Method for Generating Pseudorandom Sequence of Permutations Based on Linear Congruential Generator

Emil Faure[1,2], Eugene Fedorov[1], Iryna Myronets[1] and Svitlana Sysoienko[1]

[1] *Cherkasy State Technological University, Shevchenko Blvd., 460, Cherkasy, 18006, Ukraine*
[2] *The State Scientific and Research Institute of Cybersecurity Technologies and Information Protection of the State Service for Special Communications and Information Protection of Ukraine, M. Zaliznyaka Str., 3, bl. 6, Kyiv, 03142, Ukraine*

Abstract

The results of the study of the graph of states of a linear congruential generator (LCG) are considered and theoretically substantiated. A model of a generalized graph of LCG states has been developed. It represents each connected component of the graph in the form of cycles equipped with tree products, allows classifying the types of connectivity components of the graph of LCG states and investigating the influence of parameters on its topology. A method for generating a pseudorandom sequence (PRS) of numbers based on the linear congruential method is presented. This method allows generating uniformly distributed numbers regardless of the topology of the graph of LCG states and, consequently, minimizing the time spent on choosing its parameters, and increasing the size of the space of their allowable values to achieve the maximum period. Computer implementation of the algorithm for generating PRS of permutations based on LCG with any type of graph of its states has allowed increasing the speed of the generator compared to the permutation generator using the modern Fisher-Yates algorithm.

Keywords

Pseudorandom sequence, permutation, shuffle, random interleaver, linear congruential generator, graph of states, monad, topology, monad graph

1. Introduction

Pseudorandom sequences (PRSs) [1-2] are widely used to solve a wide class of problems. In particular, they are used to protect information from unauthorized access [3-5], to control the integrity of information [6], to form signals that provide covert data transmission and implement the modeling of complex systems and objects [7-9], to form permutations for factorial coding of information [10-12].

The linear congruential generator, the linear-feedback shift register (LFSR), and the additive generator are the simplest ones in design, high performance and ones of the most commonly used generators. Their design is often the basis of high-quality pseudorandom number generators (PRNGs) with a long repetition period.

Linear congruential generator (LCG) is proposed by D. H. Lehmer in 1949 [13]. It implements a recurrent relation ($f: S \to S$ transition function) of the form:

$$s_i = \left| K \cdot s_{i-1} + C \right|_M , \tag{1}$$

where K is the multiplier;
C is the growth;
M is the module, $K, C, s_0 \in \mathbf{Z}_M$.

CMIS-2022: The Fifth International Workshop on Computer Modeling and Intelligent Systems, Zaporizhzhia, Ukraine, May 12, 2022
EMAIL: e.faure@chdtu.edu.ua (E. Faure); fedorovee75@ukr.net (E. Fedorov); i.myronets@chdtu.edu.ua (I. Myronets); s.sysoienko@chdtu.edu.ua (S. Sysoienko)
ORCID: 0000-0002-2046-481X (E. Faure); 0000-0003-3841-7373 (E. Fedorov); 0000-0003-2007-9943 (I. Myronets); 0000-0002-0009-337X (S. Sysoienko)

Obviously, $s_i \in [0; M-1]$, and the power of the space of LCG states $|S| = M$.

Congruential sequence always forms repeating cycles [1].

LCGs are very sensitive to changes in parameters. Numerous works on the theory and application of congruential generators are aimed at choosing their parameters and assessing the quality of the obtained PRSs. Among them are both classical [1, 14-17] and modern works [18-22].

At the same time, despite the large number of studies on the choice of LCG parameters and experimental evaluation of its properties, most of them are aimed at improving the "randomness" of the formed sequence and do not take into account the structure of space of LCG states S.

Among the works devoted to the analysis of the structure of the space of LCG states, one of the main ones is the thorough scientific work of G. Marsaglia [23]. In addition, the work [24] is devoted to the development of the theory of PRS construction based on the LCG and LFSR.

The analysis of the simplest PRNGs, LCG and LFSR, shows their limitations due to the need to select parameters in order to ensure necessary PRS statistical properties. In particular, the allowable values of the generator parameters to ensure the maximum PRS period are shown in Table 1.

Table 1
PRNG parameters to achieve the maximum PRS period

Method	Period	Valid values of PRNG parameters	Size of space of permissible values of PRNG parameters
Linear congruential method [13]	$T = M$	1) $Gcd(C, M) = 1$; 2) $\|K-1\|_p = 0$ for $\forall$ simple $p : \|M\|_p = 0$; 3) if $\|M\|_4 = 0 \Rightarrow \|K-1\|_4 = 0$	$\varphi(M) \cdot P$, where P is the number of K values that satisfy conditions 2 and 3
Method based on LFSR [25-27]	$T = 2^n - 1$	Generator polynomial $G_n(x)$ is primitive one	Corresponds to the number of primitive polynomials of n degree
Method for PRS generating based on concatenation of LCG cycles [24]	$T = M$	$Gcd(K, M) = 1$	$\varphi(M) \cdot M$
Method for PRS generating based on concatenation of LFSR cycles [24]	$T = 2^n$	Generator polynomial $G_n(x)$ generates a cyclic structure of the graph of LFSR states	Corresponds to the number of polynomials of n degree that generate the cyclic structure of the graph of LFSR states

The purpose of the work is to generate PRS of permutations with high performance and necessary statistical properties without the need to select LCG parameters.

For the further study and analysis of PRNG construction based on sequential traversal of all vertices of the graph of LCG states, it is necessary to perform an in-depth study of the structure of the graph of LCG states, extended analysis of the influence of LCG parameters on the structure of its graph, and, consequently, to develop the method for generating PRS based on linear congruential method by sequentially traversing the contour of the graph of states of the generator.

2. Review of the literature

To study and generalize the structure of the graph of LCG states, we shall explore the main approaches to forming graphs of states of PRS generating devices.

2.1. Cycle graphs

A cycle graph [28], also known as a simply n-cycle [29], is a graph containing n nodes and consisting of a single cycle that passes through all its nodes. The cycle graph is denoted by C_n. The number of vertices in C_n is equal to the number of edges, each vertex has a power of 2, any vertex is incidental to two edges.

Cycle graphs are used, for example, to illustrate the structure of multiplicative groups M_n (Figure 1). Such graphs are formed by creating numbered nodes, one for each α element of the surplus class, and constructing cycles obtained by calculating α^i for $i=1,2,\ldots$. Each edge of such a graph has a bidirectional character [28].

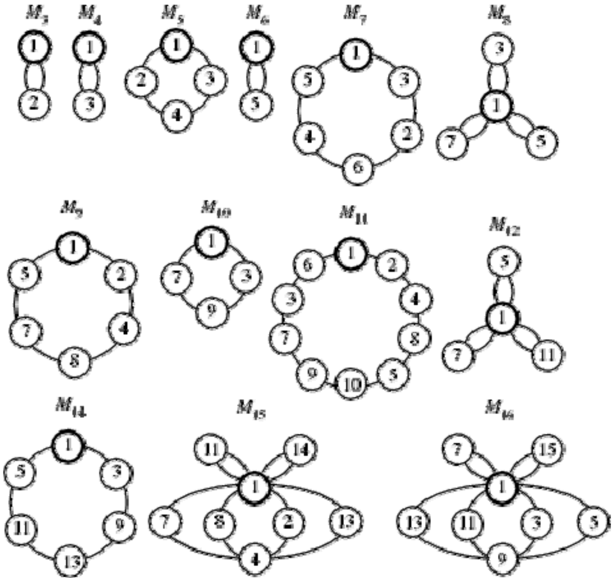


Figure 1: Graphs for some small-order multiplicative groups (from [30])

In all graphs of Figure 1 the node with $\alpha=1$ is highlighted because it is a zero cycle: $\beta_j=\left|\alpha^j\right|_M=\alpha=1$ for any j. Next, we shall use the same notation to represent zero cycles in the graph of LCG states.

Note that in Figure 1 not all represented graphs are cycle graphs. This follows from the fact that multiplicative group by M module can be isomorphic to the product of several cyclic groups (for example, $M_8 \leftrightarrow C_2 \times C_2$, and $M_{15} \leftrightarrow C_2 \times C_4$). In this case, the graph is a combination of several cycle graphs.

To visually represent the structure of the graph of LCG states, it is necessary to use an oriented cycle graph, an oriented version of the cycle graph, in which all arcs are directed in the same direction.

2.2. Algebra of monads and topology of monad graphs

This paragraph, as well as its name, is based on the V. I. Arnold works [31-32].

According to [31], a monad is a representation of a finite set in itself. The monad graph has all elements of this finite set as vertices, and oriented edges connect each element with its image.

In other words, a monad graph is an arbitrary finite oriented graph, from each vertex of which exactly one edge emerges. Iterations of the monad lead any vertex to a cycle attractor.

According to [31], each connected component of the monad graph is a forest of root-oriented root trees, the roots of which are connected by an oriented cycle (topologically circular) from the edges connecting the tree roots.

In other words, connected components of any monad are cycle attractors, which are equipped with root trees attached by their roots to each vertex of the cycle attractor [32]. The number of vertices of the cycle can be equal to 1. In this case, the whole component is one root tree.

Each connectivity component of the graph of any representation of a finite set contains one and only one cycle.

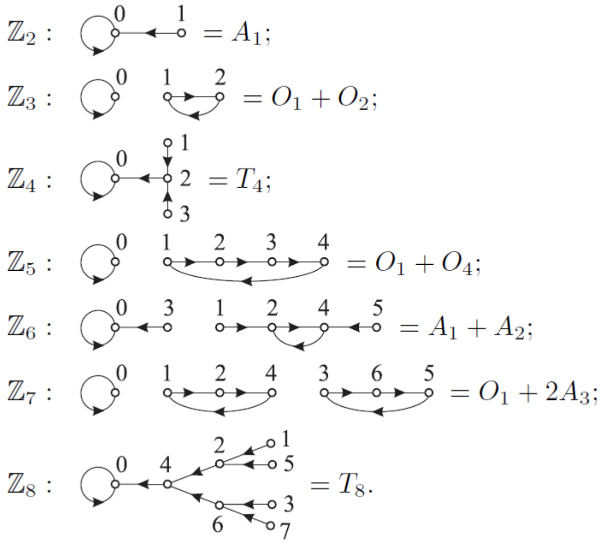
Let S be a finite group, and $f: S \rightarrow S$ representation transforms each of its elements $s \in S$ according to the expression: $f(s)=|K \cdot s+C|_M$, where K is the multiplier; C is the growth; M is the module, $K, C, s_0 \in Z_M$.

In this paper, $f: S \rightarrow S$ representation will be called the monad of S group.

According to [31], the symbols O_n, A_n, T_m, E_n will denote the following oriented graphs:

- O_n = oriented cycle of n vertices;
- A_n = connected graph of $2n$ vertices, which is a cycle of n length, equipped with n single-edge trees, which are included one in each of n vertices;
- T_{2^n} = root tree with $2n$ vertices and n floors except the root, which branches binarially on $1,\ldots,n-1$ floors; the root is considered to be the zero floor, and it also includes two edges: one is from itself and one is from a single vertex of the first floor;
- E_n = root tree with n vertices, from each of which the edge leads directly to the root (so that $E_2 = A_1 = T_2$);
- D_n = $4n$-vertex graph, consisting of O_n cycle of n length, equipped in each of its vertices with three input edges (form together with this corresponding to the cycle vertex the root tree $D_1 = E_4$).

For example, graphs of monads for additive cyclic groups in the field Z_n have the form shown in Figure 2.



$\mathbb{Z}_2$: 0 1 $= A_1;$
$\mathbb{Z}_3$: 0 1 2 $= O_1 + O_2;$
$\mathbb{Z}_4$: 0 1 2 3 $= T_4;$
$\mathbb{Z}_5$: 0 1 2 3 4 $= O_1 + O_4;$
$\mathbb{Z}_6$: 0 3 1 2 4 5 $= A_1 + A_2;$
$\mathbb{Z}_7$: 0 1 2 4 3 6 5 $= O_1 + 2A_3;$
$\mathbb{Z}_8$: 0 4 2 1 5 6 3 7 $= T_8.$

Figure 2: Graphs of monads for some simple cyclic additive groups (from [31])

According to [31], a monad that acts on the direct product $X * Y$ component by component: $(A * B)(x, y)=(Ax * By)$ is called the $A * B$ product of A and B monads that act on X and Y, respectively. The number of elements of a monad product is equal to the product of the number of elements of monad coefficients.

In [31] it is also shown that the graph of the monad product is the product of graph coefficients: $\left[graph(A*B)\right]=\left[graph(A)\right]*\left[graph(B)\right]$ / $A_n = A_1 * O_n$, $D_n = D_1 * O_n$.

Multiplying any root tree T by O_n equips n-cycle O_n with root trees of T type with roots at all points in the cycle.

3. Materials and methods

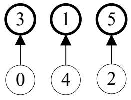
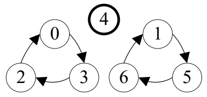
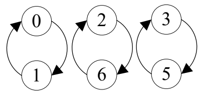
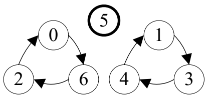
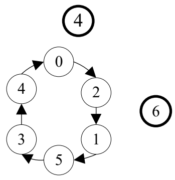
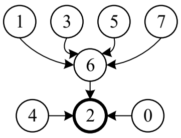
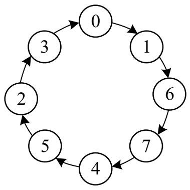
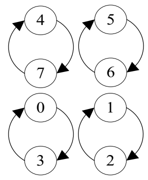
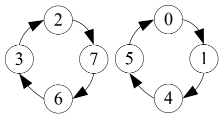
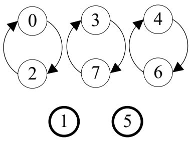
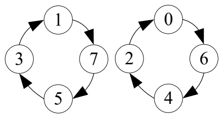
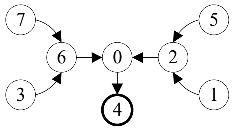
In this section, we shall investigate the LCG topology and generalize the graph of its states to develop a method for generating permutation sequences.

3.1. Graphs of linear congruential generator

Examples of graphs of monads of S group for some LCG parameters are shown in Table 2.

Table 2

Oriented graphs of LCG states for some of its parameters

LCG parameters			Graph of states	LCG parameters			Graph of states
M	K	C		M	K	C	
6	4	3		7	2	3	
7	6	1		7	4	6	
7	3	2		8	4	2	
8	5	1		8	7	3	
8	3	1		8	7	2	
8	1	6		8	6	4	

Let us analyze the structures of graphs of monads of LCG S group. We shall summarize the analyzed structures and present in Table 3 some graphs typical to LCG.

Extending the regularities given in [24], we give some graphs of LCG states and parameters for which these graphs are typical:

1. O_M – for:

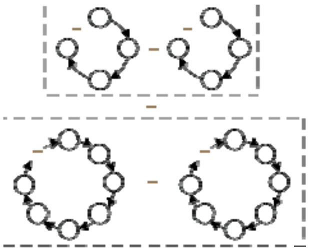
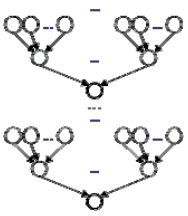
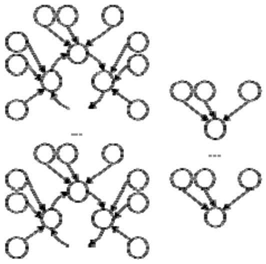
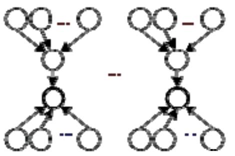
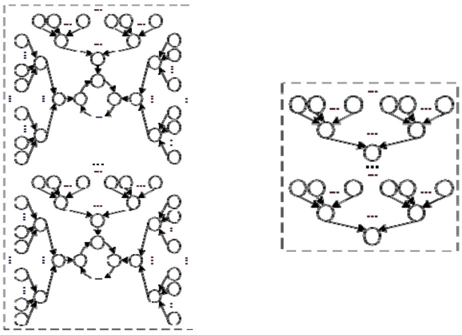
a) $M \geq 2$, $K = 1$, $C = 1$;

b) $M = 2^p$, $K = 4l + 1$, $C = 2m + 1$, $l, m \in Z \geq 0$;

c) M is simple, $K = 1$, $C \geq 1$;

Table 3

Generalized graphs of LCG states

	1. Generalized graph of cycles	
Groups of cycles of t_i length $\left(i \geq 1, t_i \geq 1, \sum_i d_i t_i = M\right)$		$\sum_i d_i O_{t_i}$
	2. Generalized graph of trees	
Group of a^n - vertex trees with n floors $\left(a \geq 2, da^n = M\right)$		dT_{a^n}
	3. Combinations of cycles and trees	
A group of cycles of t length, equipped in each of its vertices with input edges $(n-1)$, and a group of root trees with n vertices, from each of which the edge leads directly to the root $\left(n(dt + k) = M\right)$		$d\left(E_n * O_t\right) + kE_n$ $\left(d\left(T_{n^1} * O_t\right) + kT_{n^1}\right)$
A group of zero cycles with single-edge trees included in them, equipped in each of their vertices with root trees with n vertices, from each of which the edge leads directly to the root $\left(dn = M/2\right)$		$d\left(E_n * A_1\right)$ $\left(d\left(T_{n^1} * T_{2^1}\right)\right)$
A group of cycles of t length, equipped with t a^n-vertex trees with n floors, and a^n group of a^n - vertex trees with n floors $\left(n \geq 2, a^n\left(dt + k\right) = M\right)$		$d\left(T_{a^n} * O_t\right) + kT_{a^n}$

2. dO_t – for $M = 2^p$, $K = 4l - 1$, $l \in \mathrm{Z} \geq 1$, and $C = 2m + 1$, $m \in \mathrm{Z} \geq 0$. Under these conditions:

a) for $l = 2^{k-2}$ $\left(K = 2^k - 1\right)$, $k \geq 2$, $t = 2M/(K+1)$, and $d = (K+1)/2$ (or $t = 2^{p-l+1}$, $d = 2^{l-1}$) take place;

b) for $l = 2k - 1$ $\left(K = 8k - 5\right)$, $k \geq 1$, $t = M/2$, and $d = 2$ take place;

c) for $l = 2k$, $k \geq 1$, $K \neq 2^r - 1$, $r \geq 3$ (that is $k \neq 2^{r-3} : k = 2^{i-4}(2j+1)$, and $l = 2^{i-3}(2j+1)$ $\left(K = 2^{i-1}(2j+1) - 1\right)$ for $j \in \left[1; 2^{p-i} - 1\right]$, $i \in [4; p-1]$), $t = M/2^{i-2}$, $d = 2^{i-2}$ take place;

3. $dO_t + O_1$ – for simple M, $K \geq 2$, $C \in \mathrm{Z}$;

4. a tree with a root, zero cycle, for $M = 2^p$, $K \in \left\{2l : l \in \mathrm{N}, 0 < l < 2^{p-1}\right\}$, $C \in \left\{0, 1, \ldots, 2^p - 1\right\}$, $p \in \mathrm{N}$

3.2. Generalized graph of states of linear congruential generator

Analysis of possible graphs of LCG states shows that they can all be reduced to a single configuration containing a set of d cycles of the same or different length, including zero cycles, and a set of d' precycles (trees) leading to cycles. Under these conditions

$$M = \sum_{i=1}^{d} t_i + \sum_{j=1}^{d'} t_j^{'} , \qquad (2)$$

where t_i is the length of the i^{th} cycle;

$t_j^{'}$ is the number of vertices in the j^{th} tree except for the root.

The generalized LCG graph can be represented as follows:

$$G_{LCG} = \left(V_{LCG}, A_{LCG}\right),$$

where V_{LCG} is the set of vertices of the graph;

A_{LCG} is the set of arcs of the graph.

In its turn,

$$V_{LCG} = \{v_k\} \cup \{v_l^{'}\},$$

where $\{v_k\}$ is the set of vertices belonging to the cycles, $k = 1, 2, \ldots, \sum_{i=1}^{d} t_i$;

$\{v_l^{'}\}$ is the set of vertices belonging to the trees, except for their roots, $l = 1, 2, \ldots, \sum_{j=1}^{d'} t_j^{'}$;

$$A_{LCG} = \{a_k\} \cup \{a_l^{'}\},$$

where $\{a_k\}$ is the set of arcs belonging to the cycles, $k = 1, 2, \ldots, \sum_{i=1}^{d} t_i$;

$\{a_l^{'}\}$ is the set of arcs belonging to the trees, $l = 1, 2, \ldots, \sum_{j=1}^{d'} t_j^{'}$.

According to [33], for a simple M, the expressions $d' = 0$, $t_i = t$ for $\forall i[1, d-1]$ and $t_d = 1$ are fair, and expression (2) takes the form: $M = \sum_{i=1}^{d-1} t + 1$.

We present a generalized graph of LCG states in terms of the theory of algebra of monads and the topology of monad graphs.

The analysis of typical oriented graphs of LCG states shows that no more complex patterns, except for the products of trees and cycles, are found in the graphs of monads of $f : S \to S$ representation. LCG graph is an inconnected combination of cycles equipped with tree products. Since $E_n = T_{n^1} * O_1$, $A_t = T_{2^1} * O_t$, $D_t = T_{4^1} * O_t$, each connected component of LCG graph can be represented as

$T_{a^n} * \left(T_{b^m} * O_t\right)$. For example, $O_t = T_{a^0} * \left(T_{b^0} * O_t\right)$, and $A_t = T_{a^0} * \left(T_{2^1} * O_t\right) = T_{2^1} * \left(T_{b^0} * O_t\right)$, $E_n = T_{a^0} * \left(T_{n^1} * O_1\right) = T_{n^1} * \left(T_{b^0} * O_1\right)$.

Then the generalized graph of LCG states has the form:

$$G_{LCG} = \sum_{i=1}^{d} d_i \left(T_{a_i^{n_i}} * \left(T_{b_i^{m_i}} * O_{t_i}\right)\right), \tag{3}$$

where d is the number of different types of connectivity components of the graph of LCG states;

d_i is the number of connectivity components of the graph of LCG states of the B th type;

a_i, n_i, b_i, m_i, t_i are parameters of connectivity components of the graph of LCG states of the i th type.

In this case $\sum_{i=1}^{d} d_i a_i^{n_i} b_i^{m_i} t_i = M$.

Determining the rules for calculating the number of graph components $\sum_{i=1}^{d} d_i$, their types d and values of numbers a_i, n_i, b_i, m_i, t_i through LCG parameters is beyond the scope of this work and requires further research.

3.3. Method for generating sequences of permutations based on linear congruential method

The method for generating LCG-based PRS is as follows.

1. If necessary (for example, to increase the speed of PRS generating or meet the requirements for the spatial complexity of the algorithm that implements the proposed method), the type of graph of LCG states, as well as the conditions to be met by K , C and M LCG parameters to obtain a given type of structure are determined. Determining the type of the graph of LCG states can be performed in accordance with typical graphs presented in Table 3. M parameter determines the area of determination of pseudorandom variable. If the choice of the type of the graph of states is not made, LCG parameters are determined arbitrarily, taking into account the restrictions imposed on them.
2. If necessary (for example, for non-consecutive cycles of the graph of LCG states without precycles (trees) to increase the speed of PRS generating), the representatives of each cycle of the generator (boot vectors (BVs)) are determined and stored in memory.
3. The current LCG BV is determined by (random or deterministic) choosing from the set of stored BVs, if this set is specified, or from the set of integers in the range $\left[0, M-1\right]$.
4. A PRS is formed by the LCG with given parameters until the generator forms unique numbers. In the case of reappearance of any element (not necessarily equal to BV (for a graph containing continuous cycles without precycles (trees), equal to BV)) the generation of the current segment of the sequence is stopped.
5. A new current BV of LCG is determined by its (random or deterministic) choosing:

- from the set of still unused BVs, if this set is specified;
- from the set of integers of the range $\left[0, M-1\right]$ except for the numbers present in the formed part of the PRS.

6. PRS is formed for a given BV until the reappearance of the element in the formed sequence (for a graph containing continuous cycles without precycles (trees), equal to the current BV).
7. Transition to item five until the shuffle of all BVs.
8. Transition to item three until the shuffle of all combinations consistently used in items three and five of BV (if they are set and stored in memory).
9. Transition to item two until the shuffle of all BV combinations (if they are set and stored in memory).

Thus, the proposed method allows performing concatenation not only of separate and disjoint cycles in LCG graph, but also of precycles (trees), if they are contained therein.

In addition, the proposed approaches can be used to form PRS based on LFSR with an arbitrary generator polynomial. This is because the use of a reducible polynomial as a generator one for LFSR leads to an increase in the number and the change in the structure of connected components in the generator graph of states.

4. Experiments and results

The proposed approaches to constructing devices for PRS generating based on LCG are used to create software implementations of generators.

The size of the space of allowable values of LCG parameters for the above method is equal to M^2. The comparison with the corresponding indicators of the analogues in table 1 shows that this method allows increasing the size of the space of allowable values of LCG parameters to achieve the PRS period $T = M$ in $M/\varphi(M)$ times.

Let us study the speed of software implementation of the permutation generator based on the developed method and compare it with the speed of the generator that implements the modern Fisher-Yates algorithm [34]. For the sake of objectivity, the generators were implemented on one platform and tested on one computer with fixed performance indicators. The results are presented in Figure 3.

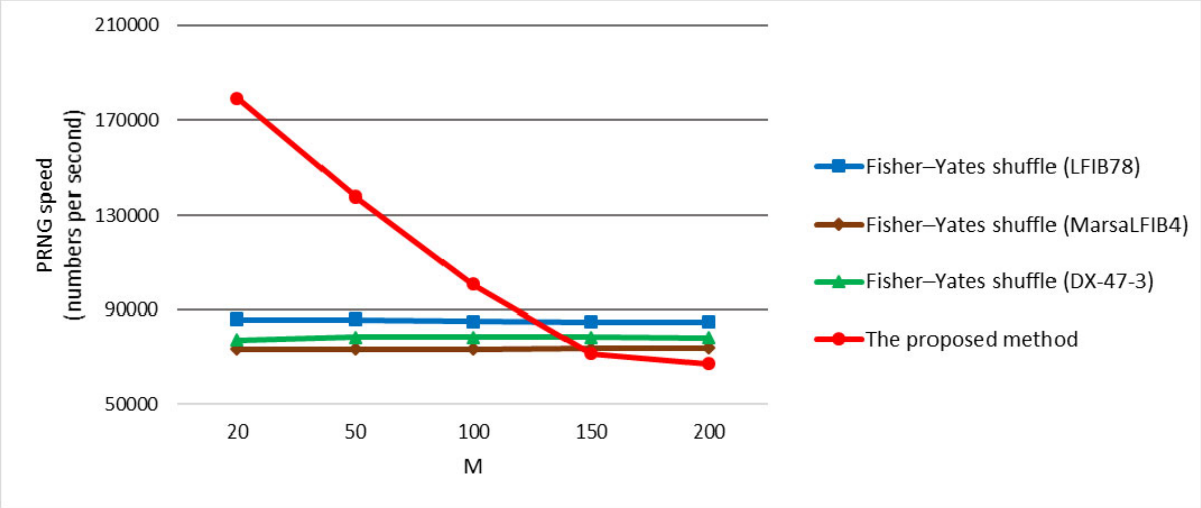


Figure 3: Graphs of dependence of speed of permutation generators on M value

The speed of the developed generator exceeds the speed of the permutation generator using the Fisher-Yates algorithm for $M \leq 125$, which expands the results obtained in [35].

It should be noted that PRS formed according to the proposed method is not cryptographically stable and can not be used in "pure" form in cryptographic transformations, for example, as a gamma for stream ciphers. However, the proposed approaches to PRS generating can be used to implement a multi-stage encryption procedure.

5. Conclusions

The scientific novelty of the study is as follows. It is developed the model for generalized graph of states of linear congruential generator, which allows to carry out the classification of types of connectivity components of the graph of its states and to investigate the influence of parameters on its topology. The developed model has allowed to improve the method for PRS generating based on linear congruential method, which allows to form PRS of uniformly distributed numbers regardless of the topology of the graph of states of linear congruential generator and, as a result, to minimize the time spent on choosing its parameters and increase the size of the space of their allowable values to achieve the PRS period $T = M$ in $M/\varphi(M)$ times.

Implementation of the algorithm for generating PRS of permutations based on LCG with any type of the graph of its states has allowed to increase the speed of the generator compared to the permutation generator based on PRNG LFIB78 using the Fisher-Yates algorithm for the permutation order $M \le 125$: in particular, for $M = 20$ – in 2.1 times; $M = 50$ – 1.6 times; $M = 100$ – 1.2 times.

6. Acknowledgements

This research was funded by the Ministry of Education and Science of Ukraine, grant number 0120U102607.

7. References

[1] D. E. Knuth, The Art of Computer Programming, Vol. 2: Seminumerical Algorithms, 3rd ed., Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[2] F. James, L. Moneta, Review of high-quality random number generators, Computing and Software for Big Science 4 (1) (2020). doi: 10.1007/s41781-019-0034-3.

[3] L. Crocetti, S. Di Matteo, P. Nannipieri, L. Fanucci, S. Saponara, Design and test of an integrated random number generator with all-digital entropy source, Entropy 24 (2) (2022). doi: 10.3390/e24020139.

[4] X. Tong, X. Chen, S. Xu, Advances in superlattice cryptography research, [超晶格密码的研究进展] Kexue Tongbao/Chinese Science Bulletin 65 (2-3) (2020) 108-116. doi 10.1360/TB-2019-0291.

[5] S. Sysoienko, I. Myronets, V. Babenko, Practical implementation effectiveness of the speed increasing method of group matrix cryptographic transformation, in: CEUR Workshop Proceedings, volume 2353, 2019, pp. 402-412.

[6] S. Gnatyuk, V. Kinzeryavyy, K. Kyrychenko, Kh. Yubuzova, M. Aleksander, R. Odarchenko, Secure hash function constructing for future communication systems and networks, Advances in Intelligent Systems and Computing 902 (2020) 561-569.

[7] M. Alawad, M. Lin, Survey of stochastic-based computation paradigms, IEEE Transactions on Emerging Topics in Computing 7 (1) (2019) 98–114. doi 10.1109/TETC.2016.2598726.

[8] S. Bandyopadhyay, R. Bhattacharya, Discrete and Continuous Simulation: Theory and Practice, CRC Press, 2014. doi: 10.1201/b17127.

[9] P. A. A. Resende, A. C. Drummond, A survey of random forest based methods for intrusion detection systems, ACM Computing Surveys 51 (3) (2018). doi: 10.1145/3178582.

[10] J. S. Al-Azzeh, B. Ayyoub, E. Faure, V. Shvydkyi, O. Kharin, A. Lavdanskyi, Telecommunication systems with multiple access based on data factorial coding, International Journal on Communications Antenna and Propagation 10 (2) (2020) 102-113. doi: 10.15866/irecap.v10i2.17216.

[11] E. Faure, A. Shcherba, Y. Vasiliu, A. Fesenko, Cryptographic key exchange method for data factorial coding, in: CEUR Workshop Proceedings, volume 2654, 2020, pp. 643–653.

[12] E. Faure, A. Shcherba, B. Stupka, Permutation-based frame synchronisation method for short packet communication systems, in: Proceedings of the 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021, volume 2, 2021, pp. 1073–1077. doi: 10.1109/IDAACS53288.2021.9660996.

[13] D. H. Lehmer, Mathematical methods in large-scale computing units, in: Proceedings of a Second Symposium on Large-Scale Digital Calculating Machinery, Cambridge, Mass., 1949, pp. 141–146.

[14] W. Freiberger, U. Grenander, A Short Course in Computational Probability and Statistics, Springer, New York, 1971.

[15] M. Greenberger, An a priori determination of serial correlation in computer generated random numbers, Mathematics of Computation 15 (76) (1961) 383.

[16] T. E. Hull, A. R. Dobell, Random number generators, SIAM Review 4 (3) (1962) 230-254.

[17] C. F. Gauss, J. Brinkhuis, C. Greiter, Disquisitiones Arithmeticae, reissue ed., Springer, New York, 1986.

[18] E. V. Faure, A. I. Shcherba, V. M. Rudnytskyi, The method and criterion for quality assessment of random number sequences, Cybernetics and Systems Analysis 52 (2) (2016) 277-284. doi: 10.1007/s10559-016-9824-3.

[19] K. Tsuchiya, Y. Nogami, Long period sequences generated by the logistic map over finite fields with control parameter four, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E100.A (9) (2017) 1816-1824. doi: 10.1587/transfun.E100.A.

[20] E. Faure, I. Myronets, A. Lavdanskyi, Autocorrelation criterion for quality assessment of random number sequences, in: CEUR Workshop Proceedings, volume 2608, 2020, pp. 675–689.

[21] P. L'Ecuyer, P. Wambergue, E. Bourceret, Spectral analysis of the MIXMAX random number generators, INFORMS Journal on Computing 32 (1) (2020) 135-144. doi: 10.1287/IJOC.2018.0878.

[22] G. Konsam, M. Loukrakpam, Triple linear congruential generator-based hardware-efficient pseudorandom bit generation, in: T. R. Lenka, D. Misra, A. Biswas (Eds.), Micro and Nanoelectronics Devices, Circuits and Systems, volume 781 of Lecture Notes in Electrical Engineering, Springer, Singapore, 2022. doi: 10.1007/978-981-16-3767-4_22.

[23] G. Marsaglia, The structure of linear congruential sequences, in: S. K. Zaremba (Ed.), Applications of Number Theory to Numerical Analysis, Academic Press, 1972, pp. 249-285.

[24] A. Lavdanskyi, Methods and Tools of Forming Pseudorandom Sequences for Computer Cryptography, Ph.D. thesis, Cherkasy State Technological University, Cherkasy, Ukraine, 2017.

[25] R. C. Tausworthe, Random numbers generated by linear recurrence modulo two, Mathematics of Computation, 19 (90) (1965).

[26] S. W. Golomb, Shift Register Sequences: Secure and Limited-Access Code Generators, Efficiency Code Generators, Prescribed Property Generators, Mathematical Models, 3rd revised ed., Aegean Park Press, Laguna Hills, CA, USA, 2017.

[27] H. T. Nguyen, G. N. Pham, A. N. Bui, B. A. Nguyen, N. T. Le, H. T. Pham, Linear feedback shift register and its applications in digital system design, International Journal of Emerging Technology and Advanced Engineering 11 (11) (2021) 204-208. doi: 10.46338/IJETAE1121_24.

[28] D. Shanks, Solved and Unsolved Problems in Number Theory, 4rd. ed., AMS Chelsea Pub., New York, 2002.

[29] S. Pemmaraju, P. S. Skiena, Computational Discrete Mathematics, Combinatorics and Graph Theory with Mathematica, 1st ed., Cambridge University Press, Cambridge, U.K., New York, 2003.

[30] Modulo Multiplication Group, (25.02.2022).
URL: http://mathworld.wolfram.com/ModuloMultiplicationGroup.html.

[31] V. I. Arnold, Topology of algebra: Combinatory of squaring, Functional Analysis and Its Applications 37 (3) (2003) 177-190.

[32] V. I. Arnold, Topology and statistics of formulae of arithmetics, Uspekhi Matematicheskikh Nauk 58:4 (352) (2003) 3-28.

[33] T. V. Mityankina, V. V. Shvydkiy, A. I. Shcherba, Randomization of a sequence of congruential numbers, Bulletin of the Engineering Academy of Ukraine 2 (2008) 107-111.

[34] Paul E. Black, Fisher-Yates shuffle, in Dictionary of Algorithms and Data Structures (2019). URL: https://www.nist.gov/dads/HTML/fisherYatesShuffle.html.

[35] F. Panca Juniawan, H. Arie Pradana, Laurentinus, D. Yuny Sylfania, Performance comparison of linear congruent method and Fisher-Yates shuffle for data randomization, Journal of Physics: Conference Series, 1196 (1) (2019). doi:10.1088/1742-6596/1196/1/012035.