A Transformation Sequencing Approach to Pseudorandom Number Generation

Syn Kiat Tan[1] and Sheng-Uei Guan[2]

[1]Department of Electrical and Computer Engineering
National University of Singapore
10 Kent Ridge Crescent, Singapore 119260

[2]School of Engineering and Design
Brunel University
Uxbridge, Middlesex, UB8 3PH, UK

Abstract

This paper presents a new approach to designing pseudorandom number generators based on cellular automata. Current cellular automata designs either focus on i) ensuring desirable sequence properties such as maximum length period, balanced distribution of bits and uniform distribution of *n*-bit tuples etc. or ii) ensuring the generated sequences pass stringent randomness tests. In this work, important design patterns are first identified from the latter approach and then incorporated into cellular automata such that the desirable sequence properties are preserved like in the former approach. Preliminary experiment results show that the new cellular automata designed have potential in passing all DIEHARD tests.

Keywords: cellular automata, pseudorandom number generation, maximum length period, randomness test

1. Introduction

Random numbers are needed in a variety of scientific, mathematical, engineering and industrial applications including cryptography, built-in self test, artificial evolution such as genetic algorithm and simulated annealing, Monte Carlo simulations, etc [1,6]. Finding appropriate pseudorandom number generators (PRNG) is a difficult task [5] - it is known that every PRNG has to fail in a certain simulation/statistical test, or in certain setups that interfere with the particular regularities of a given PRNG and thus exhibits the hidden correlations between numbers. It is also desirable to have low-cost PRNG that can generate sequences with desirable properties such as maximum period length, balanced distribution of 1 and 0, uniform distribution of the *n*-bit states, etc.

Wolfram [1] first proposed using the one-dimensional cellular automata (1-d CA) as a PRNG. Figure 1 shows a 4-bit CA - an array of four binary registers where each register's new state $s_k^{(t+1)}$ is computed by a transformation function Φ_k over a neighborhood of three nearest registers' current states, $s_k^{(t+1)} = \Phi_k(s_{k-1}^{(t)}, s_k^{(t)}, s_{k+1}^{(t)})$, $0 \le k \le n-1$.

In previously published CA PRNG works [7, 8, 11-18], authors have generally focused on a particular objective(s) from the full set of desirable PRNG properties. In [7, 8, 11, 12], the focus is on analytical techniques to configure the set of $\{\Phi_0, \Phi_1, \cdots, \Phi_{n-1}\}$ for finite length *n*-bit CA to generate sequences $\{s_k^{(0)}, s_k^{(1)}, \cdots, s_k^{(p)})$ with the maximum period $p = 2^n - 1$. Other researchers [13-18] hypothesized that by making structural changes to the CA, the

randomness quality of the generated sequences $\{s_k^{(0)}, s_k^{(1)}, s_k^{(2)}, \cdots)$ can be improved and verified though empirical testing with the DIEHARD statistical test suite [9]. These include changing the connection structure or type of neighborhood to draw inputs [3,16,18] from, dimensionality of CA [12,13], the individual transformation function used for each register [2,14,15,17], etc.

In conventional CA, the neighborhood of nearest three inputs ($s_k^{(t+1)} = \Phi_k(s_{k-1}^{(t)}, s_k^{(t)}, s_{k+1}^{(t)})$) allows up to 2^{2^3} possible transformation functions to be used by each register. In [13, 16, 18], each CA register's transformation function is allowed to use up to five inputs over a non-local neighborhood or arranged in a 2-d lattice. The overall space for the CA transformation functions $\{\Phi_0, \Phi_1, \cdots, \Phi_{n-1}\}$ is thus exponentially increased. Through extensive testing, function configurations passing all DIEHARD tests are found. Their results suggested that strong correlation between adjacent registers may be detrimental to the randomness quality of generated sequences. This hypothesis is also affirmed in [4] where the authors concluded that correlation dies out between registers that are at least four sites apart.

To further reduce correlation in single-bit sequences from adjacent registers, researchers began to explore CA registers that can be configured to use time-varying transformation functions per clock $\{\Phi_0^{(t)}, \Phi_1^{(t)}, \cdots, \Phi_{n-1}^{(t)}\}$, $t = 0,1,2,...$ [2,14,15,17]. It is hypothesized that the consecutive states are less likely to be correlated since each new state is generated from

the current state by a different function. In previous works, each new state $s_k^{(t+1)}$ is given as the successive application of the same transformation Φ_k on the current state $s_k^{(t)}$ and regularities may be inevitable in these sequences.

Interestingly, we do not know of any proposal in the literature that contains theoretical analysis on the properties of the proposed CA as well as DIEHARD test results on the generated sequences. We conducted our own DIEHARD testing on maximum length sequences [8] and found that their randomness quality is generally not satisfactory [17]. On the other hand, analysis is difficult and seldom done for the highly complex CA designed to pass all DIEHARD tests; performance evaluation is usually based on experimental results. For example, CA with nonlinear transformation functions [18] cannot be studied using the matrix approach [2] while others are difficult to model mathematically due to either the set of time-varying transformations used or the behavior to be effected.

To circumvent analysis, some authors [13-16] applied genetic algorithms [10] with the fitness function defined from the results of some relevant randomness metrics such as entropy, correlation, DIEHARD, etc. to evolve the CA transformation $\{\Phi_0, \Phi_1, \cdots, \Phi_{n-1}\}$. However, for CA to be confidently deployed as PRNG in many applications, rigorous testing needs to be conducted – this slows down evolutionary approaches tremendously because of the vast number of fitness evaluations to be performed repeatedly. For example, the 2-d array CA [16] is shown to pass all DIEHARD tests, but only three suitable

configurations were found after evolution from an initial population of 80 candidates. It is also observed that relatively large CA sizes were often used in the proposals reviewed – possibly to avoid small period lengths that can arise since the states of these CA can reside in more than one cycle.

Some recent designs of CA seem to be driven by making changes in the CA structure first, and the generated sequences are then checked for desired properties and tested for DIEHARD. When we design a CA to generate highly random number sequences, the focus should be on its global behavior - the desired sequence properties. While we are certain that the global behavior is brought about from the interaction among individual registers, there are unfortunately no clear links as to how desired global behavior can be achieved by considering separately the individual behavior of the registers contained in the CA. We can study the interaction between a few registers but this quickly becomes infeasible due to the exponential growth of possible configurations that can arise from the multiplicity of register functions, arrangements of registers, etc.

Can CA-level modifications be applied similarly at the CA transformation level that will result in the solutions we expect, such that each register function is specifically modified and the concerted effect of all modified registers generates the sequence properties we look for? Ultimately, the randomness quality of sequences generated still depends on the overall CA transformation $\{\Phi_0, \Phi_1, \cdots, \Phi_{n-1}\}$.

We have surveyed the present state of art in CA based PRNG and identified common design patterns in previous CA works that passed all 19 DIEHARD tests: (a) time-varying transformations and (b) transformation functions with more inputs over a non-local neighborhood. The high randomness quality of generated sequences is possibly attributed to these design patterns and thus they should be incorporated. Bearing desirable, analytical properties in mind, we have therefore come up with the new Transformation Sequencing (TS) approach that merges the objectives from both analytical and empirical works to provide guaranteed sequence properties:

i) long period,

ii) balanced distribution of '1' and '0',

iii) uniform distribution of *n*-bit states.

The concept of transformation sequences will be introduced in Section 2 and some properties of generated sequences will be highlighted in Section 3. Possible implementations based on the TS approach are also suggested. DIEHARD results are presented in Section 4. The conclusions are drawn in Section 5.

2. Transformation Sequencing for CA

Linear binary CA, where each register's state is computed by a linear function, are also considered as linear finite state machines (LFSM) [8]. Hereafter, all CA referred to are linear, binary, maximum length CA [11] (generate sequences with a maximum period

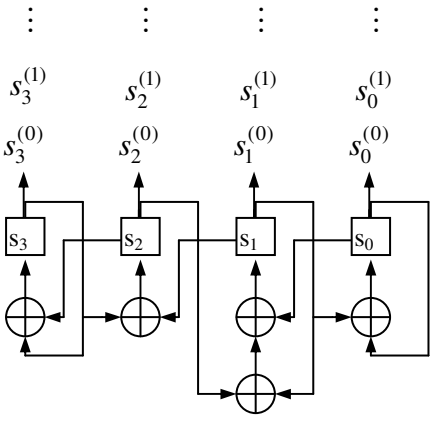
$p = 2^n - 1$) unless otherwise specified. Figure 1 shows a 4-bit example. An n-bit CA state at time (t) can be denoted by the vector $S^{(t)} = [s_{n-1}^{(t)}, s_{n-2}^{(t)}, \ldots, s_0^{(t)}]'$ and each CA has an n-by-n binary transformation matrix A which maps the current state to the next state, i.e. $S^{(t+1)} = A \cdot S^{(t)}$ [2].

For a maximum length CA, the transformation matrix has the property $A^{(2^n-1) \bmod (2^n-1)} = A^{0 \bmod (2^n-1)} = I$ such that the sequence of generated states $\{S^{(0)}, S^{(2)}, \ldots, S^{(2^n-2)}\}$ has a period $2^n - 1$, i.e. all possible n-bit tuples except $[0,0,\ldots,0]'$ are generated exactly once. As stated in [2], we have the following relations:

$$S^{(t)} = A^t \cdot S^{(0)}, t = 0,1,2,\ldots \quad (1)$$

$$S^{(t+2^n-1)} = A^{2^n-1} \cdot S^{(t)} = S^{(t)} \quad (2)$$

Besides having a maximum period, these sequences $\{S^{(0)}, S^{(1)}, \ldots, S^{(2^n-2)}\}$ have desirable properties such as balanced distribution of bits and uniform distribution of n-bit tuples [8].



$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Figure 1. A 4-bit CA and its transformation matrix A

To avoid confusion in subsequent mathematical expressions, the superscript ^{f} without parenthesis is used to denote A^{f}, which can be understood as $A^{f} \equiv \prod_{i=1}^{f} A$, while the superscript $^{(t)}$ is used to:

i) indicate the transformation matrix $A^{(t)}$ used at time *(t)* – $A^{(t)}$ can take on any value from the set $A^{(t)} \in \{A^{f}\}_{f=1}^{2^{n}-1}$ (note that generally $A^{(t)} \neq A^{t}$), and

ii) denote the output state $S^{(t)}$ vector at time *t*.

The subscript $_{k}$ is used to indicate a specific bit or register within a vector, i.e. the k^{th} bit $s_{k}^{(t)}$ within $S^{(t)}$.

To generate the state $S^{(t)}$ in a CA such as the one shown in Figure 1, the transformation A is applied to the state $S^{(0)}$ t times successively. Consider the state sequence generated in a single period by a CA, we can write the following using (1):

$$\{S^{(1)}, S^{(2)}, \ldots, S^{(2^{n}-1)}\} \equiv \{A^{1}, A^{2}, \ldots, A^{2^{n}-1}\} \cdot S^{(0)} \qquad (3)$$

This ordered sequence of transformations $\{A^{1}, A^{2}, \ldots, A^{2^{n}-1}\}$ is defined over a single complete period (from (2), it has the same period as its associated state sequence) and contains the transformation matrices applied to the initial state $S^{(0)}$ at each clock *(t)* to generate the output. Called the transformation sequence, it can be viewed as a fixed process

inherent to the CA. Regardless of the initial state $S^{(0)}$ used, the CA always uses this transformation sequence to generate the successive output states.

Let Φ be the global, iterative transformation function of an arbitrary PRNG such that its states are computed by $S^{(t+1)} = \Phi(S^{(t)})$ and its transformation sequence over a complete period be $\{\Phi^1, \Phi^2, \Phi^3, \Phi^4, \Phi^5, \Phi^6, \Phi^7\}$. If a CA is found whose transformation matrix A can be written as $\{A^1, A^2, A^3, A^4, A^5, A^6, A^7\} \equiv \{\Phi^1, \Phi^2, \Phi^3, \Phi^4, \Phi^5, \Phi^6, \Phi^7\}$, the PRNG is actually equivalent to that CA. Consider the case where the transformation sequence of an arbitrary PRNG expressed using the transformation matrix of any CA is still of the following "jumbled order" form, for example $\{A^3, A^5, A^6, A^2, A^4, A^1, A^7\} \equiv \{\Phi^1, \Phi^2, \Phi^3, \Phi^4, \Phi^5, \Phi^6, \Phi^7\}$. We hypothesize that no equivalent linear CA can be used to represent this PRNG, in other words, Φ is nonlinear and the output sequences produced by this PRNG is nonlinear. As long as each transformation $A^f, 0 \le f \le 2^n - 1$ appears exactly once in the equivalent transformation sequence (over a single period), each $S^{(t)} = [s_{n-1}^{(t)}, s_{n-2}^{(t)}, \ldots, s_0^{(t)}]'$ is generated exactly once. The corresponding PRNG's output state sequence thus looks like a permuted version of the CA's sequence, for example $\{S^{(3)}, S^{(5)}, S^{(6)}, S^{(2)}, S^{(4)}, S^{(1)}, S^{(7)}\}$ from the PRNG compared to the CA's $\{S^{(1)}, S^{(2)}, S^{(3)}, S^{(4)}, S^{(5)}, S^{(6)}, S^{(7)}\}$. We have particular interests in such "jumbled order" sequences and the reasons will be made clear in the following section. Note for now that such sequences still retain one of the desirable properties, i.e. maximum length period.

3. Properties and Implementations of Transformation Sequencing

We now describe some properties and possible implementations based on the Transformation Sequencing approach.

Proposition 1. If the transformation sequence $\{A^{(1)}, A^{(2)}, \ldots, A^{(2^n-1)}\}$ of a PRNG contains all possible $A^{(t)} \in \{A^f \mid f = 1, 2, ..., 2^n - 1\}$ exactly once, then the generated sequences have a maximum length period $2^n - 1$, balanced distribution of '1' and '0' as well as uniform distribution of *n*-bit states. Working with transformation sequences allows us to identify easily that each transformation from the set $\{A^f \mid f = 1, 2, ..., 2^n - 1\}$ appears exactly once in a single period.

Proposition 2. If we let a PRNG's transformation be $S^{(t+1)} = A^f \cdot S^{(t)}$ (for $gcd(f, 2^n\text{-}1) = 1,\ f > 1$) where *A* is the transformation matrix of any maximum length CA from [11], it can be shown that the transformation sequence $\{A^f, A^{2f}, \ldots, A^{(2^n-1)f}\}$ will contain each transformation $A^{(t)} \in \{A^f \mid f = 1, 2, ..., 2^n - 1\}$ in one period. This implies that this PRNG satisfies Proposition 1 and generates sequences having properties stated therein. Recall from Section 1 the desirable design pattern of using register functions with more inputs over a non-local neighborhood; we now show that this PRNG incorporates the above.

It can be observed that each row in A^f (where *gcd(f,2^n-1)=1,* $f>1$) may contain more elements of '1' compared to the rows of A^1. These '1's correspond to the inputs used by each individual register function to compute its new state [2]; and each row vector $\vec{A}_k^f$ ‡ in A^f corresponds to a linear function for the *k-th* register. Since $\{A^{(1)}, A^{(2)}, \ldots, A^{(2^n-1)}\}$ generates a sequence with a period as 2^n-1, each $\vec{A}_k^f$, $0 \le f < 2^n-1$ cycles through [00…1] to [11…1]. To illustrate, consider the complete set of transformations $\{A^1, A^2, ..., A^{15}\}$ obtained from the 4-bit CA in Figure 1.

$$A=\begin{bmatrix}1&1&0&0\\1&0&1&0\\0&1&1&1\\0&0&1&0\end{bmatrix},\ A^2=\begin{bmatrix}0&1&1&0\\1&0&1&1\\1&1&1&1\\0&1&1&1\end{bmatrix},\ A^3=\begin{bmatrix}1&1&0&1\\1&0&0&1\\0&0&1&1\\1&1&1&1\end{bmatrix},\ A^4=\begin{bmatrix}0&1&0&0\\1&1&1&0\\0&1&0&1\\0&0&1&1\end{bmatrix},$$

$$A^5=\begin{bmatrix}1&0&1&0\\0&0&0&1\\1&0&0&0\\0&1&0&1\end{bmatrix},\ A^6=\begin{bmatrix}1&0&1&1\\0&0&1&0\\1&1&0&0\\1&0&0&0\end{bmatrix},\ A^7=\begin{bmatrix}1&0&0&1\\0&1&1&1\\0&1&1&0\\1&1&0&0\end{bmatrix},\ A^8=\begin{bmatrix}1&1&1&0\\1&1&1&1\\1&1&0&1\\0&1&1&0\end{bmatrix},$$

$$A^9=\begin{bmatrix}0&0&0&1\\0&0&1&1\\0&1&0&0\\1&1&0&1\end{bmatrix},\ A^{10}=\begin{bmatrix}0&0&1&0\\0&1&0&1\\1&0&1&0\\0&1&0&0\end{bmatrix},\ A^{11}=\begin{bmatrix}0&1&1&1\\1&0&0&0\\1&0&1&1\\1&0&1&0\end{bmatrix},\ A^{12}=\begin{bmatrix}1&1&1&1\\1&1&0&0\\1&0&0&1\\1&0&1&1\end{bmatrix},$$

$$A^{13}=\begin{bmatrix}0&0&1&1\\0&1&1&0\\1&1&1&0\\1&0&0&1\end{bmatrix},\ A^{14}=\begin{bmatrix}0&1&0&1\\1&1&0&1\\0&0&0&1\\1&1&1&0\end{bmatrix},\ A^{15}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}.$$

The register functions in A^f ($f \neq 1, 2^n-1$) can have an increased number of inputs from any register in the CA and the notion of local neighborhood is not applicable. See the 4-bit CA configured using A^2 in Figure 2 for a clearer view - the third register's transformation

‡ $\vec{A}_k^f$ refers to the *k*th row vector of A^f, $0 \le k \le n-1$.

function actually XOR all the four neighboring inputs to compute its next state. The state sequence is then generated as $S^{(t)} = A^{2 \cdot t} \cdot S^{(0)}$.

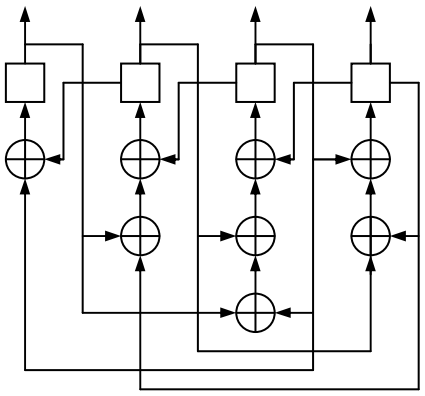


Figure 2. A 4-bit CA configured using A^2

Proposition 3. If a PRNG's transformation sequence can only be expressed as a permuted order form of any other CA's transformation sequence, it implies that time-varying linear transformations are used. For example we have $\{\Phi^1, \Phi^2, \Phi^3, \Phi^4, \Phi^5, \Phi^6, \Phi^7\} \equiv \{A^3, A^5, A^6, A^2, A^4, A^1, A^7\}$ for any A. The actual state transformation at each clock *(t)* is respectively $S^{(1)} = A^3 \cdot S^{(0)}$, $S^{(2)} = A^2 \cdot S^{(1)}$, $S^{(3)} = A \cdot S^{(2)}$,… It can be seen that such sequences have the properties stated in Proposition 1 and this PRNG incorporates both design patterns given in Section 1.

One possible way to implement a PRNG with the above transformation sequence is via

$$Z^{(t)} = A^{|C^{(t)}|} \cdot S \qquad (4)$$

The transformation matrix $A^{|C^{(t)}|}$ at each clock is given by the decimal state $|C^{(t)}|$ of another maximum length CA. Here, *S* is the initial state. Since the sequence $\{|C^{(t)}|\}_{t=1}^{2^n-1}$ has a period 2^n-1, each transformation from the set $\{A^f \mid f=1,2,...,2^n-1\}$ appears exactly once.

4. DIEHARD Results

We now provide the DIEHARD [9] test results for randomness quality of the generated sequences from the PRNG described in Proposition 3 (codenamed TS-1) and the PRNG in Proposition 2 (codenamed TS-2). Details on the DIEHARD tests can be found in the cited reference [9]. For each *n* (the length of the CA or TS-1 used), a transformation matrix *A* is obtained from the list given in [11]. The state sequence $\{S^{(t)}\}_{t=1}^{2^n-1}$ for the CA is generated using $S^{(t)} = A^t \cdot S^{(0)}$ (from (1)). For the TS-1, the sequence $\{|C^{(t)}|\}_{t=1}^{2^n-1}$ is obtained using $C^{(t)} = A^t \cdot C^{(0)}$ while the final state sequence is obtained using (4). For the TS-2, we examine sequences generated from a 24-bit example. These sequences are generated from $S^{(t)} = A^{f \cdot t} \cdot S^{(0)}$ where $f = 2^0, 2^1, 2^2, ..., 2^{23}$ The results are averaged from experiments conducted using 20 different initial states $S^{(0)}$. The results for ordinary linear maximum length CA are provided as well for benchmark purposes in Figure 3, since both TS-1 and TS-2 are constructed from such maximum length CA.

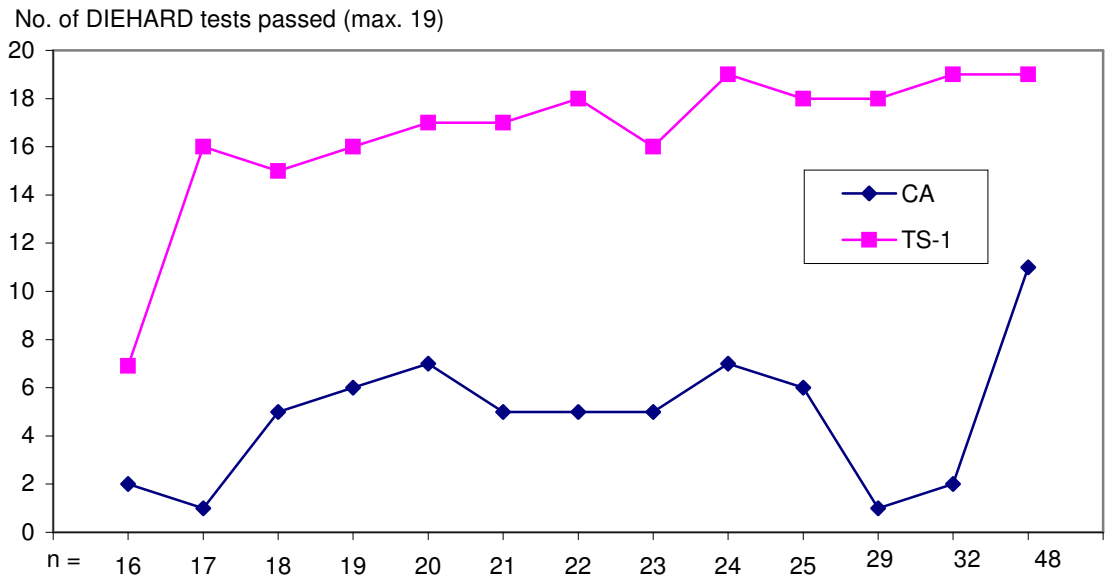


Figure 3. Number of DIEHARD tests passed (maximum score of 19) by 16- to 48-bit CA and TS-1

In Figure 3, the vertical axis shows the number of DIEHARD tests passed (maximum 19) as a function of *n*, the length of the CA or TS-1 used. The randomness quality of maximum length CA sequences is generally not good, as detected by the DIEHARD tests. The random quality of TS-1 clearly exceeds the CA in all cases tested. For 24-bit and larger cases, nearly all 19 tests are passed.

The results of for the 24-bit version of the PRNG in Proposition 2 (codenamed TS-2) are given in Figure 4. Since there are many A^f that satisfy the constraint $\gcd(f, 2^n - 1) = 1,\ f > 1$, we chose A^{2^k}, $k = 1, 2, \ldots n-1$ since $\gcd(2^k, 2^n - 1) = 1$ holds.

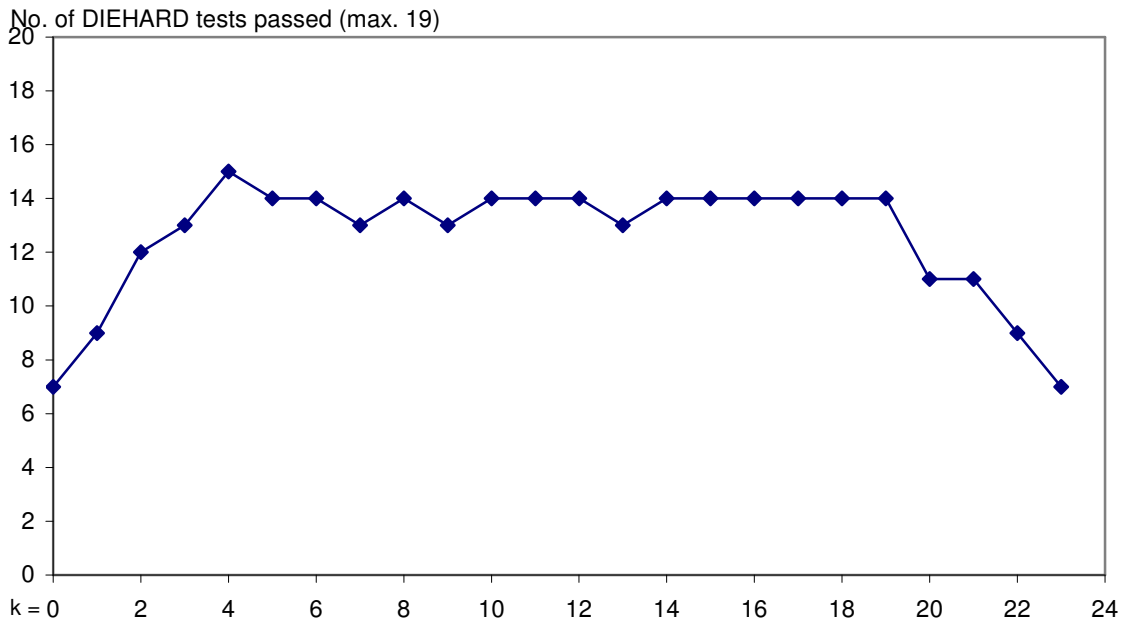


Figure 4. Number of DIEHARD tests passed (maximum score of 19) by 24-bit TS-2 with $A^{2^k}, k = 0,1,...,23$

In Figure 4, the vertical axis shows the number of DIEHARD tests passed (maximum 19) as a function of k in the transformation A^{2^k} used in TS-2. The data point on the vertical axis ($k=0$) represents the result of the ordinary CA with transformation A. The random quality of TS-2 exceeds the CA although the degree of improvement is not very consistent for the various values of k. The result of the 24-bit version of TS-1 outperforms the TS-2. This can be explained by TS-1 incorporating both design patterns listed in Section 1 while TS-2 only incorporates one pattern.

5. Conclusion

New PRNG can be designed from existing linear maximum length CA such that generated sequences can be viewed as having *n*-bit blocks permuted compared to the original CA sequence. Working with transformation sequences allows us to identify easily that each transformation from the set $\{A^f \mid f = 1, 2, ..., 2^n - 1\}$ appears exactly once such that the generated sequences have a maximum length period as $2^n - 1$, balanced distribution of '1' and '0' as well as uniform distribution of *n*-bit states. The DIEHARD results of the PRNG in Propositions 2 and 3 suggested that incorporating the design patterns improves the randomness quality of generated sequences, especially when both patterns are used. Although CA is used for illustration, the transformation sequence approach can be used with any maximum length LFSM [7].

References

[1] S. Wolfram, "Theory and Applications of Cellular Automata: Including Selected Papers 1983-1986", World Scientific publishing Co., Inc., River Edge, NJ. 1986.

[2] P. Pal Chaudhuri, D. Roy Chowdhury, S. Nandi and S. Chattopadhyay, "Additive Cellular Automata Theory and Applications", Volume 1, IEEE Computer Society Press, Los Alamitos, ISBN 0-8186-7717-1, California, 1997.

[3] S. Nandi and P. Pal Chaudhuri, "Analysis of Periodic and Intermediate Boundary 90/150 Cellular Automata", IEEE Trans. on Computers, Vol. 45, No. 1, pp. 1-12, Jan. 1999.

[4] P. D. Hortensius, R. D. Mcleod, W. Pries, D. M. Miller and H. C. Card, "Cellular Automata-Based Pseudorandom Number Generators for Built-in Self-Test", IEEE Trans. on Computer-Aided Design, Vol. 8, No. 8, pp. 842-859, 1989.

[5] P. Hellekalek, "Good Random Number Generators are (Not So) Easy to Find", In Mathematics and Computer in Simulation, 46, pp. 485-505, 1998.

[6] P. Sarkar, "A Brief History of Cellular Automata, ACM Computing Surveys", Vol. 32, No. 1, pp. 80-107, 2000.

[7] G. Mrugalski, J. Rajski and J. Tyszer, "Ring Generators - New Devices for Embedded Test Applications", IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol. 23, No. 9, pp.1306–1320, Sept. 2004.

[8] Solomon W. Golomb, "Shift Register Sequences", Aegean Park Press, 1981.

[9] G. Marsaglia, "Diehard", http://stat.fsu.edu/~geo/diehard.html, 1998.

[10] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs", Berlin: Springer-Verlag, 1992.

[11] S. Zhang, D.M. Miller and J. C. Muzio, "The Determination of Minimal Cost One-Dimensional Linear Hybrid Cellular Automata", Elec. Letters, Vol. 27, pp. 1625 - 1627, 1991.

[12] K. Cattell, S. Zhang, M. Serra and J. C. Muzio, "2-by-n Linear Hybrid Cellular Automata with Regular Configurations," IEEE Trans. Computers, Vol. 48, pp. 285 - 295, 1999.

[13] M. Tomassini, M. Sipper and M. Perrenoud, “On the Generation of High-Quality Random Numbers by Two-Dimensional Cellular Automata”, IEEE Trans. on Computers, Vol. 49, pp. 1146-1151, 2000.

[14] Sheng-Uei Guan and Shu Zhang, “An Evolutionary Approach to the Design of Controllable Cellular Automata Structure for Random Number Generation”, IEEE Trans. on Evolutionary Computation, Vol. 7, No. 1, pp. 23 -36, Feb. 2003.

[15] Sheng-Uei Guan and Shu Zhang, “Incremental Evolution of Cellular Automata for Random Number Generation”, International Journal of Modern Physics C, Vol. 14, No. 7, pp. 881-896, Sep. 2003.

[16] Sheng-Uei Guan, Shu Zhang, and Marie Therese Quieta, "2-d CA Variation with Asymmetric-Neighborship for Pseudorandom Number Generation", IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems, Vol. 23, No. 3, pp. 378-388, Mar. 2004.

[17] Sheng-Uei Guan and Syn Kiat Tan, "Pseudorandom Number Generation with Self Programmable Cellular Automata", IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems, Vol. 23, No. 7, pp. 1095-1101, July 2004.

[18] Franciszek Seredynski, Pascal Bouvry and Albert Y. Zomaya, “Cellular Automata Computations and Secret Key Cryptography”, Journal of Parallel Computing, Vol. 30, Issue 5-6, Elsevier, pp. 753-766, 2004.