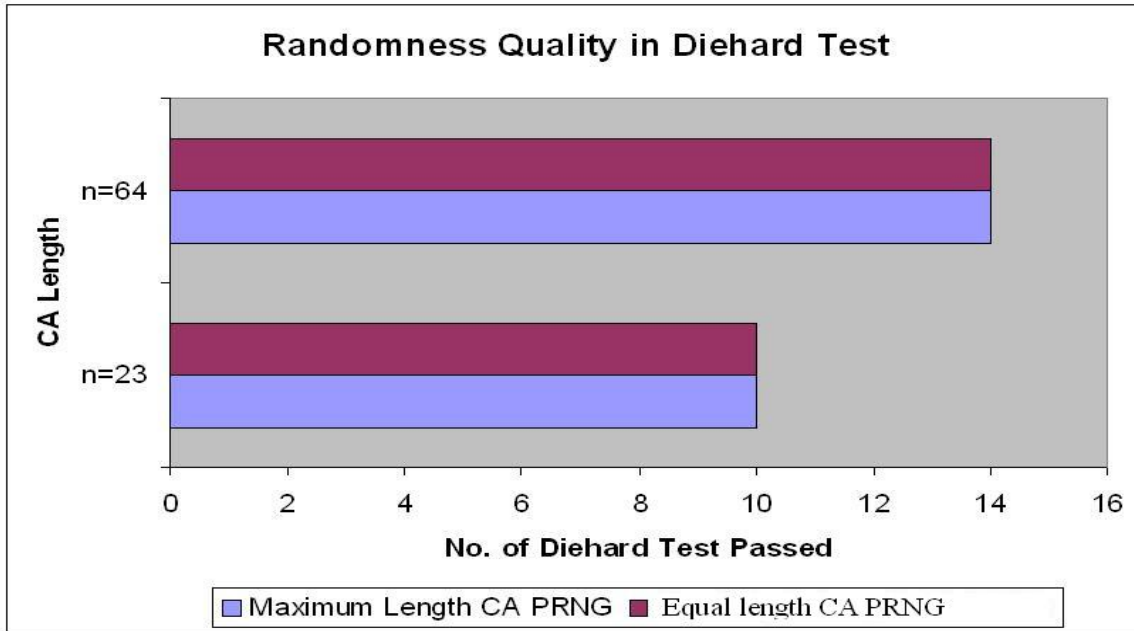


Figure 9. Diehard Performance Graph

The results obtained from Table 4 and Figure 9 ensures that the proposed random number generator, posses the maximum degree of randomization with a reference to the number of Diehard Test passes. This result is similar to the result achieved for maximum length CA based random number generator. The various complexities of these two CA based methodologies are reflected in Table 4.

Table 5: Complexity Comparison between Maximum Length CA and Proposed Methodology

Name of the Complexity	Comparison Result
Space	Same
Time	Slightly Improved
Design	Improved
Searching	Improved

Table 5 signifies that the space complexity is same for both procedures as total length of an n-cell CA is same for both the cases, but there are some changes in other complexities. Other complexities have been improved in case of our proposed methodology. The proposed methodology is allowed only to generate random patterns from smaller cycles that exclude PPS. The PPS exclusion feature from the main cycle, improves the design and searching complexities.

Results based on Table 4, Table 5 and Figure 8, it can be concluded that our proposed methodology can be used as a better source of random patterns as it posses the maximum degree of randomness with reference to all other established RNGs and it is less costly to implement.

5. CONCLUSION

The Diehard Test results show the quality of randomness achieved from the various samples of random data sets. The number of passes shows the quality of randomness achieved by particular method. Table 4 shows the randomness achieved from the random data sets produced through Equal Length CA is having the maximum randomness as it passes the maximum number of Diehard Tests which is same as for the maximum length CA. Hence this methodology is suitable for generating random sequences as the cost associated is much cheaper in terms of time complexity and hardware implementation. The Equal Length CA uses only the sub cycle which consists of lesser number of states than of maximum length CA. Thus it completes its full cycle in lesser time and requires lesser hardware implementation cost. The time complexity of this proposed methodology maintains a linear time complexity with increased number of cells in the CA. It is also convenient that the Equal Length CA random number generator is more flexible as it excludes the prohibited pattern set (PPS).

REFERENCES

- [1] Arnab Mitra, Anirban Kundu; (2012) Cost optimized Approach to Random Numbers in Cellular Automata; The Second International Conference on Computer Science, Engineering & Applications (ICCSEA); India.
- [2] S. Wolfram; Wolfram Mathematica Tutorial Collection: Random Number Generation.<http://www.wolfram.com/learningcenter/tutorialcollection/RandomNumberGeneration/RandomnumberGeneration.pdf>
- [3] Robert G. Brown. dieharder: A Random Number Test Suite, 2006a. <http://www.phy.duke.edu/~rgb/General/dieharder.php>. C program archive dieharder, version 1.4.24.
- [4] Dirk Eddelbuettel; Random: An R package for true random numbers;<http://dirk.eddelbuettel.com/bio/papers.html>
- [5] <http://www.random.org/>
- [6] S. Wolfram; (1986) Theory and Application of Cellular Automata; World Scientific.
- [7] Sukanta Das, Biplab K. Sikdar, P. Pal Chaudhuri; (2004) Characterization of Reachable/ Nonreachable Cellular Automata States; International Conference on Cellular Automata for Research and Industry; ACRI; Netherlands.
- [8] Sukanta Das, Anirban Kundu, Biplab K. Sikdar, P. Pal Chaudhuri; (2005) Design of Nonlinear CA Based TPG Without Prohibited Pattern Set In Linear Time; JOURNAL OF ELECTRICAL TESTING: Theory and Applications.
- [9] Biplab K. Sikdar, Sukanta Das, Samir Roy, Niloy Ganguly, Debesh K. Das; (2005) Cellular Automata Based Test Structures with Logic Folding; VLSI Design; India.
- [10] Sukanta Das, Haffizur Rahaman and Biplab K Sikdar; (2005) Cost Optimal Design of NonlinearCA Based PRPG for Test Applications; IEEE 14th Asian Test Symposium; India.
- [10] Sukanta Das, Debdas Dey, Subhayan Sen, Biplab K. Sikdar, Parimal Pal Chaudhuri; (2004) An efficient design of non-linear CA based PRPG for VLSI circuit testing; ASP-DAC; Japan.
- [12] Sukanta Das, Anirban Kundu, Biplab K. Sikdar; (2004) Nonlinear CA Based Design of Test Set Generator Targeting Pseudo-Random Pattern Resistant Faults; Asian Test Symposium; Taiwan.
- [13] Sukanta Das, Biplab K. Sikdar, Parimal Pal Chaudhuri; (2004) Nonlinear CA Based Scalable Design of On-Chip TPG for Multiple Cores; Asian Test Symposium; Taiwan.
- [14] Sukanta Das, Anirban Kundu, Subhayan Sen, Biplab K. Sikdar, Primal PalChaudhuri; (2003) Non-Linear Cellular Automata Based PRPG Design (Without Prohibited Pattern Set) In Linear Time Complexity; Asian Test Symposium; China.
- [15] Sukanta Das, Niloy Ganguly, Biplab K. Sikdar, Parimal Pal Chaudhuri; (2003) Design Of A Universal BIST (UBIST) Structure; VLSI Design; India.

- [16] Niloy Ganguly, Anindyasundar Nandi, Sukanta Das, Biplab K. Sikdar, Parimal Pal Chaudhuri; (2002) An Evolutionary Strategy To Design An On-Chip Test Pattern Generator Without Prohibited Pattern Set (PPS); Asian Test Symposium; Guam.
- [17] Peter D. Hortensius, Werner Pries, Howard C. Card; (1989) Cellular Automata based Pseudorandom Number generators for Built-In Self-Test; IEEE Transactions on Computer-Aided Design; VI. 8 No. 8.
- [18] Paul H. Bardell; (1990) Analysis of Cellular Automata Used as Pseudorandom pattern Generators; International Test Conference.
- [19] D. de la Guia Martinez, A. Peinado Doinguez; (1999) Pseudorandom number generation based on Nongroup Cellular Automata; IEEE 33rd Annual International Carnahan Conference on Security Technology.

Authors:

Mr. Arnab Mitra

Mr. Mitra is presently working as an Assistant Professor in the Department of CSE with Adamas Institute of Technology-W.B. (India). He is pursuing his research work as a Research Assistant at Innovation Research Lab, under guidance of Dr. Anirban Kundu. His broad research interests are Artificial Intelligence and applications of Cellular Automata.



Dr. Anirban Kundu

Dr. Kundu is a Post Doctoral Research Fellow at Kuang-Chi Institute of Advanced Technology, P.R. China. He has obtained his Ph.D. from Jadavpur University (India). In India he is associated with Netaji Subhash Engineering College as an Assistant Professor & Head of the Department (IT). He is also associated with Innovation Research Lab-W.B. (India) as an Advisor & Consultant. His research interests include Search Engine Oriented Indexing, Ranking, Prediction, Web Page Classification with essence of Cellular Automata. He is also interested in Multi-Agent based system design, Fuzzy controlled systems and Cloud Computing. He has authored more than 40 journal and conference papers.

