Pseudorandom Pattern Generation Using 3-State Cellular Automata

Kamalika Bhattacharjee(✉), Dipanjyoti Paul, and Sukanta Das

Department of Information Technology,
Indian Institute of Engineering Science and Technology,
Shibpur 711103, West Bengal, India
kamalika.it@gmail.com, dipanjyotipaul@gmail.com, sukanta@it.iiests.ac.in

Abstract. This paper investigates the potentiality of pseudo-random pattern generation of 1-dimensional 3-state cellular automata (CAs). Here, a pattern represents configuration of a CA of length n. We have identified 805 CAs which have great potentiality to act as pseudorandom pattern generator (PRPG).

Keywords: Pseudo-random pattern generator (PRPG) · 3-state Cellular Automata(CAs) · Fixed-Point Graph (FPG) · Diehard · TestU01

1 Introduction

This paper investigates the (pseudo) randomness behaviour of 1-dimensional 3-neighborhood (that is, nearest neighbor) CAs having 3 states per cell with periodic boundary condition. A list of works already exists in literature using binary (2-state) 3-neighborhood CAs as source of randomness [2,3,8]. However, the randomness of binary CAs with increased neighborhood dependency is not known, but it is well-known after Smith that, a CA with higher neighborhood dependency can always be emulated by another CA with lesser, say 3-neighborhood dependency [7]. In this paper, we have selected 1-D 3-neighborhood finite 3-state CAs, and checked the randomness of the patterns which are configurations of the CAs.

As the rule space of 3-state CAs is huge, we have developed greedy strategies (Sect. 3) and some theories to filter out the potential CAs (Sect. 4.1). These CAs are, however, further tested for randomness using *Diehard* battery of tests [5] as the testbed (Sect. 4.2). Finally, we have got 805 CAs which are verified and claimed to be excellent source of randomness (Sect. 5). We have also tested some existing PRPGs on the same testbed Diehard with same specifications as our PRPGs and compared the result. It is seen that, our PRPGs beat the existing CAs based PRPGs for $n = 15$ (Sect. 5.3).

This research is partially supported by Innovation in Science Pursuit for Inspired Research (INSPIRE) under Dept. of Science and Technology, Govt. of India.

S. El Yacoubi et al. (Eds.): ACRI 2016, LNCS 9863, pp. 3–13, 2016.
DOI: 10.1007/978-3-319-44365-2_1

2 Background

Here, we have considered 1-D 3-neighborhood 3-state CAs, where each cell can take any of the states $S = \{0, 1, 2\}$. The local rules are expressed by a tabular form (see Table 1), where the table contains entries for the combinations of left (x), self (y) and right (z) neighbors of a cell. Each of these combinations with respect to the value $R(x, y, z)$, where R is the local rule, is termed as *Rule Min Term (RMT)*(r) and is generally represented by its decimal equivalent. In this paper, $R(x, y, z) \equiv R[r]$. The number of RMTs of a 3-state CA rule is $3^3 = 27$. We represent R by the values of $R[r]$ with $R[0]$ as the right most digit.

Table 1. Rules of 3-state CAs. Here, PS is present state and NS is next state

P.S.	222	221	220	212	211	210	202	201	200	122	121	120	112	111	110	102	101	100	022	021	020	012	011	010	002	001	000
RMT	(26)	(25)	(24)	(23)	(22)	(21)	(20)	(19)	(18)	(17)	(16)	(15)	(14)	(13)	(12)	(11)	(10)	(9)	(8)	(7)	(6)	(5)	(4)	(3)	(2)	(1)	(0)
	2	1	1	2	1	2	1	1	2	0	2	0	0	0	0	0	2	0	1	0	2	1	2	1	2	0	1
N.S.	1	0	2	0	1	2	1	0	2	0	1	2	1	0	2	1	0	2	0	2	1	0	2	1	0	1	2
	1	1	2	2	2	1	0	1	0	1	1	2	2	2	1	0	0	0	1	1	2	2	2	1	0	0	0

Definition 1. *An RMT r of a rule R is said to be self-replicating, if $R[r] = y$, where $r = x \times 3^2 + y \times 3 + z$ and $x, y, z \in \{0, 1, 2\}$.*

For example, for the CA 102012102012102102021021012 (4^{th} row of Table 1), the self-replicating RMTs are RMTs $2(002), 3(010), 7(021), 10(101), 14(112), 15(120), 19(201), 22(211)$ and $24(220)$. Moreover, a rule is called *balanced* if it contains equal number of RMTs for each of the three possible states for that CA [1]. In Table 1, the rule of 5^{th} row is unbalanced, whereas the rest rules are balanced.

A configuration of a CA can also be represented as a sequence of RMTs, called *RMT sequence*. For example, let 0120 be a configuration of a 4-cell CA. Then the RMT sequence corresponding this configuration is $\langle 1, 5, 15, 18\rangle$. To get a RMT sequence, we consider an imaginary window of length 3 which slides over the configuration, one cell right at a time. The decimal value corresponding to this 3-cell window is i^{th} RMT in the sequence. Note that, in a RMT sequence, only a specific set of RMTs can be selected after a RMT. For example, after RMT 1(001), either of the RMTs 3(010), 4(011) or 5(012) can be present in a RMT Sequence. For any RMT, the set of 3 RMTs from which the next RMT for the RMT sequence is selected, are named as *sibling* RMTs. Similarly, a set of 3 RMTs always results in creating the same sibling RMT set, which are termed as *equivalent* RMTs.

There are $3^2 = 9$ sets of equivalent RMTs and 9 sets of sibling RMTs. We represent $Equi_i$ as a set of RMTs that contains RMT i and all of its equivalent RMTs. That is, $Equi_i = \{i, 9 + i, 18 + i\}$, where $0 \leq i \leq 8$. Similarly, $Sibl_j$ represents a set of sibling RMTs where $Sibl_j = \{3j, 3j + 1, 3j + 2\}$ $(0 \leq j \leq 8)$. Table 2 shows the relationship among the RMTs of 3-state CAs [1]. If in a RMT sequence, a RMT is chosen from $Equi_i$, then the next RMT in the sequence must be chosen from $Sibl_i$.

Table 2. Relations among the RMTs for 3-State CA

Equivalent set			Sibling set		
#Set	Equivalent RMTs	Decimal equivalent	#Set	Sibling RMTs	Decimal equivalent
$Equi_0$	000, 100, 200	0, 9, 18	$Sibl_0$	000, 001, 002	0, 1, 2
$Equi_1$	001, 101, 201	1, 10, 19	$Sibl_1$	010, 011, 012	3, 4, 5
$Equi_2$	002, 102, 202	2, 11, 20	$Sibl_2$	020, 021, 022	6, 7, 8
$Equi_3$	010, 110, 210	3, 12, 21	$Sibl_3$	100, 101, 102	9, 10, 11
$Equi_4$	011, 111, 211	4, 13, 22	$Sibl_4$	110, 111, 112	12, 13, 14
$Equi_5$	012, 112, 212	5, 14, 23	$Sibl_5$	120, 121, 122	15, 16, 17
$Equi_6$	020, 120, 220	6, 15, 24	$Sibl_6$	200, 201, 202	18, 19, 20
$Equi_7$	021, 121, 221	7, 16, 25	$Sibl_7$	210, 211, 212	21, 22, 23
$Equi_8$	022, 122, 222	8, 17, 26	$Sibl_8$	220, 221, 222	24, 25, 26

Definition 2. *A fixed-point attractor is a configuration of CA, for which the next configuration is the configuration itself. That means, if a CA reaches to a fixed-point attractor, then it remains at that particular configuration forever.*

For example, 0^n is a fixed point attractor of the CA 112221010112221010112221000 with n cells, $n \geq 3$. Generally, one state x is called a quiescent state, when $R(x, x, x) = x$, where R is the rule of the CA. If a CA has a quiescent state at x, then there exists a fixed point attractor at x^n, for any cell length n.

3 Cellular Automata as PRPG

CAs are considered to be a good source of randomness. However, for 3-state CAs, total number of rules is $3^{3^3} = 7.625597485 \times 10^{12}$, which is a huge number for exhaustive testing. So, we concentrate on finding some properties of a CA, which make it a candidate to have good randomness quality. Following is the first property:

Property 1: The randomness of balanced rules, in general, are better than that of unbalanced rule.

If a CA rule is unbalanced, then at least one of the states 0/1/2 has more presence in the rule than the other state(s). Therefore, during evolution from an arbitrary configuration, the CA will be biased towards the state(s) having more presence in the unbalanced rule. The number of balanced 3-state CA rules is $\frac{3^3!}{(3^2!)^3} = 227873431500$, which is also a big number. However, in a random system, information on a localized change eventually flows through the whole system. In a CA based random system, a small change at a local cell by a local rule eventually propagates throughout it, effecting globally. Hence, to have good randomness, the CA must have a sufficient rate of information transmission, so that it does not become stable in a finite time. But, only balancedness does not ensure the flow of information on left or right side in CA. Therefore, we

take a greedy approach to choose the balanced rules which have a constant rate of information transmission on at least right direction. Success of this scheme, however, remains on how efficiently we are choosing the balanced rules.

Please recall that, the set $\{xy0, xy1, xy2\}$ represents the set of sibling RMTs, where $x, y \in \{0, 1, 2\}$ (see Table 2). So, our strategy is to choose the CA rules, where the sibling RMTs have different next state values, which implies that, there is a constant rate of information transmission towards the right side.

STRATEGY: *Pick up the balanced rules in which the RMTs of a sibling set have the different next state values, that is, no two RMTs of* $Sibl_i$ $(0 \leq i \leq 8)$ *have same next state value* [1].

There are $(3!)^{3^2} = 10077696$ balanced rules that can be selected as candidates following this strategy. These CAs are potential nominees to be good PRPGs. Moreover, we also want to consider the flow of information to the left direction for these rules. This implies the RMTs of $Equi_i$, $0 \leq i \leq 8$, should have different states. We define an index, termed as *equiRMTCount*, which measures the amount of information flow towards left direction by observing the equivalent RMT sets.

Definition 3. *The* equiRMTCount *is the cumulative sum of the number of RMTs in* $Equi_i$ $(0 \leq i \leq 8)$*, which have the same next state value. That means,* equiRMTCount *is increased by* 1 *if* $R[r_1] = R[r_2] \neq R[r_3]$ *(or* $R[r_1] \neq R[r_2] = R[r_3]$*, or* $R[r_1] = R[r_3] \neq R[r_2]$*) and it is increased by* 2 *if* $R[r_1] = R[r_2] = R[r_3]$*, where* $r_1, r_2, r_3 \in Equi_i$*,* $\forall i$ *and* R *is the rule of the CA.*

Table 3 shows an example of finding *equiRMTCount* of a rule R. First column notes the set number, whereas, next 3 pairs of columns shows the RMTs and corresponding next state values. RMTs 0, 9 and 18 of the rule have same next state values, so, *equiRMTCount* is 2 for $Equi_0$. Similarly, RMTs 10 and 19 of the rule have same state value, so $Equi_1$ gives 1 increment to *equiRMTCount*.

This index gives idea about the information flow in the left direction. For example, for a CA 012012021012012021012012021, *equiRMTCount*= 18, that is, no information flow at the left direction. However, for the rule of Table 3, *equiRMTCount* = 8, that means, there is at maximum $\frac{18-8}{18} = 55.56\,\%$ chance of information travel in the left direction.

Note that, our requirement is to select the rules which have a constant information transmission in the right direction, as well as, at least a certain rate of information transmission in the left side. This is to ensure that, a small ripple in a local cell propagate in both sides. To validate our argument, an experiment is constructed, where some rules are arbitrarily chosen and tested on Diehard testbed; *equiRMTCount* for these rules is also calculated. Figure 1 shows the plot of these rules. In this figure, X-axis represents *equiRMTCount*, Y-axis the number of randomness tests passed and the count of rules with any particular *equiRMTCount* value that passes any number of tests is shown in Z-axis.

This figure clearly shows that, if *equiRMTCount* value is high (≥ 15), then practically there are insignificant number of rules that passes any randomness tests. So, in our work, we have chosen the rules following STRATEGY which

Table 3. Calculation of *equiRMTCount* for the CA 10201210201210210202102101 2(R)

#Set	Equivalent RMTs						Match count
	RMT i	$R[i]$	RMT i	$R[i]$	RMT i	$R[i]$	
$Equi_0$	0	2	9	2	18	2	2
$Equi_1$	1	1	10	0	19	0	1
$Equi_2$	2	0	11	1	20	1	1
$Equi_3$	3	1	12	2	21	2	1
$Equi_4$	4	2	13	0	22	1	0
$Equi_5$	5	0	14	1	23	0	1
$Equi_6$	6	1	15	2	24	2	1
$Equi_7$	7	2	16	1	25	0	0
$Equi_8$	8	0	17	0	26	1	1
equiRMTCount =							8

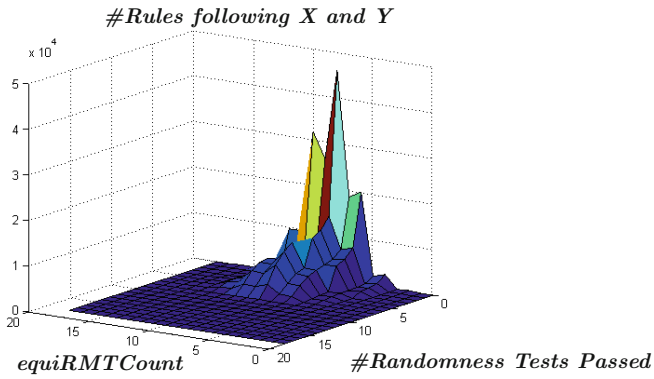


Fig. 1. Test Result of 61249 arbitrarily selected CAs

have *equiRMTCount* ≤ 14. There are 10067760 rules that pass this condition. In the next section we define some filtering criteria on these rules based on the inherent structure of the CAs and experiment to select the potential PRPGs.

4 Two-Step Filtering

In this section, the set of 10067760 rules are first filtered based on some theories developed in the following subsection and then, on these rules randomness tests are applied repeatedly for different seeds.

4.1 Theoritical Filtering

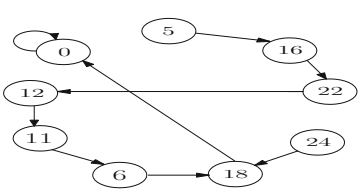
Here, we have worked with the CAs, which have at least one quiescent state. Recall that, for a quiescent state, a fixed-point attractor in generated in the CA.

Now, in the configuration transition diagram of CAs, one can observe that, a fixed-point attractor [9] may be associated with long chains of configurations, or small chains, or it may be isolated and most of the configurations are part of a long cycle. Moreover, a CA with long cycle length (or very long chains) have better randomness property than that of the CAs with small cycles (or, small chains). This is because, longer the cycle length, lesser the number of times any state in the cycle get repeated - implying better randomness. Note here that, max-length CAs [2] has maximum possible cycle length; but for classical CAs, there is no known existence of max-length CA. However, a classical CA can have long cycle only when it does not have a tendency to converge to the fixed-point attractor. Therefore, identifying the fixed-point attractor and its connection to other configurations is important for selecting the CAs having good randomness property. We now define a graph, termed as fixed-Point graph (FPG) which helps to identify any fixed-point attractor and its nature in the CA. This graph was introduced in [6] for asynchronous CAs.

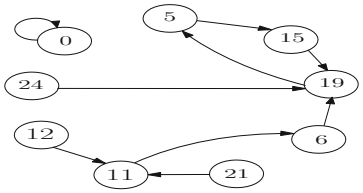
Fixed-Point Graph: The fixed-point graph (FPG) is a directed graph, where the nodes represents the self-replicating RMTs. To draw the FPG for a CA, first a forest is formed with the self-replicating RMTs of the CA as the individual nodes. Now, there is a directed edge from vertex u to vertex v, if $u \in Equi_k \Rightarrow v \in Sibl_k$, $0 \leq k \leq 8$, for any u, v (see Table 2). For example, if RMTs 1, 3 and 9 are self-replicating for a CA, then we can draw directed edges from RMT 1 to RMT 3, RMT 3 to RMT 9 and RMT 9 to RMT 1. But we can not draw directed edge from RMT 1 to RMT 9, as RMT $1 \in Equi_1$, but RMT $9 \notin Sibl_1$.

Example 1. This example illustrates the procedure of drawing the FPG for the CA 10201221012002102101210212 0. First, the self-replicating RMTs for this CA, i.e. the RMTs $0, 5, 6, 11, 12, 16, 18, 22$ and 24 are drawn as individual vertices. Now, we start from vertex 0, the minimum of the RMTs, as the first vertex. RMT $0 \in Equi_0$ and the sibling RMTs from RMT 0 are $Sibl_0 = \{0, 1, 2\}$. However, only RMT 0 is a vertex in this graph, so, a self loop is drawn to vertex 0. The next vertex is vertex 5. Now, RMT $5 \in Equi_5$, and from RMT 5, the next RMTs are $Sibl_5 = \{15, 16, 17\}$. Among these RMT 16 is a vertex, so, a directed edge is drawn from vertex 5 to vertex 16. Similarly, from vertex 16, we draw an edge to vertex 22, from vertex 22 to vertex 12, vertex 12 to vertex 11, vertex 11 to vertex 6, vertex 6 to vertex 18 and vertex 18 to vertex 0. Finally, from vertex 24, directed edge is drawn to vertex 18 completing the graph. Figure 2a shows the fixed-point graph for this CA.

Every fixed-point attractor in a CA can be identified easily by using this graph. To identify a fixed-point, we start with any vertex in the graph. Now, if this vertex can be reached again by traversing a sequence of vertices in the graph, then the RMT sequence corresponding to this traversal represents a fixed-point attractor. That means, if there is a loop of length l in the FPG, then the RMT sequence corresponding to this loop portrays a fixed-point attractor for the CA when cell length is equal to multiples of l.



(a) 10201221012002102101210212 0



(b) 10202120110202102012102120

Fig. 2. Fixed-point graphs (FPGs) for CAs

Example 2. Figure 2b shows the fixed-point graph for the CA 10202120110202102012102120. In this graph, vertex 0 has self-loop. So, the configuration 0^n represents a fixed-point attractor for any values of n. Apart from that, starting with vertex 5, this vertex can be reached by traversing the edges connecting vertices 5 to 15, 15 to 19 and 19 to 5. Therefore, the RMT sequence $\langle 5, 15, 19\rangle$ represents the fixed-point attractor $(120)^n$ or $120120\cdots120$ for the CA, when n is multiple of 3. However, for the CA of Fig. 2a, there is no other loop in the graph, except the self-loop at vertex 0. Therefore, 0^n is the only fixed-point attractor for the CA 10201221012002102101210212 0 for any values of n.

Note that, for the CAs following this strategy, there are 9 vertices in the FPGs and one RMT from each of $Sibl_i$, $0 \leq i \leq 8$ forms a vertex in the graph. However, our target is to get the CAs which have long cycles. Nevertheless, we define some conditions for filtering these CAs.

Filtering Conditions: We select those CAs as our candidates for experiment with battery of randomness tests, which have one and only one fixed-point attractor. To achieve this, the following two conditions are applied on the CAs–

(a) Only 1 **quiescent state:** Here, we consider the CAs, which have only one quiescent state. For this quiescent state, a fixed-point attractor is created in the CA. For 3-state CAs, the RMTs for quiescent state are RMT 0(000) for 0 as the quiescent state, RMT 13(111) for 1 as the quiescent state and RMT 26(222) for 2 as the quiescent state. So, we select the CAs which have 0 at $R[0]$, and 0/2 at $R[13]$ & 0/1 at $R[26]$, for providing the quiescent state 0. Similarly, the condition for only quiescent state at 1 (respectively, 2) is $R[13] = 1$ (respectively, $R[26] = 2$) and $R[0] \neq 0$ & $R[26] \neq 2$ (respectively $R[0] \neq 0$ & $R[13] \neq 1$), where $R[i]$ implies the RMT i of rule R. The following Table 4 gives some examples.

(b) No other fixed-point attractor: To ensure that the fixed-point attractor due to the quiescent state is the only fixed-point attractor in the CA, the fixed-point graph (FPG) for that CA is used. Recall that, any loop in this graph represents a fixed-point attractor for the CA. However, as the CAs fulfill the above condition, so, the fixed-point graphs have a self-loop at either of RMT 0, RMT 13 or RMT 26. If apart from this self loop, there is any other loop in the

Table 4. Some 3-state CAs having only one fixed-point

Fixed-point	Rules	Condition
0	102012210120021021012102120	$R[0] = 0$, $R[13] = 2$, $R[26] = 1$
0	012012120012021012012012210	$R[0] = 0$, $R[13] = 2$, $R[26] = 0$
1	012012102201012210012012102	$R[0] = 2$, $R[13] = 1$, $R[26] = 0$
2	201210021102102201210012012	$R[0] = 2$, $R[13] = 0$, $R[26] = 2$

FPG, then it implies a RMT sequence, that is a valid configuration depicting another fixed-point attractor. In this case, we reject the CA. For example, in the FPG for the CA 012012120012021012012012210 (Fig. 2a), there is only one fixed-point at 0^n for the quiescent state 0. So, this CA satisfies the filtering conditions. However, the FPG for the CA 102021201102021012012102120 (Fig. 2b) has another fixed-point attractor apart from the same for the quiescent state. So, this CA is rejected.

Among the 10067760 CA rules from the strategy, we get 1117008 rules which satisfy these theoretical filtering conditions. Now, there is greater chance of getting long cycle, if the fixed-point is isolated and not reachable from any other state. In the next subsection, we assure this through experiment and apply randomness tests on the selected rules.

4.2 Experimental Filtering

Some more rules can be screened out, if the possibility of reaching the trivial configurations from other configuration is considered, that is, if the trivial configurations are isolated and not connected with other non-trivial configurations.

Non-reachable Trivial Configuration: If any of the trivial configuration (0^n, 1^n or 2^n) is reachable from other non-trivial configuration, and the trivial configuration is associated with a fixed-point attractor, then there is a tendency to converge to that fixed-point attractor from any non-trivial configuration. In this case, the fixed-point attractor is not isolated and there is a chance of getting small cycles. This is an undesirable situation which weakens randomness property of the corresponding CA. Note that, this behavior is cell dependent, and relates with the length of the loop in the fixed-point graph for the CA. So, to avoid those rules, an experiment is conducted, where for each rule, it is checked whether the trivial configuration is reachable from the standard non-trivial configuration (all cells are 0, except the middle cell which is 2, i.e. 0^k20^k, $k = \lfloor \frac{n}{2} \rfloor$), similar to [8]. If the trivial configurations are reachable and has a fixed-point attractor, then the rule is discarded. This experiment is repeated on each of the 1117008 rules for the cell length n, $5 \leq n \leq 15$. We have found 637406 rules which have reachable trivial configuration. Therefore, the working rule set is of size 479602 on which the randomness tests are performed. We have used Diehard battery of tests as the initial testbed.

Filtering with Diehard Battery of Tests: Although from the theoretical development, 479602 rules are selected as the working set of candidates for PRPG with 3-state CA using the strategy, but, all of these rules may not be good for every circumstances demanding randomness. Therefore, rigorous and exhaustive randomness testing is applied on these rules to get the set of best rules. Note that, a rule is part of this best set only when it passes a minimum number of tests on every initial condition.

Wolfram in [8] showed that the binary CA with rule 30(00011110) is a good source of randomness. However, in that paper, he considered the randomness of the vertical sequence generated by the middle cell for a certain number of time stamps. But, for our CA, we have considered the randomness of the pattern generated in the configuration of a n-cell CA, where each cell can take any of the states $\{0, 1, 2\}$. Therefore, for our CA based PRPGs, finding n as minimum as possible is very important.

All the 479602 rules are tested with Diehard for arbitrary initial configuration as well as fixed initial configuration (i.e. 0^k20^k, $k = \lfloor \frac{n}{2} \rfloor$). By experimentation, we have got the minimum cell length for which the PRPG beats other existing CA based PRPGs is $n = 15$. So, for testing each CA, n is taken as 15 uniformly.

At each time, the CAs are tested with Diehard with random initial configuration and only the good CAs are put to test again. Here, a CA is considered *good*, when it passes at least 7 tests out of the 15 tests of Diehard. We have repeated this experiment several times. The rules, which have passed the minimum tests (that is, at least 7 tests) in all these runs, are selected to be potential PRPGs for any seed or initial condition. We have got 805 such 3-state CA rules. Table 5 gives some of these rules.

5 Verification of Result

In this section, we verify our final set of rules of size 805 to confirm their competency as excellent PRPGs. To reaffirm that, we have tested these CAs again with Diehard battery of tests for 5 different arbitrary initial seeds and fixed initial seed (i.e. 0^k20^k, $k = \lfloor \frac{n}{2} \rfloor$) and with more stringent TestU01 library.

5.1 Test with TestU01 Library

As TestU01 library [4] offers many more stringent battery of tests, we test our final rule set with TestU01 library. Among the different battery tests, we have selected the battery *rabbit* (bbattery_RabbitFile()) which takes a binary file as input and contains 39 stringent tests. Each of the 805 rules, when tested with the battery rabbit, passes $12 - 15$ tests for any arbitrary initial configuration. Some of the results of the 805 rules with TestU01 library for fixed initial configuration is shown in Table 5.

Table 5. Sample of good PRPGs. The test results are for fixed initial configuration

3-state CA Rules	#Tests passed in Diehard ($n = 15$)	#Tests passed in TestU01 ($n = 20$)	3-state CA Rules	#Tests passed in Diehard ($n = 15$)	#Tests passed in TestU01 ($n = 20$)
012012120021021021201021210	9	13	210120201201021210120021021	8	12
210201102210102012201102201	8	10	210201021210120201120102021	8	11
012012102012012102120210012	7	15	012012021120012210021120012	9	15
210120021210201021021120021	7	14	102201102012210120102210201	9	10
210102102210201012210102012	8	13	120210021012210012021210012	8	14
210201201210201102021120201	8	13	120012120201210120120021012	9	10
210210021201120012021021201	8	8	012012120102012102210012102	8	14
012012102102012201012012021	9	14	012012120012012102210012102	7	14
012012102021012021102102021	7	14	012120201012201210210021210	9	12
210201021201201021201021021	7	14	102210012021210201120210012	9	14
012012210102210120120210102	9	13	210102102210102012210102201	7	15
012012201102012102210012102	9	12	012012021201012012021012012	7	13
210210102201102102120102102	7	14	012012201201012201102012102	7	14
201102102210102201201102201	9	12	210210120210120102210120102	8	14
201102102210120201210120021	9	13	012021012201012201021210201	8	14
210201102210120201021210102	8	11	210210210210021012201021021	8	7
012210012210012201120210012	8	11	012021021021210201021210021	8	11
012021120012102012012021210	8	12	012021120012120120021021210	8	12
012021120120120120120201120	8	11	012021210012021021012210120	8	12
021102210210102012102102210	8	10	012120120012120021120021120	8	11
012120201012201120021201120	8	12	012201012210210102021210201	8	9
012201102012012012012210102	8	9	012102210201012120210012102	8	9
012201201021201210012201210	8	12	012201210201102102201102120	8	12
012102201201210120201210201	8	8	012210120021012102120102102	8	8
012210120210210012012210102	8	11	012210201021210120012210012	8	12
021012012021012012201012102	8	12	021012012210012102201012201	8	11
201102012102201120201210012	7	9	102210120120210021120210012	7	12
102210120012120120102102120	7	9	102210120102102120102210210	7	11
201021201201021021210210102	7	12	012021012012012102021012201	7	12

5.2 Cycle Length Test

We have conducted an experiment to find out the cycle lengths of these 805 rules for different values of n. Table 6 gives cycle lengths for a sample run on different values of n for some rules of Table 5. Note that, although, for some CAs, cycle length varies with cell length, but, we can observe that, the 805 CAs selected as possible PRPGs, have sufficiently long cycle length for most values of n, especially for $n = 15$ and thus strengthens our selection process.

Table 6. Cycle lengths from a sample run for some 3-state CAs of Table 5

Rule	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 10$	$n = 11$	$n = 12$	$n = 13$	$n = 14$	$n = 15$
012021120012102012012021210	104	65	22	55	1043	104	4630	5210	28456	22	123824
012021120120120120120201120	4	5	139	2	77	4	170	6575	20643	13005	96764
012021210012021021012210120	4	13	244	47	140	739	4421	1015	7201	7944	116924
012102201201210120201210201	94	95	258	1	1214	223	274	89	2846	8987	49094
012102210210102012102102210	54	17	42	231	2114	144	2309	17	19980	3156	91184
012120201012201120021201120	34	8	27	51	368	34	527	10475	11998	29553	36224
012201012210210102021210201	34	23	146	271	22	34	208	911	4731	2461	84284
012210012210012201120210012	94	17	167	143	188	339	2375	2639	378	9561	74234
012210201021210120012210012	19	5	167	3	143	39	21	357	17146	293	88364
021012012012012012201012102	36	5	251	3	134	2999	10262	8	24556	11570	151214
021012012210012102201012201	109	62	153	687	152	189	1033	9239	8878	8735	137534
201102102210120201210120021	49	8	31	2	458	49	2826	3563	6759	11129	54554

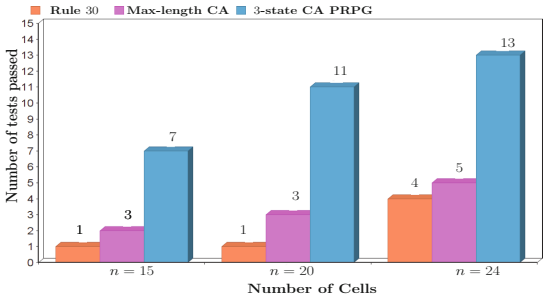


Fig. 3. Comparison of rule 30, max-length CA and 3-state CAs as potential PRPGs

5.3 Comparison

We have selected most popular binary CA with rule 30 [8] and a max-length CA [2] with rules 150 and 90 and tested on Diehard testbed for different values of n, such as 15, 20, & 24 with fixed initial seed, i.e., all zero except the middle bit as one (0^k10^k, $k = \lfloor \frac{n}{2} \rfloor$). Figure 3 shows the comparison result. In this figure, among the 805 3-state CAs, a CA is arbitrary selected as our PRPG, which is the rule 012012102012012102120210012 in this case. It can be observed that, our PRPG beats the PRPG based on rule 30 as well as the max-length CA for the minimum cell length $n = 15$.

References

1. Bhattacharjee, K., Das, S.: Reversibility of d-state finite cellular automata. J. Cell. Automata **11**(2–3), 213–245 (2016)
2. Chaudhuri, P.P., Chowdhury, D.R., Nandi, S., Chatterjee, S.: Additive Cellular Automata Theory and Applications. IEEE Computer Society Press, New York (1997). ISBN 0-8186-7717-1
3. Das, S., Sikdar, B.K.: A scalable test structure for multicore chip. IEEE Trans. CAD Integr. Circ. Syst. **29**(1), 127–137 (2010)
4. L'Ecuyer, P., Simard, R.: TestU01: A C library for empirical testing of random number generators. ACM Trans. Math. Softw. **33**(4), 22 (2007)
5. Marsaglia, G.: DIEHARD: a battery of tests of randomness (1996). http://stat.fsu.edu/geo/diehard.html
6. Sethi, B., Roy, S., Das, S.: Asynchronous cellular automata and pattern classification. Complexity (2016). doi:10.1002/cplx.21749
7. Smith III, A.R.: Cellular automata complexity trade-offs. Inf. Control **18**, 466–482 (1971)
8. Wolfram, S.: Random sequence generation by cellular automata. Adv. Appl. Math. **7**(2), 123–169 (1986)
9. Wuensche, A.: Complex and chaotic dynamics, basins of attraction, and memory in discrete networks. Acta Phys. Pol. B-Proc. Suppl **3**, 463–478 (2010)