Permutation and Sampling with Maximum Length CA for Pseudorandom Number Generation

Sastra Wijaya, Syn Kiat Tan, Sheng-Uei Guan[1]

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

Abstract—In this paper, we study the effect of dynamic permutation and sampling on the randomness quality of sequences generated by cellular automata (CA). Dynamic permutation and sampling have not been explored in previous CA work and a suitable implementation is shown using a two CA model. Three different schemes that incorporate these two operations are suggested - Weighted Permutation Vector Sampling with Controlled Multiplexing, Weighted Permutation Vector Sampling with Irregular Decimation and Permutation Programmed CA Sampling. The experiment results show that the resulting sequences have varying degrees of improvement in DIEHARD results and linear complexity compared to the CA.

Keywords—cellular automata, pseudorandom number generation, randomness testing, data-dependent permutation, dynamic sampling.

[1] Corresponding Author, email: sg_1_1@yahoo.com

I. INTRODUCTION

Pseudorandom number generators [3] (PRNG) are used widely in a variety of scientific, mathematical, engineering and industrial applications. PRNG are evaluated on the randomness quality of its generated sequences. The 19 DIEHARD statistical tests [4], widely regarded as the most comprehensive and stringent, are typically used for this purpose. Unpredictability is also an important indicator of randomness and is conventionally measured by the sequence's linear complexity [17].

Binary cellular automata (CA) are observed to display chaotic behavior and fractal patterns [1]. Since then, CA has been widely studied in the design of pseudo random numbers generators (PRNG) [5-15] and the results showed that CA generated sequences have superior randomness quality over conventional designs [7]. Furthermore, CA designs allow massive parallelism, locality of cellular interactions, and simplicity of basic processing units [6].

Initially, the research focus is on the configuration of linear CA with n binary registers and XOR gates to produce maximum length sequences of period $p=2^n-1$. As researchers [7] have pointed out, such linear maximum-length CA (m-CA) suffers from the same weakness with conventional linear designs such as the linear feedback shift registers [17] (we have included some results of m-CA for reference in Fig. 3). These shortcomings prompted new research trends for CA-based PRNG; towards the exploration of two-dimensional CA, hybrid and dynamic state transformations [7-14]. In particular, modular designs are suggested in [17] where a linear m-CA is used for generating sequences with guaranteed long periods and a nonlinear Boolean function is

used to remove linearity and improve unpredictability. Suitable nonlinear Boolean functions are usually very large and complex whose logic gate count can be exponential with the number of inputs.

Several authors hypothesized that increased complexity in the dynamic behavior of CA can lead to better randomness quality [6,13,14]. The Programmable CA [6] consists of a main-CA and an external control source such that the main-CA registers use different functions depending on the state of the corresponding control bit. A variety of Controllable CA [13] uses two external control-CA to generate the control bits. The Self-Programmable CA [14] suggested the use of time-lagged memory bits as the control source. These designs generate sequences passing all 19 DIEHARD tests. In these CA models, the focus is on the configuration of register functions and for each CA length, substantial work is often involved to configure the CA models before DIEHARD can be passed.

Since m-CA configurations are readily available [5], we explore the feasibility of applying external processing - data dependent permutation and sampling, on the sequences generated by m-CA of various lengths such that DIEHARD can be passed without extensive tuning. The term “data dependent” is used to signify that these permutation and sampling are dynamic. These two processing are implemented using the two-CA model suggested in [14]. Following the modular approach in nonlinear designs [17], the PRNG design is segregated such that the requirement for a long period is satisfied by the main-CA (i.e. an m-CA is used) while additional processing to remove linearity is performed by the control-CA (which implements the permutation and sampling behavior).

Permutation, a widely used technique in the design of ciphers [19], have not been considered in the design of CA based PRNG as far as we know. Only the fixed forms of sampling – time spacing and site spacing, have been used [7] for improving DIEHARD results. We explore permutation and sampling in various combinations to improve the randomness quality of sequences generated by m-CA.

II. CELLULAR AUTOMATA

An *n*-bit CA is an array of *n* binary registers $S^{(t)} = [s_0^{(t)}, s_1^{(t)}, ..., s_{n-1}^{(t)}]$ where each register's state $s_j \in \{0,1\}$ and $0 \le j \le n-1$. During each discrete time *(t)*, each register in the CA updates its state using a transformation function $f_j(.)$ applied to the current states of neighboring registers, $s_j^{(t+1)} = f_j(s_{j-r}^{(t)}, ..., s_j^{(t)}, ..., s_{j+r}^{(t)})$ where *r* denotes the neighborhood radius. The conventional nearest-three-input neighborhood, having $r = 1$, consists of the register itself s_j, and its left/right neighbors s_{j-1} / s_{j+1}.

An *n*-bit CA is called maximum length (m-CA) when it generates sequences of period $2^n - 1$ where all non-zero *n*-bit tuples appears exactly once. Fig. 1 shows a 4-bit m-CA. Using only XOR gates for their linear state transition functions $f_j(.)$, these m-CA have very simple linear structure and low cost implementation associated with the nearest-three input neighborhood. In [5], m-CA configurations are provided up to $n = 500$.

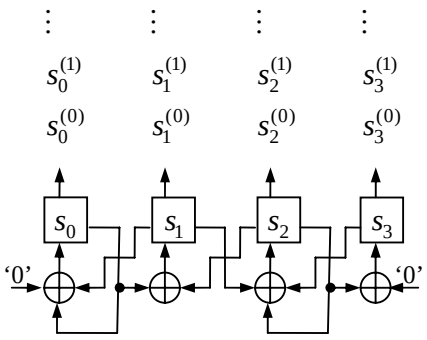

Fig. 1. A 4-bit m-CA implemented in hardware.

The states of a CA during each discrete time step can be successively sampled to form a pseudorandom n-bit word sequence $\{S^{(0)}, S^{(1)}, S^{(2)}, \ldots\}$ or sampled from a particular register to form the single-bit sequence $\{s_j^{(0)}, s_j^{(1)}, s_j^{(2)}, \ldots\}$. For the m-CA, these single-bit sequences are cyclically equivalent [6] with a phase shift.

A. Performance of m-CA

We now present the results on sequences generated by the m-CA to demonstrate the need for more sophisticated methods to improve the randomness quality of these sequences. The two tests used throughout this paper are the 19 DIEHARD statistical tests [4] (details about the tests can be found in the given reference and Table I) for evaluating the randomness quality and the Massey-Berlekamp algorithm [17] for measuring linear complexity (see Section IV.B). A 10M byte sequence is required for DIEHARD testing and sequences with period smaller than 10M bytes are not likely to pass all tests.

Fig. 2 shows the DIEHARD results of single-bit sequences drawn from each register from the 16-, 24- and 32-bit m-CA. For the 32-bit m-CA, the performance is satisfactory with an average of 16.5 DIEHARD tests passed. For 16- and 24-bit m-CA, the DIEHARD results are poor with an average of 10.5 and 4 tests passed respectively. The above results suggest that a larger CA size leads to better DIEHARD results – partly explained by DIEHARD's testing requirement of 10M byte sequences. The 24-bit m-CA has a period of $2^{24}-1$ and is the smallest m-CA satisfying the 10M byte requirement without any repeating. Obviously the sequences from the 16-bit m-CA have repeated many times in the 10M byte sequence and this non-randomness is detected by most of DIEHARD tests.

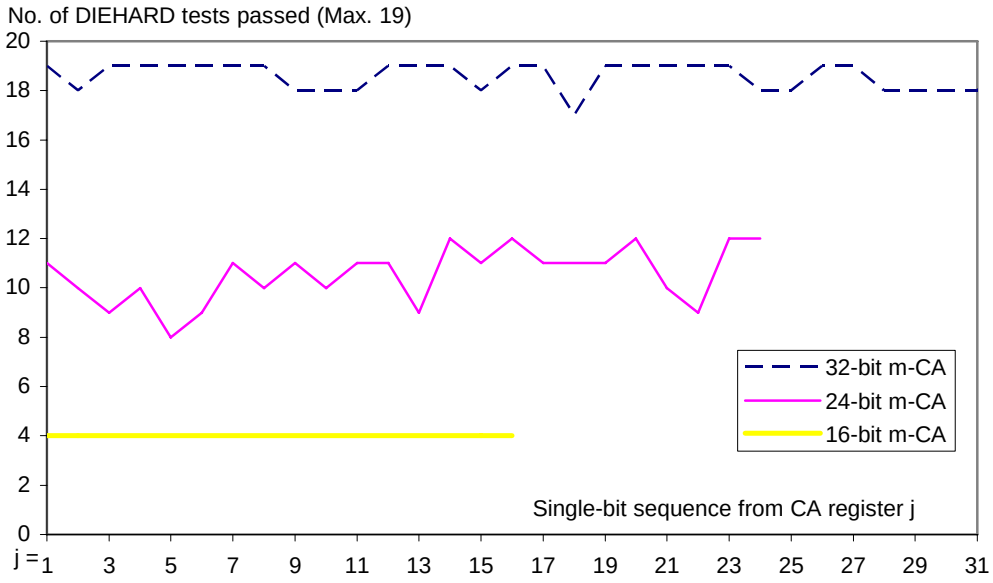

Fig. 2. DIEHARD results of 16-, 24- and 32-bit m-CA

Notice the results for 16-bit m-CA is consistent compared to 24- and 32-bit m-CA. Firstly, recall that the single-bit sequences drawn from any m-CA register is cyclically equivalent. The 16-bit m-CA produces sequences with period $p = 2^{16} - 1$, which is

repeated several times within the 10M byte sequence tested by DIEHARD. For 24- and 32-bit with $p = 2^n - 1,\ n = 24, 32$, only part of the complete period forms the 10M byte sequence for testing. If the phase shift is sufficiently large, effectively different sequences are tested by DIEHARD and different local non-randomness in these sequences can give rise to non-uniformity in DIEHARD results among the sequences produced by the different registers in the same m-CA.

III. Proposed Directions: Permutation and Sampling

A. Data dependent permutation – GRP operation

Shannon [18] proposed two significant concepts that lay the foundation for the design of modern ciphers – confusion and diffusion. By themselves, these two concepts are cryptographically weak and are implemented by simple operations (substitution and permutation respectively) but when combined repetitively, desired levels of security can be attained. Substitution has been widely explored in the designs of filter generators, combiner generators [17], etc. and normally a highly nonlinear Boolean function is used for the purpose. This function is normally very complex and the logic gate count can be exponential with the number of inputs. On the other hand, permutation is a linear operation that does not improve security when used directly by itself.

Data-dependent permutation [16] is dynamic and the actual permutation is selected by a secret key (data). Recently, data-dependent permutation is proposed in the design of cryptographic primitives and future generation processors; several new variants of data-dependent permutation have been proposed - GRP, OMFLIP [16], etc.

For some pairs of m-CA registers, there can be an undesirable high level of correlation among their output sequences (especially if their phase shift is small). It is hypothesized that if data-dependent permutation is used to permute the CA states $S^{(t)} = [s_0^{(t)}, s_1^{(t)}, ..., s_{n-1}^{(t)}]$ at each clock *(t)*, these correlations can be destroyed since the output sequences now contains states from other registers in an unpredictable manner, e.g. permuting the states of a 5-bit CA $S^{(t)} = [s_0^{(t)}, s_1^{(t)}, s_2^{(t)}, s_3^{(t)}, s_4^{(t)}]$, $S^{(t+1)} = [s_0^{(t+1)}, s_3^{(t+1)}, s_1^{(t+1)}, s_2^{(t+1)}, s_4^{(t+1)}]$, $S^{(t+2)} = [s_2^{(t+2)}, s_0^{(t+2)}, s_4^{(t+2)}, s_3^{(t+2)}, s_1^{(t+2)}]$, ….

We implemented the GRP [16] data-dependent permutation operation on a main-CA's states by using a secondary control-CA (see Fig.3). The main-CA states are permuted by dividing them into two groups depending on the states of the corresponding control-CA states. The main-CA states corresponding to a '0' in the control bit are placed to the left group while a '1' in the control bit are placed to the right group.

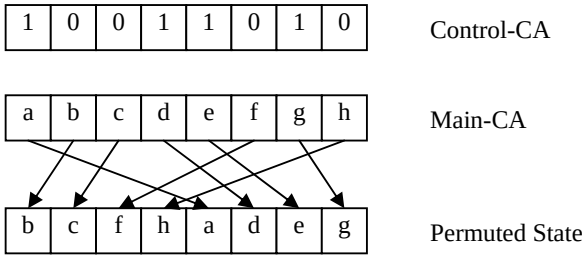

Fig. 3. GRP operation implemented using a two-CA model

B. Dynamic sampling

At each clock some main-CA register states are sampled for output while the unsampled states are not used. Sampling can be fixed, where only certain registers are

selected for output all the time. Dynamic sampling on the main-CA states is performed using a secondary control-CA (see Fig. 4). A ‘1’ in the control CA state means the corresponding register in the main-CA is sampled for output while a ‘0’ mean its state is discarded.

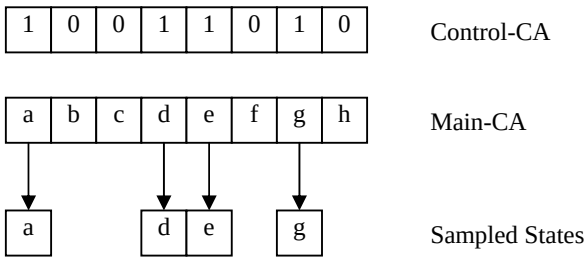

Fig. 4. Sampling operation implemented using 2 CA

In both permutation and sampling, the main-CA and control-CA are similar length m-CA since these low-cost configurations are readily available [5].

IV. WEIGHTED VECTOR PERMUTATION SAMPLING (WVPS)

The basic WVPS scheme incorporates both permutation and sampling, and it comprises of three *n*-bit entities: the main-CA, the control-CA and an array of binary registers V (see Fig. 5). Firstly, both main-CA and control-CA states are updated at every clock. The state of V is then permuted according to the control-CA states. Only the main-CA registers corresponding to ‘1’ in V are sampled for output. By permuting V, different main-CA registers are sampled at each clock and any linearity or correlation in the sampled sequence are hypothesized to be destroyed.

To generate a constant number of m bits from a n-bit CA at every clock (where $m \le n$), V is initialized with a binary vector of Hamming weight $wt(V) = m$. Our

scheme allows a fixed number of random bits to be sampled at each clock; this constant output rate avoids the problem of buffer management issues in schemes without a fixed output rate [17]. We now introduce two different schemes to form the required pseudorandom sequences from the sampled states shown in Fig. 5.

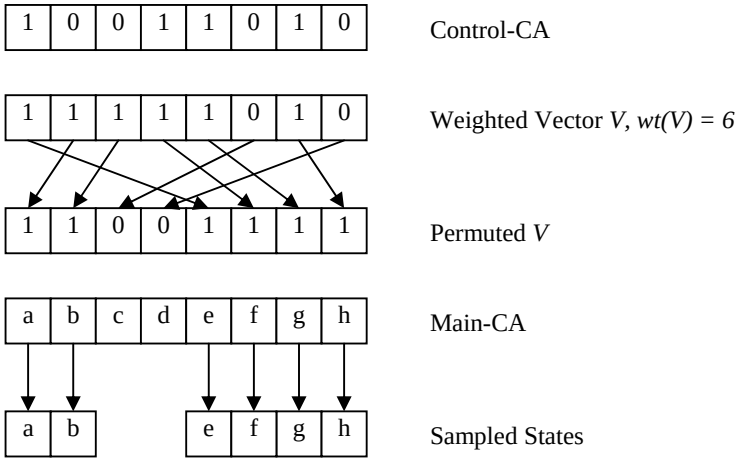

Fig. 5. Operation of the WVPS

A. WVPS with controlled multiplexing (WVPS-CM)

In WVPS-CM, the output is obtained as n single-bit sequences $\{s_j^{(0)}, s_j^{(1)}, s_j^{(2)}, ...\}$, where $0 \le j \le n-1$. At every clock, only m bits can be assigned from the CA registers, so the n sequences receive the sampled bit each in round-robin manner. It thus takes at least n/m clocks for each of the n sequences to receive one bit. Fig. 6 illustrates the WVPS-ID method with a 6-bit m-CA – by t=3, the single-bit sequences are *{a,a}, {b,c}, {c,d}, {a}, {c}, {e}* respectively.

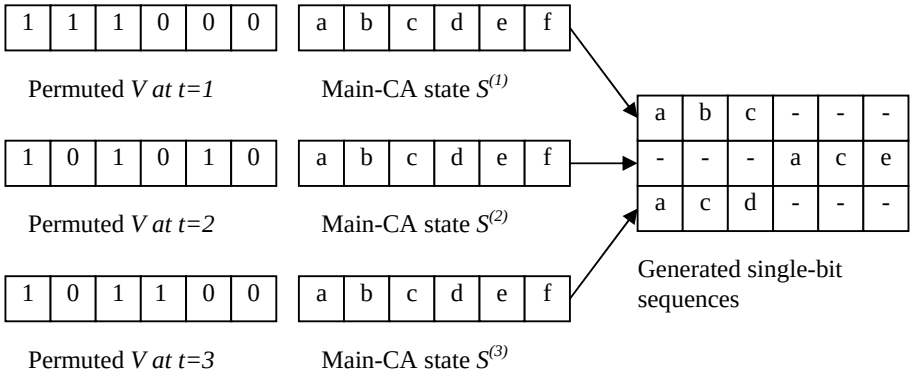

Fig. 6. Operation of the WVPS-CM on a 6-bit CA

B. WVPS with irregular decimation (WVPS-ID)

In WVPS-ID, the output is obtained as n single-bit sequences $\{s_j^{(t_1)}, s_j^{(t_2)}, s_j^{(t_3)}, ...\}$, where $0 \le j \le n-1$. Each sequence consists of only bits drawn from a particular register $s_j^{(t_i)}$ at irregular time instants (t_i) if $v_j^{(t_i)} = 1$ while $s_j^{(t_i)} \mid v_j^{(t_i)} = 0$ are discarded. The result is equivalent to irregular decimation of bits from the original single-bit sequence $\{s_j^{(0)}, s_j^{(1)}, s_j^{(2)}, ...\}$. Fig. 7 illustrates the WVPS-ID method with a 6-bit m-CA – at *t=3*, the single-bit sequences are *{a,a,a}, {b}, {c,c,c,}, {d}, {e}, {-}* respectively.

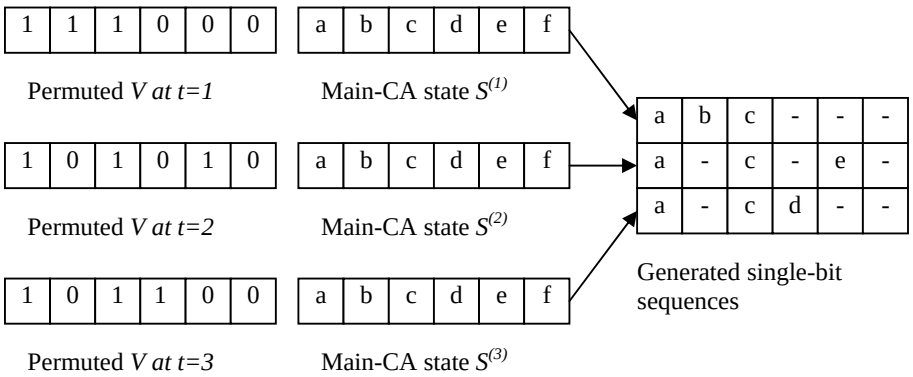

Fig. 7. Operation of the WVPS-ID on a 6-bit CA

V. PERMUTATION PROGRAMMED CA WITH SAMPLING (PPCAS)

Fig. 8. shows each iteration of the PPCAS scheme being described in four discrete steps. Firstly, both the main-CA and control-CA update their states independently at each clock. Then, the new main-CA's state is dynamically permuted by GRP, keyed by the new control-CA state and the permuted bits are then stored as the resultant main-CA state for this clock *(t)*. For convenience, we fixed the leftmost $wt(V) = m$ registers to be selected for sampling. This fixed sampling is reasonable since the leftmost *m* states at each clock are randomly selected by GRP.

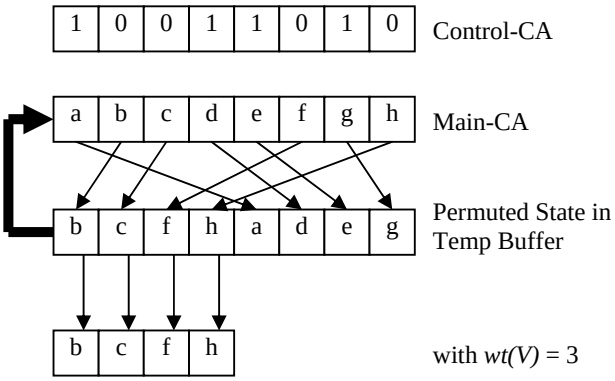

Fig. 8. Operations of the PPCAS

The two main differences between WVPS and PPCAS are:

1) WVPS employs dynamic sampling while PPCAS uses fixed sampling.

2) Main-CA register states in PPCAS are permuted while the weighted vector states in WVPS are permuted.

VI. Experimental Results

A. DIEHARD results

Table 1 shows the 19 DIEHARD tests used for evaluating the randomness quality of generated sequences. Table 2 shows the DIEHARD results for WVPS-CM, WVPS-ID and PPCAS used with 16-, 24- and 32-bit m-CA.

TABLE I. LIST OF DIEHARD TESTS

Index	Test Name		Index	Test Name
1	Overlapping sum		10	Count the ones 2
2	Runs test		11	Bitstream test
3	3D sphere		12	Craps test
4	A parking lot		13	Minimum distance
5	Birthday spacing		14	Overlapping permutation
6	Count the ones 1		15	Squeeze
7	Binary rank 6*8		16	OPSO test
8	Binary rank 31*31		17	OQSO test
9	Binary rank 32*32		18	DNA test
			19	Overall KS test

Both WVPS-CM and WVPS-ID generated some sequences passed all 19 DIEHARD tests. For both WVPS-CM and WVPS-ID, it is clear that using certain *wt(V)* lead to sequences that passed lesser DIEHARD tests. Comparing over the three m-CA examined, WVPS-ID has more consistent results over WVPS-CM, based on the criteria that sequences that passed at least 18 DIEHARD tests should be generated for each m-CA used. WVPS-CM failed this criterion for the 16-bit m-CA. Overall, PPCAS sequences have very consistent DIEHARD results – at least 18 tests are passed in all cases. Adjustment of *wt(V)* does not seem have a significant effect on the DIEHARD results.

TABLE II. NO. OF DIEHARD TESTS PASSED

m-CA length	Sampling weight, *wt(V)*	No. of DIEHARD tests passed (values in brackets shows %improvement over m-CA)		
		WVPS-CM	**WVPS-ID**	**PPCAS**
16	1	5.8 (45%)	18.4 (360%)	18.5 (363%)
	2	10.0 (150%)	14.4 (259%)	18.4 (360%)
	4	9.1 (128%)	18.6 (364%)	18.3 (358%)
	8	10.9 (173%)	18.6 (364%)	18.6 (364%)
	15	7.6 (91%)	18.4 (360%)	18.3 (358%)
24	1	18.4 (75%)	18.3 (74%)	18.3 (74%)
	4	17.0 (62%)	17.8 (70%)	18.5 (76%)
	8	16.8 (60%)	16.0 (52%)	18.0 (71%)
	16	17.7 (68%)	17.8 (70%)	18.8 (78%)
	23	14.6 (39%)	8.5 (-19%)	18.4 (75%)
32	1	18.5 (12%)	18.5 (12%)	18.5 (12%)
	4	18.5 (12%)	18.4 (13%)	18.6 (13%)
	8	18.5 (12%)	17.6 (7%)	18.5 (12%)
	16	18.5 (12%)	16.1 (-3%)	18.6 (13%)
	24	18.3 (11%)	18.3 (11%)	18.4 (11%)
	31	18.4 (13%)	7.7 (-53%)	18.3 (11%)

B. Linear complexity

The linear complexity (denoted *LC*) of an arbitrary sequence is defined as the number of registers required in an equivalent linear system to reproduce that sequence. Unpredictability is often stated in terms of linear complexity since each bit $s^{(t)}$ can be predicted using only the previous LC bits $\{s^{(t-1)}, s^{(t-2)}, ..., s^{(t-LC)}\}$. The Massey-Berlekamp algorithm [17] is used to measure linear complexity from the generated sequences. This algorithm has efficiency $O(n^2)$ and requires $2LC$ bits. For some designs, we have LC increasing exponentially with the PRNG size and testing up to the actual LC is not feasible. We only tested linear complexity for WVPS-ID (better results than WVPS-CM) and PPCAS with 8-bit m-CA due to resource limitations, see Fig. 9.

It is well known that an *n*-cell m-CA has linear complexity n^2; this is why linear sequences are never used directly in cryptography applications. Linear complexity for the 8-bit m-CA is 64 and is shown as a horizontal line in Fig. 9 since the horizontal-axis represents the parameter *wt(V)* which is constant for the m-CA case. The linear complexities of WVPS-ID and PPCAS are substantially improved over the m-CA. Interestingly, the linear complexity for PPCAS seems to increase linearly with *wt(V)*.

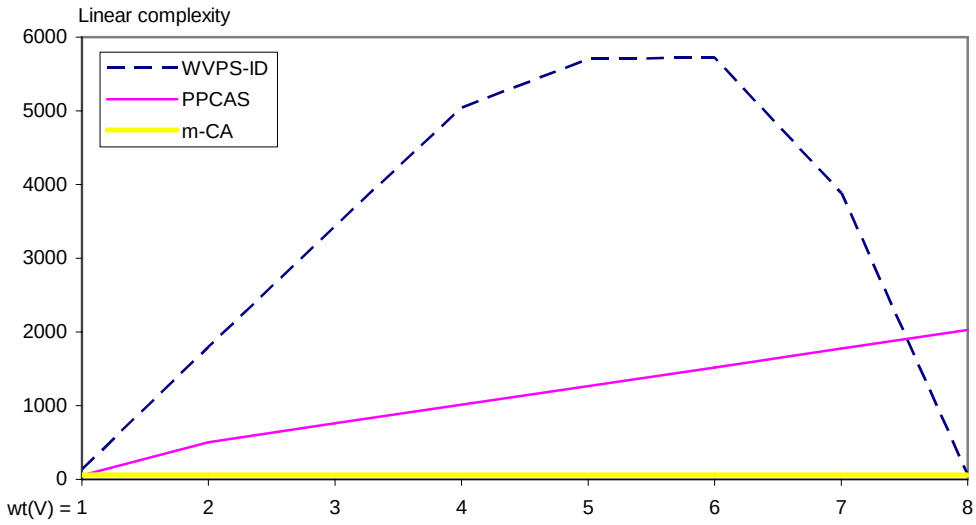

Fig. 9. Linear complexity of generated sequences

The linear complexity for WVPS-ID is shown to be >> n^2 and is highest around the median value of *wt(V)*. Setting $wt(V) = 8$ corresponds to switching off WPVS-ID, and the results are identical to the m–CA case, i.e. $LC_{wt(V)=8} = 64$. At $wt(V) = 1$, only 1 bit is produced per clock thus the period length of the produced sequence will be reduced. Subsequently, linear complexity is similar affected since it is bounded by the period [17]. By increasing $wt(V)$ appropriately, the generated sequences increase in both

period and LC till the maximum at $LC_{wt(V)=5} \approx 5934$. However, when $wt(V)$ is beyond a certain threshold, less linearity will be removed from the sequences despite having a possibly larger period, these sequences are not able to attain high LC.

The *wt(V)* parameter was initially designed into the various methods so that a compromise can be reached between DIEHARD, linear complexity and output efficiency. However, results in Table 2 and Fig. 9 show that compromises are not always required. If an application has the following requirements for pseudorandom sequences: 1) passes most DIEHARD tests, 2) very high linear complexity is not required and 3) high output efficiency, PPCAS is a suitable candidate. If very high linear complexity is required, then WVPS-ID can be attempted with trial values of *wt(V)* to determine an optimal compromise point between the three objectives.

Future work can be carried out in similar methods to increase output efficiency without sacrificing either randomness or linear complexity. One possible direction: a high weight permutation vector and a low weight permutation vector are used alternately.

C. Comparison to previous work

We now provide the published results of some reported CA that passed all DIEHARD tests in the literature. In [8,9], several 8-by-8 two-dimensional (2-d) CA are shown to pass all DIEHARD tests where each register's transformation function is a XOR with at least four inputs from surrounding registers to form the next state. This is extended in [10] to include nonlinear functions. In [12], a 2-d array CA consisting of *m* arrays of *n*-bit 1-d CA with boundary wiring is proposed. 48- and 50-bit versions of this 2-d array

CA are shown to pass all DIEHARD tests. Genetic algorithms are applied in both cases to configure the CA. A wide range of results is not available from these authors although it is mentioned the 2-d CA must have at least 7-by-7 registers to ensure satisfactory DIEHARD results. In another approach using time varying transformations $\Phi^{(t)}$, several CA [14] (with 36 to 48 registers) are shown to generate *n*-bit sequences passing all 19 DIEHARD tests. However, the sequences from these linear models cannot be used without additional processing to inject nonlinearity. Furthermore, 2-d CA structures are actually equivalent to 1-d with an increased number of inputs and non-local connections.

Both WVPS-ID and PPCAS schemes compares favorably with the above works. More interestingly, the 16-bit m-CA can be used with PPCAS to generate sequences that passed more than 18 DIEHARD tests.

VII. Conclusion

We have shown that permutation and sampling can be used in different combinations to improve the randomness quality of sequences generated from maximum length CA. These operations offer an alternative direction to existing work on techniques to improve randomness quality. The linear complexity (i.e. unpredictability) of the various methods is also compared. The suitability of the WVPS-ID and PPCAS schemes are identified in applications based on the objectives: DIEHARD results, linear complexity and output efficiency.

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] D. E. Knuth, “The Art of Computer Programming, Vol. 2: Seminumerical Algorithms”, 3rd ed., Reading, Mass.: Addison-Wesley, 1998.

[3] P. Hellekalek, “Good Random Number Generators Are (Not So) Easy to Find”, In Mathematics and Computer in Simulation, Vol. 46, pp. 485-505, 1998.

[4] G. Marsaglia, “Diehard”, http://stat.fsu.edu/~geo/diehard.html, 1998.

[5] K. Cattell and S. Zhang, “Minimal Cost One-Dimensional Linear Hybrid Cellular Automata of Degree Through 500”, Journal of Electronic Testing: Theory and Applications, Kluwer Academic Publishers, Boston, Vol. 6, pp. 255-258, 1995.

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

[7] P. D. Hortensius, R. D. Mcleod, Werner Pries, D. Michael Miller and H. 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.

[8] 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.

[9] M. Tomassini and M. Perrenoud, “Cryptography With Cellular Automata”, Applied Soft Computing, Vol. 1, pp. 151 – 160, 2001.

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

[11] Sheng-Uei Guan and Shu Zhang, “An Evolutionary Approach to the Design of Controllable Cellular Automata Structure for Random Number Generation”, IEEE Transactions on Evolutionary Computation, Vol. 7, pp. 23 -36, Feb 2003.

[12] 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, pp. 378-388, Mar 2004.

[13] Sheng-Uei Guan and Shu Zhang, “Pseudorandom Number Generation Based on Controllable Cellular Automata”, Special issue: Advanced services for Clusters and Internet computing, Future Generation Computer Systems, Vol. 20, pp. 627-641, 2004.

[14] 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, pp. 1095-1101, Jul 2004.

[15] A. Martín del Rey, J. Pereira Mateus and G. Rodríguez Sánchez, “A Secret Sharing Scheme based on Cellular Automata”, Applied Mathematics and Computation, Vol. 170, Issue 2, pp. 1356-1364, Nov 2005.

[16] R. B. Lee, R. L. Rivest, M. J. B. Robshaw, Z. J. Shi, and Y. L. Yin, “On Permutation Operations in Cipher Design,” Proceedings of the International Conference on Information Technology (ITCC), Vol. 2, pp. 569-577, Apr 2004.

[17] A.J. Menezes, P.C. van Oorschot and S.A. Vanstone, “Handbook of Applied Cryptography”, CRC Press, Boca Raton, 1997.

[18] Shannon C. E, “Communication Theory of Secrecy Systems," Bell System Technical Journal, Vol. 28, pp. 656-715, 1949.

[19] A. Shamir, “On the Security of DES”, Advances in Cryptology – CRYPTO’85, LNCS Vol. 218, pp. 280-281, 1985.