ADBA Computer Science
Cutting-Edge Scientific Solutions

e-ISSN: 3023-8544
RESEARCH ARTICLE
Vol.3 / No.1 / 2026 / pp.1-5
https://doi.org/10.69882/adba.cs.2026011

Determination of Optimal Linear Congruence Generators Parameters with Heuristic Optimization Algorithms

Sinem Akyol [*,1] **and Fatih Özkaynak** [*,α,2]
[*]Fırat University Department of Software Engineering Elazığ, Türkiye, [α]Kriptarium R&D Software Consulting Defense Industry and Trade Ltd. Co., 23119, Elazığ, Türkiye.

ABSTRACT Random number generators play a critical role in many scientific and engineering applications, particularly in simulation, cryptography, and optimization. Among these generators, Linear Congruential Generators (LCGs) are widely used due to their simplicity and computational efficiency. However, the statistical quality of the generated sequences strongly depends on the proper selection of generator parameters. In this study, the determination of optimal LCG parameters is formulated as a heuristic optimization problem. The Improved Grey Wolf Optimizer (IGWO) is employed to search for suitable multiplier, increment, and seed values. The proposed approach aims to achieve a uniform distribution of generated numbers while maintaining low correlation between consecutive values. The performance of the optimized LCG is evaluated using the objective fitness function as well as additional statistical performance metrics derived from the generated sequences. The effectiveness of the proposed IGWO-based optimization approach is demonstrated through repeated independent runs using the same fitness evaluation framework. Experimental results demonstrate that the proposed approach provides improved parameter selection for LCGs and enhances the statistical properties of the generated random sequences.

KEYWORDS
Linear congruential generator
Random number generation
Heuristic optimization
Grey wolf optimizer
Statistical randomness

INTRODUCTION

Random number generation constitutes a fundamental component of modern scientific computing and engineering applications (Bikos *et al.* 2023). Pseudo-random number generators (PRNGs) are extensively used in Monte Carlo simulations, numerical integration, stochastic differential equations, optimization algorithms, and uncertainty quantification frameworks (Zhao *et al.* 2023; Ozkaynak and Yavuz 2013; Park *et al.* 2022). In these domains, the statistical quality of generated random sequences directly influences the accuracy, stability, and reproducibility of computational results (Zhao *et al.* 2023; Zhao and Ma 2024). Consequently, the design, analysis, and evaluation of reliable random number generators remain an active and critical area of research. Monte Carlo methods, in particular, rely heavily on high-quality pseudo-random sequences to approximate solutions to problems that are analytically intractable (Li *et al.* 2024; Irfan *et al.* 2020). The convergence rate and variance of Monte Carlo estimators are strongly affected by the uniformity and independence properties of the underlying random numbers (Foreman *et al.* 2024).

Deficiencies in randomness may introduce systematic bias, slow convergence, or misleading confidence intervals, ultimately compromising the validity of simulation-based studies (Ferreira *et al.* 2023). Similar concerns arise in numerical methods where random sampling is employed, such as randomized algorithms for linear algebra, probabilistic numerical schemes, and sensitivity analyses (Foreman *et al.* 2024; Ferreira *et al.* 2023; Álvarez *et al.* 2022). Random number generation also plays a significant role in machine learning and stochastic optimization techniques. Algorithms such as stochastic gradient descent, evolutionary computation, particle swarm optimization, and reinforcement learning depend on randomness for exploration, parameter initialization, and probabilistic decision-making (Maksymovych *et al.* 2022).

Poor statistical properties may lead to premature convergence, reduced diversity, or unstable learning dynamics, especially in large-scale or high-dimensional problems. While randomness is also essential in cryptographic applications, the requirements for cryptographic security differ fundamentally from those of simulation-oriented randomness (Maksymovych *et al.* 2022). Cryptographically secure random number generators demand resis-

Manuscript received: 1 December 2025,
Revised: 15 January 2026,
Accepted: 19 January 2026.

[1]sakyol@firat.edu.tr
[2]ozkaynak@firat.edu.tr(**Corresponding author**).

tance against adversarial prediction and reconstruction, whereas simulation-focused PRNGs prioritize statistical quality, reproducibility, and computational efficiency (Ozkaynak 2014; Nannipieri *et al.* 2021; Crocetti *et al.* 2022). As emphasized by Knuth, Marsaglia, and L'Ecuyer, a generator that is suitable for simulation purposes may be entirely inadequate for cryptographic use, and vice versa (Ozkaynak 2020; Rojas-Muñoz *et al.* 2022). Therefore, it is essential to clearly distinguish between these application domains when evaluating and optimizing random number generators (Liu *et al.* 2021). Extensive research has demonstrated that many widely used PRNGs exhibit hidden structural weaknesses that may only become apparent under rigorous statistical testing.

Marsaglia's Diehard tests and L'Ecuyer's TestU01 framework have revealed that generators passing basic theoretical criteria can still fail advanced statistical evaluations (Nam *et al.* 2022; Hajduk 2024). These findings underscore the necessity of systematic parameter selection and empirical validation, particularly for generators intended for large-scale simulations and numerical experiments (Almaraz Luengo and Román Villaizán 2023). In this context, improving the statistical reliability of simulation-oriented random number generators remains a crucial challenge (Ozkaynak 2015). Rather than relying solely on classical parameter choices or theoretical constraints, modern approaches increasingly emphasize empirical performance evaluated through comprehensive test suites. In these contexts, the quality of pseudo-random sequences impacts both the efficiency of the optimization process and the robustness of learned models. Therefore, the design and evaluation of random number generators remain an active research topic.

Linear Congruential Generators (LCGs) are among the simplest and most commonly used pseudo-random number generators due to their ease of implementation and low computational cost (Steele Jr. and Vigna 2022). An LCG generates a sequence of numbers based on a recurrence relation defined by a multiplier, an increment, a modulus, and an initial seed value (Steele Jr. and Vigna 2022). Despite their simplicity, LCGs are known to suffer from statistical weaknesses when their parameters are not carefully selected (L'Ecuyer 1999). Poor parameter choices may result in short periods, non-uniform distributions, and strong correlations between successive numbers. Although LCGs have been extensively studied in the literature, many existing works primarily focus on theoretical conditions for achieving maximum period lengths or optimize parameters using a single statistical criterion (Panda and Ray 2020). However, optimizing LCG parameters based solely on one performance measure may lead to suboptimal random behavior when other statistical properties are taken into account. In practice, the quality of a random number generator should be evaluated using multiple criteria, such as uniformity and independence, simultaneously. In this study, the parameter selection problem of LCGs is formulated as a heuristic optimization problem. The Improved Grey Wolf Optimizer (IGWO) is employed to search for suitable LCG parameters. The main contributions of this work can be summarized as follows:

- The formulation of the LCG parameter selection problem as a heuristic optimization task.
- The application of the Improved Grey Wolf Optimizer (IGWO) to determine suitable generator parameters.
- The evaluation of the optimization performance through multiple independent runs using a consistent fitness function.

Due to the scope of the present study, the analysis is limited to a single heuristic optimization method and a fitness-based evaluation, while broader comparative and multi-metric analyses are left for future investigations. The remainder of this paper is organized as follows. Section 2 introduces the LCG model, the proposed optimization framework and the experimental results, and Section 3 concludes the paper.

LCG MODEL AND THE PROPOSED OPTIMIZATION FRAMEWORK

Heuristic optimization algorithms are a class of methods inspired by natural systems, designed to solve complex problems by incorporating heuristic knowledge into the problem-solving process. This section explains the fundamental principles of heuristic optimization algorithms and their application in optimizing random number generators.

Intelligent heuristic optimization algorithms are widely used to solve complex real-world problems due to their simplicity and high performance (Akyol 2022). Many of these problems can be modeled as optimization problems, and heuristic methods can be quickly adapted as solution-seeking strategies Qiu *et al.* (2024). Heuristic optimization algorithms are developed based on inspiration from various disciplines, including genetic algorithms, particle swarm optimization, simulated annealing, and harmony search (Yu *et al.* 2024). Their core principle is to work iteratively on a population of candidate solutions in the problem domain to find the best solution. These algorithms evaluate the current solutions at each step and use these evaluations to create new generations of solutions (Chen and Zheng 2024). The process concludes with the maximization or minimization of a specific criterion or objective function (Rajwar *et al.* 2023). These algorithms are classified into nine categories based on their sources of inspiration: music-based, mathematics-based, swarm-based, social-based, biology-based, chemistry-based, sports-based, physics-based, and hybrid methods combining these approaches (Akyol and Alatas 2017). Among these, biology-based and swarm-based metaheuristic methods are frequently preferred for search and optimization problems. In this study, an improved version of the Grey Wolf Optimizer, a swarm-based heuristic algorithm inspired by the hunting strategies of grey wolves, has been utilized (Mirjalili *et al.* 2014).

Heuristic optimization algorithms are commonly used in optimizing random number generators. Specifically, they can be effectively employed to determine and improve the parameters of random number generators such as LCGs. Heuristic optimization algorithms optimize an objective function, which, in the case of LCG parameter optimization, typically evaluates the statistical properties of the generated random numbers. The objective function aims to maximize uniform distribution, minimize serial correlation, and achieve the desired period of random numbers. Heuristic optimization algorithms explore the parameter space and use an iterative approach to find the best parameter combination. This process is crucial to ensure that the random number generator possesses the desired statistical properties.

In this study, the Improved Grey Wolf Optimizer (IGWO) proposed by Nadimi-Shahraki et al. Nadimi-Shahraki *et al.* (2021) was used to determine LCG configurations with good statistical properties. IGWO is a swarm-based metaheuristic method inspired by the hunting skills and leadership hierarchies within packs of grey wolves. In the IGWO method, which draws on the hunting behavior and natural social leadership of grey wolves, there are four types of wolves. The wolves in the best positions are categorized as α, β, and δ wolves, while the remaining ω wolves are guided by these leaders. The hunting process of the wolves consists of seven stages: encircling, hunting, attacking prey, initiating, moving, selecting, and updating (Mirjalili *et al.* 2014). The hunting strategy is illustrated in Figure 1.

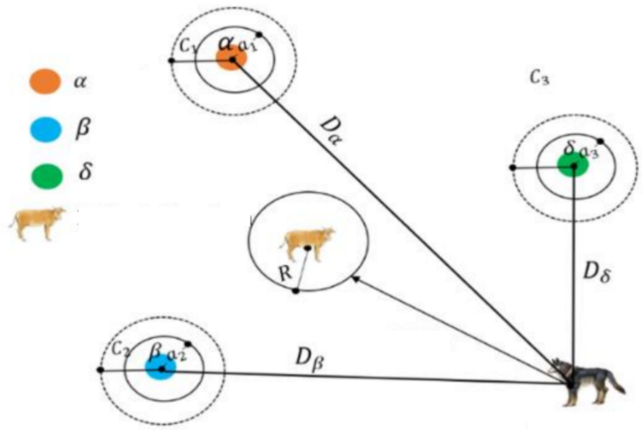

Figure 1 Hunting strategies of gray wolves

Encirclement: This is the phase where the prey is surrounded by grey wolves and is expressed by Eq. (1) and Eq. (2).

$$\vec{D} = |\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)| \tag{1}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{2}$$

Where $\vec{X}_p$ represents the location of the prey, t represents the current iteration, $\vec{X}$ represents the location vector of the gray wolf, and the coefficients $\vec{A}$ and $\vec{C}$ are calculated with the equations in Eq. (3) and Eq. (4) (Mirjalili *et al.* 2014).

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \tag{3}$$

$$\vec{C} = 2 \cdot \vec{r}_2 \tag{4}$$

Each time the algorithm is run, the element values of the $\vec{a}$ vector are linearly reduced from 2 to 0. $\vec{r}_1$ and $\vec{r}_2$ are random vectors that take values in the range $[0, 1]$.

Hunting: In this stage, which mathematically models the hunting behavior of wolves, ω wolves follow α, β and δ wolves, which are assumed to know the location of the prey best. The equations for the hunting stage are given in Eq. (5)-(7) (Nadimi-Shahraki *et al.* 2021).

$$\vec{D}_\alpha = |\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}|, \quad \vec{D}_\beta = |\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}|, \quad \vec{D}_\delta = |\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}| \tag{5}$$

Where $\vec{D}_\alpha$, $\vec{D}_\beta$, and $\vec{D}_\delta$ are calculated using Eq. (4).

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha, \quad \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta, \quad \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot \vec{D}_\delta \tag{6}$$

Here, $\vec{X}_\alpha$, $\vec{X}_\beta$, and $\vec{X}_\delta$ are the three best candidate solutions at the t-th iteration. $\vec{X}_1$, $\vec{X}_2$, and $\vec{X}_3$ are calculated using Eq. (6).

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{7}$$

Attack: The hunting process of wolves ends when the prey stops moving and then the attack process begins. This process is mathematically expressed as the decrease of the $\vec{a}$ value from 2 to 0 throughout the iterations.

Initialization phase: In this phase, N wolves are randomly created in the search space as shown in Eq. (8) (Nadimi-Shahraki *et al.* 2021).

$$X_{i,j} = lb_j + rand \times (ub_j - lb_j) \tag{8}$$

Where dim represents the dimension of the problem. The fitness value of X_i is evaluated according to the fitness function representing the location of the i-th wolf in the t-th iteration.

Movement phase: In addition to the group hunting strategy, the dimensional learning-based hunting (DLH) strategy has been added for individual hunting in this phase. In the DLH phase, each wolf is learned by neighboring wolves as another candidate for the new current location of X_i (Nadimi-Shahraki *et al.* 2021).

DLH search strategy: This strategy, which is used to prevent the diversity in the population from decreasing, also takes into account other wolves in the population when updating the location. In the DLH search strategy, the location of the wolf X_{i-DLH} is calculated using Eq. (11). A randomly selected wolf from the population and different neighbors are used when updating the location. In this strategy, another candidate solution called X_{i-DLH} is produced for the wolf X_i in addition to X_{i-GWO}. First, the value of $R_i(t)$, which is the Euclidean distance between the candidate solution $X_i(t)$ and $X_{i-GWO}(t)$, is calculated using Eq. (9) (Nadimi-Shahraki *et al.* 2021).

$$R_i(t) = ||X_i(t) - X_{i-GWO}(t)|| \tag{9}$$

Then, the neighbors of $X_i(t)$ indicated by $N_i(t)$ are found using Eq. (10) depending on the radius of $R_i(t)$.

$$N_i(t) = \{X_j(t) \mid D_i(X_i(t), X_j(t)) \leq R_i(t), \quad X_j(t) \in Pop\} \tag{10}$$

Here, D_i represents the distance between X_i and X_j. After the neighborhood $N_i(t)$ is created, multi-neighbor learning is calculated using Eq. (11). The d-th dimension of X_{i-DLH} is calculated using the neighbor $X_{n,d}(t)$ randomly selected from $N_i(t)$ and the randomly selected $X_{r,d}(t)$ from the population.

$$X_{i-DLH,d}(t+1) = X_{i,d}(t) + rand \times (X_{n,d}(t) - X_{r,d}(t)) \tag{11}$$

Selection and update phase: This is the phase where the most suitable individual is determined according to the fitness values of two candidates, X_{i-GWO} and X_{i-DLH}.

$$X_i(t+1) = \begin{cases} X_{i-GWO}(t+1) & \text{if } f(X_{i-GWO}) < f(X_{i-DLH}) \\ X_{i-DLH}(t+1) & \text{otherwise} \end{cases} \tag{12}$$

Finally, if the fitness value of the obtained X_{new} is better than X_i, the value of X_i is updated as the new location. Otherwise, X_i remains unchanged. After all these operations are applied to each candidate solution in the population, the iteration number is increased by one and the process repeats until the termination conditions are met (Nadimi-Shahraki *et al.* 2021). The flow diagram of IGWO is shown in Figure 2.

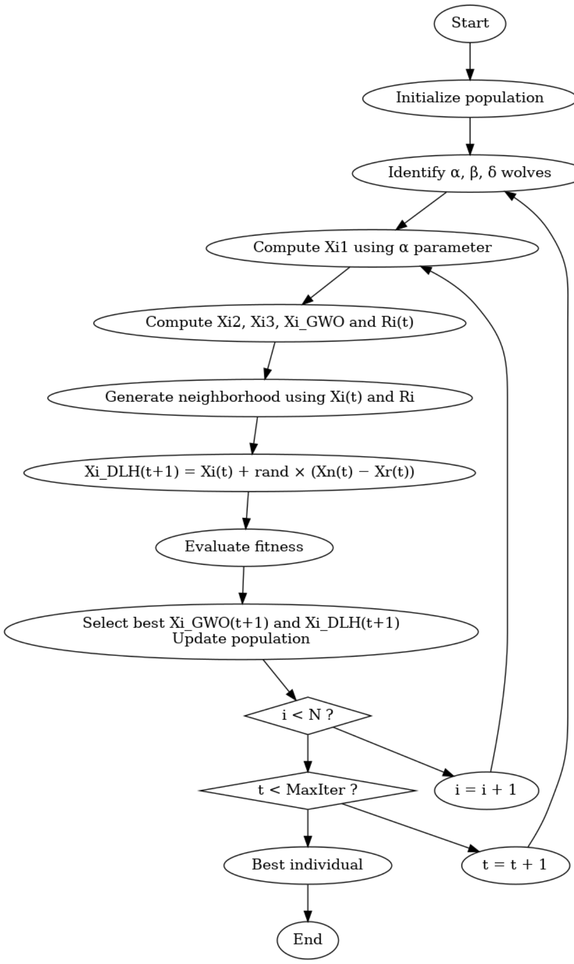

Figure 2 Flow chart of the IGWO algorithm

The a, c and X_0 values in the formula used by the LCG model constitute the decision variables of IGWO. The n value is taken as a constant of 10. 30 individuals are produced as the initial population and the termination condition of the algorithm is determined as 100 iterations. The aim of this study is to ensure that when 10000 numbers are generated, each number comes at an equal rate. Since the n value is taken as 10, the aim is to find the a, c and X_0 values that will ensure that each of the numbers from 0 to 9 comes approximately 1000 times. For this purpose, Eq. (13) is used as the fitness function.

$$Minimize\ F = \sum_{k=0}^{n-1} (Count_k - 1000)^2 \tag{13}$$

The optimization results obtained using IGWO are analyzed based on the best and average fitness values over 50 independent runs. The results are shown in Table 1. In each column of the table, the values of a, c and X_0 are given respectively. In all runs, all numbers between 0-9 were obtained 1000 times.

Table 1 50 a, c and X_0 values that gave the optimum result

a, c, X_0	a, c, X_0	a, c, X_0	a, c, X_0	a, c, X_0
6, 1, 5	1, 3, 5	7, 7, 5	7, 7, 9	1, 9, 3
5, 5, 7	4, 7, 3	1, 7, 4	1, 1, 2	1, 9, 4
1, 3, 6	1, 1, 5	1, 9, 10	8, 1, 10	6, 9, 2
1, 3, 2	7, 7, 3	8, 1, 3	1, 1, 4	6, 6, 7
6, 6, 1	5, 7, 0	1, 7, 5	3, 7, 5	4, 1, 1
2, 5, 4	5, 7, 0	1, 1, 9	1, 9, 3	1, 9, 4
1, 3, 8	5, 7, 4	1, 3, 10	2, 1, 6	1, 7, 7
1, 7, 6	1, 1, 3	5, 7, 8	1, 3, 2	1, 7, 3
5, 1, 5	8, 9, 6	5, 1, 7	1, 1, 1	1, 3, 5
1, 7, 9	1, 3, 8	5, 1, 7	1, 9, 2	1, 7, 1

The determined optimal LCG configurations tend to have the desired uniform distribution, low seriality level, and long period. The obtained results will increase the reliability of random number generators used in scientific calculations and simulations. LCG configurations with good statistical properties are important for obtaining more accurate and repeatable results. The experimental results obtained from multiple independent runs indicate that IGWO provides a stable and effective search mechanism for optimizing LCG parameters under the defined fitness function. The results indicate that guided heuristic optimization can provide more reliable parameter selections for LCGs.

CONCLUSION

In this study, the problem of selecting suitable parameters for Linear Congruential Generators was formulated as a heuristic optimization task. The Improved Grey Wolf Optimizer was employed to search for optimal multiplier, increment, and seed values. Unlike approaches based on a single evaluation criterion, the proposed framework evaluates generator quality using multiple statistical performance metrics in addition to the objective fitness function.

Although the present study focuses on the optimization of LCG parameters using the Improved Grey Wolf Optimizer and evaluates performance based on a fitness-driven criterion, several directions remain open for future research. First, the proposed framework can be extended by incorporating additional heuristic optimization algorithms, such as genetic algorithms or particle swarm optimization, in order to perform a comprehensive comparative analysis. Second, the evaluation methodology may be enriched by employing multiple statistical performance metrics, including goodness-of-fit tests and serial correlation measures, to provide a more detailed assessment of randomness quality. Finally, future studies may benefit from expanding the reference framework by systematically integrating recent advances in heuristic optimization and random number generation literature. These extensions are expected to further strengthen the robustness and generalizability of the proposed approach.

Acknowledgments

This study was supported by Fırat University Scientific Research Projects Coordination Unit (FUBAP) with project number MF.24.52, TEKF.24.27 and TÜBİTAK with project number 123R055.

Ethical standard

The authors have no relevant financial or non-financial interests to disclose.

Availability of data and material

Not applicable.

Conflicts of interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

LITERATURE CITED

Akyol, S., 2022 A new hybrid method based on aquila optimizer and tangent search algorithm for global optimization. Journal of Ambient Intelligence and Humanized Computing **13**: 1–21.

Akyol, S. and B. Alatas, 2017 Plant intelligence based metaheuristic optimization algorithms. Artificial Intelligence Review **47**: 417–462.

Almaraz Luengo, E. and J. Román Villaizán, 2023 Cryptographically Secured Pseudo-Random Number Generators: Analysis and Testing with NIST Statistical Test Suite. Mathematics **11**: 4812.

Bikos, A., P. E. Nastou, G. Petroudis, and Y. C. Stamatiou, 2023 Random Number Generators: Principles and Applications. Cryptography **7**: 54.

Chen, S. and J. Zheng, 2024 A hybrid grey wolf optimizer for engineering design problems. Journal of Combinatorial Optimization **47**: 86.

Crocetti, L., S. Di Matteo, P. Nannipieri, L. Fanucci, and S. Saponara, 2022 Design and Test of an Integrated Random Number Generator with All-Digital Entropy Source. Entropy **24**: 139.

Ferreira, M. J., N. A. Silva, A. N. Pinto, and N. J. Muga, 2023 Statistical Validation of a Physical Prime Random Number Generator Based on Quantum Noise. Applied Sciences **13**: 12619.

Foreman, C., R. Yeung, and F. J. Curchod, 2024 Statistical Testing of Random Number Generators and Their Improvement Using Randomness Extraction. Entropy **26**: 1053.

Hajduk, Z., 2024 Field-Programmable Gate Array-Based True Random Number Generator Using Capacitive Oscillators. Electronics **13**: 4819.

Irfan, M., A. Ali, M. A. Khan, M. Ehatisham-ul Haq, S. N. Mehmood Shah, *et al.*, 2020 Pseudorandom Number Generator (PRNG) Design Using Hyper-Chaotic Modified Robust Logistic Map (HC-MRLM). Electronics **9**: 104.

L'Ecuyer, P., 1999 Tables of linear congruential generators of different sizes and good lattice structure. Mathematics of Computation **68**: 249–260.

Li, S., Z. Lin, Y. Yang, and R. Ning, 2024 A High-Performance FPGA PRNG Based on Multiple Deep-Dynamic Transformations. Entropy **26**: 671.

Liu, J., Z. Liang, Y. Luo, L. Cao, S. Zhang, *et al.*, 2021 A Hardware Pseudo-Random Number Generator Using Stochastic Computing and Logistic Map. Micromachines **12**: 31.

Maksymovych, V., M. Shabatura, O. Harasymchuk, R. Shevchuk, P. Sawicki, *et al.*, 2022 Combined Pseudo-Random Sequence Generator for Cybersecurity. Sensors **22**: 9700.

Mirjalili, S., S. M. Mirjalili, and A. Lewis, 2014 Grey wolf optimizer. Advances in Engineering Software **69**: 46–61.

Nadimi-Shahraki, M. H., S. Taghian, and S. Mirjalili, 2021 An improved grey wolf optimizer for solving engineering problems. Expert Systems with Applications **166**: 113917.

Nam, J.-W., J. Kim, and J.-P. Hong, 2022 Stochastic Cell- and Bit-Discard Technique to Improve Randomness of a TRNG. Electronics **11**: 1735.

Nannipieri, P., S. Di Matteo, L. Baldanzi, L. Crocetti, J. Belli, *et al.*, 2021 True Random Number Generator Based on Fibonacci-Galois Ring Oscillators for FPGA. Applied Sciences **11**: 3330.

Ozkaynak, F., 2014 Cryptographically secure random number generator with chaotic additional input. Nonlinear Dynamics **78**: 2015–2020.

Ozkaynak, F., 2015 A novel method to improve the performance of chaos based evolutionary algorithms. Optik **126**: 5434–5438.

Ozkaynak, F., 2020 A Novel Random Number Generator Based on Fractional Order Chaotic Chua System. Elektronika ir Elektrotechnika **26**: 52–57.

Ozkaynak, F. and S. Yavuz, 2013 Security problems for a pseudorandom sequence generator based on the Chen chaotic system. Computer Physics Communications **184**: 2178–2181.

Panda, A. K. and K. C. Ray, 2020 A coupled variable input LCG method and its VLSI architecture for pseudorandom bit generation. IEEE Transactions on Instrumentation and Measurement **69**: 1011–1019.

Park, S., K. Kim, K. Kim, and C. Nam, 2022 Dynamical Pseudo-Random Number Generator Using Reinforcement Learning. Applied Sciences **12**: 3377.

Qiu, Y., X. Yang, and S. Chen, 2024 An improved gray wolf optimization algorithm solving to functional optimization and engineering design problems. Scientific Reports **14**: 14190.

Rajwar, K., K. Deep, and S. Das, 2023 An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges. Artificial Intelligence Review **56**: 13187–13257.

Rojas-Muñoz, L. F., S. Sánchez-Solano, M. C. Martínez-Rodríguez, and P. Brox, 2022 True Random Number Generation Capability of a Ring Oscillator PUF for Reconfigurable Devices. Electronics **11**: 4028.

Steele Jr., G. L. and S. Vigna, 2022 Computationally easy, spectrally good multipliers for congruential pseudorandom number generators. Software: Practice and Experience **52**: 443–458.

Yu, M., J. Xu, W. Liang, *et al.*, 2024 Improved multi-strategy adaptive Grey Wolf Optimization for practical engineering applications and high-dimensional problem solving. Artificial Intelligence Review **57**: 277.

Zhao, W., Z. Chang, C. Ma, and Z. Shen, 2023 A Pseudorandom Number Generator Based on the Chaotic Map and Quantum Random Walks. Entropy **25**: 166.

Zhao, W. and C. Ma, 2024 Modification of Intertwining Logistic Map and a Novel Pseudo Random Number Generator. Symmetry **16**: 169.

Álvarez, R., F. Martínez, and A. Zamora, 2022 Improving the Statistical Qualities of Pseudo Random Number Generators. Symmetry **14**: 269.

How to cite this article: Akyol, S., and Özkaynak, F. Determination of Optimal Linear Congruence Generators Parameters with Heuristic Optimization Algorithms. *ADBA Computer Science*, 3(1), 1-5, 2026.