Received: August 14, 2023. Revised: October 7, 2023.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

An Efficient Cryptanalysis of Nonlinear Stream Cipher Based on Swarm Intelligent System

Mohammed H. Ahmed[1]* Asaad N. Hashim[2] Khalid A. Hussein[1]

[1]*Department of Computer Science, College of Education, Mustansiriyah University, Baghdad, Iraq*
[2]*Department of Computer Science, College of Computer Science and Math, University of Kufa, Najaf, Iraq*
* Corresponding author's Email: mohammedalbawi@uomustansiriyah.edu.iq

Abstract: Cryptography plays a crucial role in safeguarding privacy and confidentiality. Therefore, it is imperative to ensure the security and resistance to cryptanalysis of the employed ciphers. In this paper, an innovative approach called three levels multiple particle swarm optimization (TL-MPSO) has been proposed to enhance the traditional particle swarm optimization (PSO) for cryptanalysis of nonlinear stream ciphers through a ciphertext only attack. We focus on the geffe generator system, which relies on a set of linear feedback shift registers (LFSRs) with a nonlinear combining function. TL-MPSO employs a three-levels approach with distinct search strategies. In the first level, known as the discovery level, particles are randomly distributed into equally sized sub-swarms with no communication between particles in different sub-swarms. The primary objective is to thoroughly explore the search space, with communication limited to neighbouring particles within the same sub-swarm. The second level, called the extraction level, is generated artificially by perturbing the best particles from the discovery level. Particles in different sub-swarms communicate in parallel and co-evolve. Finally, the third level, the optimization level, is artificially generated from the previous level, with all sub-swarms merged into a single large swarm. In all levels, an adaptive movement strategy for each particle is established through a new inertia weight updating process, and a dynamic regrouping strategy is applied to enhance the chances of achieving the global best. Evaluation results demonstrate that the proposed system requires only a minimal number of characters and achieves optimal CPU times. Specifically, when attacking Geffe generators with lengths ranging from 15 to 35, the proposed algorithm consistently achieves optimal solutions with just 10 to 50 cipher characters. Our experiments reveal that TL-MPSO is more accurate and faster than traditional PSO, representing an efficient technique compared to existing cryptanalysis methods.

Keywords: Cryptography, Cryptanalysis, Ciphertext only attack, Particle swarm optimization, Geffe generator.

1. Introduction

Cryptology, the field of study concerned with ensuring the secrecy and authenticity of information, encompasses various tools and methods. It is comprised of two subfields: cryptanalysis and cryptography [1]. Cryptography focuses on designing cryptosystems that safeguard confidentiality and privacy [2], while cryptanalysis deals with the task of breaking or solving these designed cryptosystems. In essence, cryptanalysis can be viewed as an optimization problem [3]. Among the fundamental components within the cryptology system are transposition, substitution, symmetric and asymmetric cryptosystems, as well as stream and block ciphers [4].

Stream ciphers are an integral part of modern cryptosystems, utilizing pseudorandom key bits, known as keystream, to sequentially encrypt plaintext, either bit by bit or byte by byte [5]. The cryptanalysis of stream cipher systems presents a significant challenge, requiring the use of contemporary and advanced methods to conduct successful attacks. Exploring modern and promising techniques is crucial for effectively analyzing and exploiting vulnerabilities in stream ciphers [6].

Intelligent systems are one of the most promising techniques for studying and analyzing stream cipher systems [7]. Swarm intelligent systems are an

essential and important part of intelligent systems, which derive from the biological realm of nature and involve simulating the collective behaviour of animal flocks, such as birds and fish [8]. Many complex real-life problems are solved using swarm intelligence. The particle swarm optimization algorithm is a subfield of swarm intelligence, originally inspired by the collective behavior of bird flocks and fish schools, and it has proven to be a powerful tool for solving optimization problems [9, 10].

Our study introduces a new methodology for analyzing stream cipher systems with nonlinear internal states, leveraging swarm intelligence techniques. We present a unique approach known as three-levels multi-particle swarm optimization (TL-MPSO) to effectively target nonlinear stream ciphers of various lengths using cipher-only attacks. The primary objective of the TL-MPSO algorithm is to efficiently explore extensive solution spaces and find optimal or near-optimal solutions. By harnessing the concept of swarm intelligence, where particles interact and communicate to collectively converge towards optimal solutions, we anticipate that our proposed algorithm can effectively identify weaknesses and vulnerabilities in the targeted stream ciphers.

To demonstrate the effectiveness and validity of our approach in the field of cryptanalysis, we conduct simulations comparing the TL-MPSO algorithm with brute force attacks, traditional particle swarm optimization (PSO), and some of the previous works. These comparisons are made on the target system, as well as related works, to provide a comprehensive evaluation of the proposed system. Through these simulations and comparisons, we aim to showcase the advantages and capabilities of our new approach, highlighting its superiority in terms of efficiency, accuracy, and robustness. This research contributes to the field of cryptanalysis by offering a promising method for attacking stream cipher systems with nonlinear internal states and demonstrating its viability through extensive evaluations.

The remainder of this paper is organized as follows. Section 2 provides an overview of related works that have explored the utilization of intelligent systems in cryptanalysis. Section 3 presents detailed explanations of the Particle Swarm Optimization (PSO) algorithm and the proposed algorithm. In section 4, comprehensive information regarding the cryptographic systems under consideration is provided. The attack on Geffe stream cipher using the TL-MPSO algorithm is introduced in section 5. The analysis of the results obtained from these attacks is discussed in section 6. Finally, the paper concludes with a summary of findings, potential future directions, and areas of further exploration in the last section.

2. Related works

Many previous works have advocated for the application of intelligent systems in the cryptanalysis of stream cipher systems. For instance, H. A. M Al_Sharifi [11] applied PSO to determine the initial state of a stream cipher based on a single LFSR. Their work demonstrated the effectiveness of PSO in attacking stream ciphers, although their primary focus was on LFSRs with relatively short lengths, with polynomial degrees not exceeding 11. Maiya Din and their research team [12] explored two approaches: the divide-and-conquer attack and the genetic algorithm to target the stream cipher based on the Geffe generator. Their proposed system reduced the need for exhaustive searches and efficiently obtained the correct initial states of all shift registers with various lengths. However, their attack duration proved to be relatively extended, exceeding an hour, and they targeted each register individually. In contrast, our study executed a comprehensive system attack in a considerably shorter time frame, focusing on longer registers through the utilization of a larger polynomial attack.

Iwona Polak & Mariusz Boryczka [13] employed optimization methods called Genetic Algorithms (GA) to break the RC4 stream cipher. This work successfully found 59-80% of RC4 keystream bits. Klaus pommerening [14] focused on analyzing NLFSRs based on the BK algorithm in two scenarios. The first scenario occurs when the cryptanalyst has no knowledge of the feedback function, while the second involves the cryptanalyst being aware of an efficient limit to the number of variable coefficients in the feedback function. In the first scenario, the BK algorithm is not a realistic attack, but it is considered a highly efficient algorithm for cryptanalysis in the second scenario. Maiya Din and their colleagues [15] developed an evolutionary algorithm called Cuckoo Search (CS) to solve LFSRs with lengths of [11, 13, 17, 19]. In their study, CS required approximately 200 characters from the ciphertext to determine the seed key of the LFSR with the same polynomial degree. In contrast, our proposed system achieves the same outcome with a significantly reduced requirement of only 20 characters from the ciphertext. Furthermore, our attack demonstrates notable time efficiency compared to the approach presented in the referenced paper.

Salim A. Abbas and Riyam N. J. Kadhum [16] suggestions an improving PSO algorithm using simulated annealing (SA) to attack Geffe stream

cipher based on cipher only attack. The hybridization between PSO and SA successfully recovered the initial values of the Geffe cipher. However, it's important to note that the lengths of the LFSRs in the referenced work were considerably shorter when compared to our research. In the work of Faez Hassan Ali and their collaborators [17], a swarm intelligent algorithm, known as the Bees Algorithm, was used to attack a stream cipher. Their proposed algorithm successfully attacked nonlinear stream ciphers that depend on 3 LFSRs with lengths of 3, 5, and 7. Their approach employed a probable word attack. However, it's important to note that the experimental results in this study were limited by the use of short total lengths for LFSRs. Attack stream cipher using the DNA algorithm produced by Sattar B. Sadkha & Basim S. Yaseen [18]. This study successfully achieved the recovery of the initial state of three FSRs, each with a length of up to 10, requiring only 9 characters from the cipher. However, it's important to note that the computational time was extensive, reaching up to 50 hours. In contrast, our system accomplishes the same task with 10 characters and a significantly reduced computational time of just a few seconds.

In 2019, Maiya Din and their co-authors [19] introduced a novel discrete particle swarm (DPS) algorithm to break the cryptosystem based on the RC4 algorithm. Based on the results obtained, the proposed algorithm was capable of solving RC4 with a key length of 48 bits and a cipher length of up to 300 characters. A nonlinear Geffe generator was attacked based on a swarm intelligent algorithm by Maiya Din and the research team [20]. The B_PSO swarm intelligent algorithm was introduced to find the correct initial state and recover the plaintext when applied to solve ciphertext with a length of up to 400 characters encrypted based on a stream cipher with 3 LFSRs and a polynomial degree of up to 31. This research achieved success in attacking the Geffe cipher, with a total length of up to 59. However, it required a minimum of 400 characters from the cipher, taking more than an hour to complete. In contrast, the system proposed in this paper achieves the same results with a requirement of just 200 characters and significantly less time consumption.

Iwona Polak and Mariusz Boryczka [21] focused on the cryptanalysis of VMPC stream cipher systems using a Tabu search approach and successfully determined the permutation state for various VMPC ciphers through known plaintext attacks. In their study [22], they conducted a significant cryptanalysis of the RC4+ stream cipher. This involved employing a Tabu search approach along with probable words, which required checking an average of 250 possible internal states to identify the correct one. Their experiments demonstrated the effectiveness of this method in decrypting the RC4+ cipher. In their work, Faez Hassan Ali and Riyam Noori Jawad [23] developed genetic algorithm (GA) and ant colony optimization (ACO) techniques for cryptanalyzing the shrinking generator, a nonlinear stream cipher system based solely on ciphertext. The proposed evolutionary algorithms were successful in finding the initial values when the key length was set to 12 or 16. Girish Mishra and colleagues [24] introduced a new method for validating RC4, Trivium, and TRIAD stream ciphers using black-box analysis deep learning. Their approach eliminates the need for prior assumptions or intricate mathematical/statistical analysis. A novel method was proposed by R. M. Rizk-Allah and their collaborators [7] to attack RC4 stream cipher. Combination between B_PSO and EO was applied to find the initial key of RC4 stream cipher and the result illustrated that the proposed system able to recover initial state by visited only 104 internal stages based on probable word attack. R. N. Jawad [25] developed a modification of PSO to attack stream ciphers with different generator systems. The combination of PSO and SA algorithms successfully produced better results than traditional methods when attacking nonlinear stream ciphers based on 3 LFSRs with lengths of 15 and 19. The outcomes of this research were promising, particularly in terms of the time consumed during the attack. However, it's worth noting that the lengths of the LFSRs were relatively short.

In this study, we extend upon the existing body of research in stream cipher cryptanalysis. While previous works have shown the effectiveness of various algorithms in attacking stream ciphers, our research distinguishes itself by addressing several limitations present in prior approaches. Specifically, we propose a novel methodology that employs a comprehensive cipher-only attack to recover the initial states of three linear Feedback shift registers (LFSRs), each with a polynomial degree of up to 29. Furthermore, our approach significantly reduces the required ciphertext length and computational time while successfully handling larger LFSRs. By achieving these advancements, our research contributes to the broader understanding of stream cipher cryptanalysis, offering more efficient and practical solutions for enhancing cryptographic security.

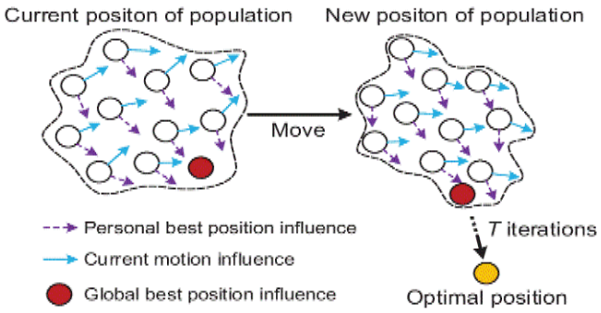


Figure. 1 Framework of traditional PSO algorithm

3. Methods

3.1 Particle swarm optimization (PSO)

In 1995, J. Kennedy & R. C. Eberhart introduced a new approach called particle swarm optimization (PSO) [26]. PSO is a part of swarm intelligence techniques, which fall within artificial intelligence systems and simulate the behavior of animals that exist in groups and have the ability to communicate with each other in a specific environment. Each particle has two specifications: the first is position, denoted by P_i^t, and the other is velocity, denoted by V_i^t. The particles correct their positions and velocities based on the information received from the entire population. Initially, PSO was used to solve nonlinear continuous optimization problems, but over time, it has been applied to discrete optimization problems as well. In the traditional PSO paradigm, a single population is randomly initialized in 2D dimensions, with each particle having a position and velocity representing a candidate solution. At each iteration, the fitness is directly calculated for each particle. The implementation of the traditional PSO algorithm is defined by two equations [27]:

$$V_i^{t+1} = w \cdot V_i^t + c_1 U_1^t \left(P_{b_i}^t - P_i^t\right) + c_2 U_2^t \left(G_b^t - P_{iD}^t\right) \quad (1)$$

$$P_{iD}^{t+1} = P_i^t + v_i^{t+1} \quad (2)$$

where V_i^{t+1} refers to the velocity of particle (i) at time $(t+1)$,the symbol w refers to the inertia weight which is restricted to a range between w^{min} and w^{max} (0.4,0,9) and is calculated based on Eq. (3) [28], G and g refers to the total and current generation respectively, $U_1^t$and $U_2^t$are random values in the range [0, 1], c_1 represents the individual (cognitive) learning rate and c_2 represents the group (social) learning rate, P_i^t refers to the current position where $P_{b_i}^t$ denotes the locally best position and G_b^t refers to the best position of the entire population in the current iteration. Fig. 1 illustrated the framework of the traditional PSO algorithm [29, 30].

$$w = \frac{w^{max} - w^{min} * g}{G} \quad (3)$$

3.2 Three levels adaptive multi swarms particle swarm optimization (TL_MPSO)

In the standard particle swarm optimization (PSO), it is not uncommon to encounter a situation where the swarm reaches a stagnant state when the global best position remains unchanged for several consecutive iterations [26]. To overcome this challenge, introducing new search direction can be beneficial in assisting the swarm to escape from local optima. For clarity and reference, we have compiled a comprehensive list of the parameters and their corresponding definitions utilized throughout the remainder of this paper in Table 1.

This work proposes a division of the total generations of the PSO algorithm into three distinct levels. The first level, called the discovery level, involves the random distribution of particles into predefined sub-swarms. Each particle within a sub-swarm explores its local space, influenced solely by its neighbouring particles within the same sub-swarm. Communication with particles in other sub-swarms is absent, allowing for independent evolution. The primary objective of this level is to extensively explore the search space.

The second level, known as the extraction swarm, is generated by introducing a random perturbation to the best particles from the previous level. It is important to note that the selected 'best particle' may represent a local optimum rather than the global optimum. To enhance the chances of achieving the global best, when the extraction swarm encounters a state of stagnation, it is reset, and a sequence of iterations produces a new best particle. This particle is then perturbed to form a new extraction swarm.

The final level involves the merging of all sub-swarms from the previous level into a single large swarm, referred to as the optimization swarm. Each level within the proposed approach employs its unique update strategies. To ensure a more objective and optimal control of the PSO algorithm, the population distribution characteristics undergo changes not only based on the generation number but also on the evolutionary state [31]. Initially, in the early stages, particles may be dispersed across various regions, resulting in a wide-ranging population distribution. As the evolutionary process

Table 1- List of notations

symbols	Description
N, D	Population size and number of dimensions respectively
V_i^t , w	Velocity and inertia weight respectively
c_1, c_2, c_3	Individual(cognitive) learning rate, group((social) learning rates and coefficient vector respectively
U_1^t, U_2^t, U_3^t	Random values in the range [0, 1]
P_i^t, $P_{b_i}^t$, G_b^t, S_b^t	The current position, the locally best position, the best position of the current sub-swarm, and the best particles in the overall population, respectively
N_1, N_2, N_3	Maximum iteration of (Discovery, extraction and optimization) levels respectively
V_{min}, V_{max}	Minimum and maximum velocity [-4,4]
d_g, d_{max} , d_{min}	Distances of the (best, maximum and minimum) particles respectively
DF	Distribution factor
ub , lb	Lower and upper boundaries
dr	Gaussian distribution with $dr^i \in N(0, (0.1)^2)$
j	Cooperative coefficient between [0 1]
G, g	Maximum and current generation respectively
Fs, SP	Failed sub-swarms that have reached a local minimum and the total number of sub-swarms in the population respectively
f_{fs}, T_h	The first stage of fitness and the threshold of fitness respectively
p_t, C_{Pt} , f_{cpt}	Language statistics of plaintext, the statistics of decrypted text and Fitness of the candidate solution respectively
u, b	Uni-gram and bi-gram statistics
α, β	Weights assigning different priorities to uni-gram and bi-gram with $\alpha + \beta = 1$
L, n	The size of alphabet and decryption key respectively
AK, PD, ML	Actual Key, Polynomial Degree and maximum length of cipher respectively
$Bgen$, BTs, OS	Better generation, best time in seconds and Optimal solution respectively

unfolds, particles tend to cluster and converge towards local or global optima, leading to a different population distribution compared to the early stages.

To address this, the present work calculates the mean distance between each particle and all other particles, enabling an adaptive population distribution [32]. It is reasonable to anticipate that the mean distance from the globally best particle to other particles would be minimal in the final level, as the global best tends to be surrounded by the swarm. Conversely, in the first level, this mean distance would be maximal, since the global best is likely to be distant from the densely packed swarm.

Furthermore, in the second level, an evolution rate is calculated to gauge the progress of particles within each sub-swarm. This rate helps determine the appropriate timing for regrouping based on the evolutionary status of the sub-swarms [33]. Fig. 2 illustrates the flowchart of the proposed system.

3.2.1. Particles representation and initialization

Each particle within the population is characterized by two main vectors: position and velocity. The position vector consists of real numbers ranging from [-1,1] , representing the potential solutions for our proposed system. The position vector consists of real numbers ranging from [-1, 1], representing potential solutions for our proposed system. For the ith particle, its position is denoted as $P_i^t = \left(P_i^1, P_i^2, \cdots, P_i^D\right)$, where D refers to the dimensional search space encompassing all possible solutions, including the global optimum [34]. The second vector, $V_i^t = \left(V_i^1, V_i^2, \ldots, V_i^D\right)$, represents the velocity and is composed of real numbers ranging from -4 to 4. It has the same length as the position vector. Whenever a velocity value is computed, the following piece-wise function is applied to restrict them to the range [$V_{\min}, V_{\max}$]:

$$V_i^t = \begin{cases} V_{max}, \text{ if } v_{id}^t > V_{max} \\ V_i^t, \quad \text{if } V_i^t \le V_{max} \\ V_{min}, \text{ if } v_{id}^t < V_{min} \end{cases} \tag{4}$$

To form the population, each particle's position and velocity are assigned randomly within the specified ranges. In this research, we convert the position vector of each particle into a binary representation using the following equation:

$$P_i^t = \begin{cases} 1, \text{ if } P_i^t > 0 \\ 0, \text{ otherwise} \end{cases} \tag{5}$$

3.2.2. An adaptive population distribution

Compute the average distance between each particle and all other particles based on the Euclidean metric at the present position, as described in the equation below [32]:

$$d_i = \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} \sqrt{\sum_{k=1}^{D} \left(P_i^k - P_j^k\right)^2} \tag{6}$$

where N and D are the population size and the number of dimensions, respectively. Then compute

Received: August 14, 2023. Revised: October 7, 2023.

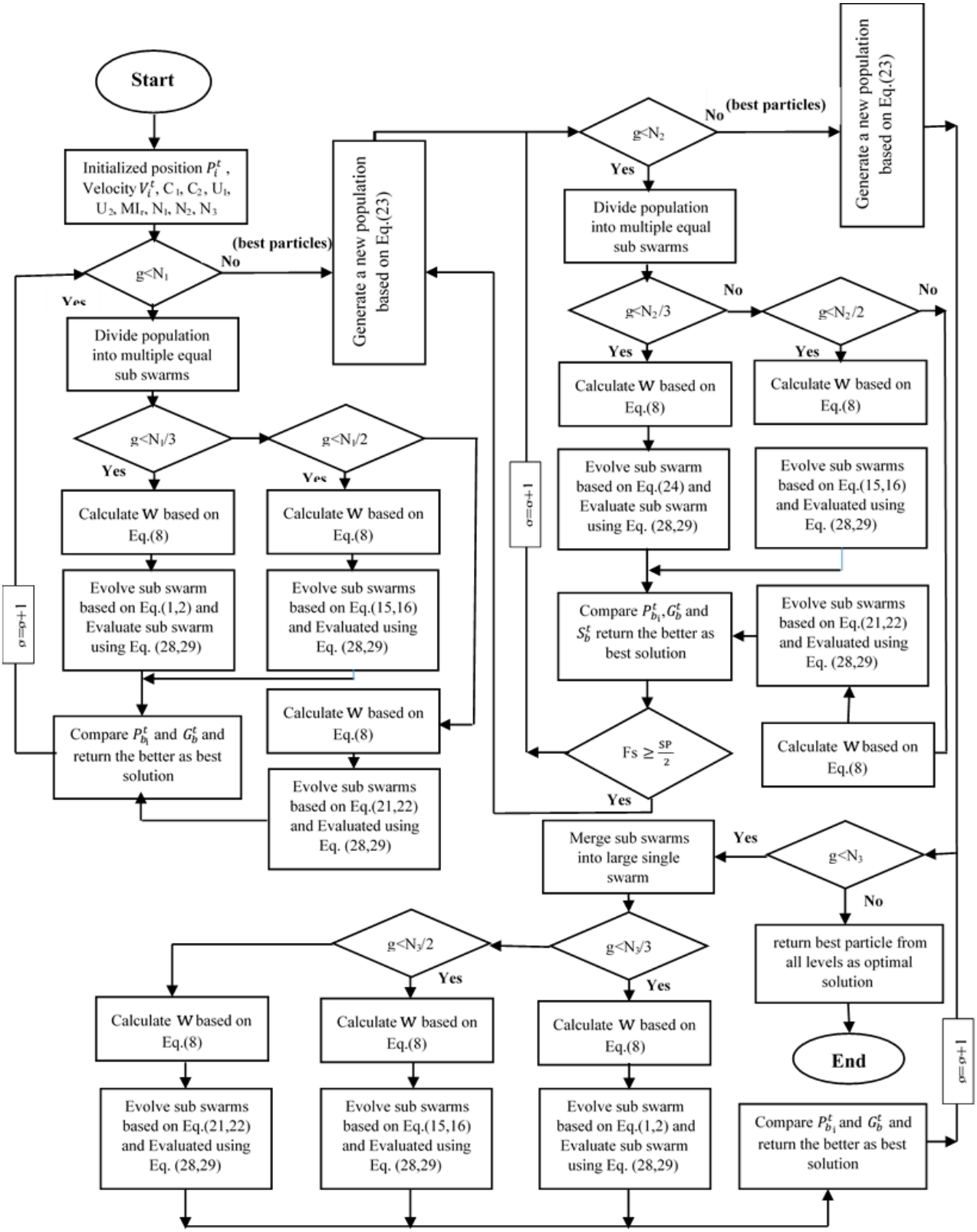


Figure. 2 Flowchart of proposed system

population distribution factor (DF) based on the following equation:

$$DF = \frac{d_g - d_{min}}{d_{max} - d_{min}} \in [0,1] \quad (7)$$

where d_g refers to distance of the best particle at current iteration, d_{min} and d_{max} refers to the minimum and maximum distances respectively.

In the proposed system, the distribution factor is substantial in the Discovery level, gradually

decreasing in the subsequent levels and eventually diminishing in the final level unless there are environmental changes.

To balance global and local search abilities in the PSO algorithm, the inertia weight (w) is used. Prior studies often suggested a large w value in the early generations, gradually decreasing over time [28, 35]. However, this approach may not guarantee optimal performance for all problem types. Therefore, the distribution factor (DF) shares similarities with the inertia weight, starting relatively large in the initial stages and gradually decreasing.

In this study, the inertia weight (w) is determined by the distribution factor (DF) based on the equation provided below [36].

$$w = \frac{1}{1+1.5e^{-2.6DF}} \in [0.4,0.9]\ \forall DF \in [0,1] \quad (8)$$

3.2.3. Discovery level

It's the first level of the proposed system, and the primary objective is to conduct a comprehensive exploration of the search space and generate smaller search regions for the next level (extraction). To achieve this, after constructing the initial population as described in the previous section, the population is partitioned into a collection of equally-sized predefined sub-swarms to cover multiple regions in dimensions, randomly distributed across the dimensional space. To determine the appropriate number of sub-swarms, this work assumes that the number of sub-swarms should be a factor of the population size [37, 38].

These sub-swarms may not necessarily be adjacent to one another, and each sub-swarm evolves independently without any communication to maximize the exploration of the search space during the evolving process. This level is divided into three equal predefined periods. During the first period, the standard PSO algorithm is employed to update velocity and position based on Eqs. (1) and (2). In the second period, two additional candidate velocities and positions are created, resulting in three candidate points for each velocity and position, as described in the equations below [9].

$$a^1 = w \cdot V_i^t \quad (9)$$

$$a^2 = c_1 \cdot U_1^t \cdot \left(P_{b_i}^t - P_i^t\right) \quad (10)$$

$$a^3 = c_2 \cdot U_2^t \cdot \left(\ G_b^t - P_{iD}^t\right) \quad (11)$$

$$v^1 = a^1 \quad (12)$$

$$v^2 = a^1 + a^2 \quad (13)$$

$$v^3 = a^1 + a^2 + a^3 \quad (14)$$

$$P_{iD}^J = P_i^t + v^J\ ,\ J = 1,2,3 \quad (15)$$

These three points specified in Eq. (15) are assessed based on Eq. (16) and the one with the highest fitness is chosen as the final update point. Simultaneously, its corresponding velocity, which contributed to the creation of the best point, is also selected as the initial velocity for the subsequent iteration.

$$Y\left(P_{iD}^{t+1}, v_i^{t+1}\right) = \begin{cases} \left(P_{iD}^1, v^1\right) if\ f\left(P_{iD}^1\right) \geq f\left(P_{iD}^2\right)\ and\ f\left(P_{iD}^1\right) \geq f\left(P_{iD}^3\right) \\ \left(P_{iD}^2, v^2\right) if\ f\left(P_{iD}^2\right) \geq f\left(P_{iD}^1\right)\ and\ f\left(P_{iD}^2\right) \geq f\left(P_{iD}^3\right) \\ \left(P_{iD}^3, v^3\right)\ otherwise \end{cases} \quad (16)$$

In the last period, four additional updating velocities can be obtained using Eqs. (17-21), in addition to the three previous velocities from Eqs. (9-14). This results in seven candidate points for each velocity and position, as described in the equations below:

$$v^4 = a^2 \quad (17)$$

$$v^5 = a^3 \quad (18)$$

$$v^6 = a^1 + a^3 \quad (19)$$

$$v^7 = a^2 + a^3 \quad (20)$$

$$P_{iD}^J = P_i^t + v^J, \quad J = 1,2,3 \ldots,7 \quad (21)$$

These seven points were evaluated based on Eq. (22), and the one with the highest fitness was selected as the final update point.

$$\left(P_{iD}^{t+1}, v_i^{t+1}\right) = \max_k f\left(P_{iD}^J\right), J = 1,2,\ldots,7 \quad (22)$$

In order to maintain a higher diversity and enhance the exploratory capabilities, the inertia weight (w) is designed to follow the distribution factor, resulting in a non-linear decrease from 0.9 to 0.6 as indicated by Eq. (8).

3.2.4. Extraction level

The initialization of the extraction swarm level depends on the previous level. It involves creating

new sub-swarms by selecting the best particle from each sub-swarm in the first level, with the aim of enhancing the likelihood of attaining the local optimum. This is achieved by perturbing the position of the best particle into a set of particles within a small neighbourhood, as described by the equation below [35].

$$P_i^t = P_i^t + dr \cdot (lb - ub) \quad (23)$$

where $dr^i = [dr^1, dr^2, \dots, dr^D]$, and dr follows the Gaussian distribution with $dr^i \in N(0, (0.1)^2)$, lb and ub are the lower and upper bound respectively.

Similar to the previous level, this level is divided into three equal periods. The same processes as in the previous level were applied, except for the calculation of Eq. (1), in which particles in different sub-swarms communicate completely in parallel and co-evolutionary manner with each other and dynamically change their positions based on Eq. (24). The $P_{b_i}^t$ of each particle was selected based on the historically best position, and G_b^t was selected from all particles in the sub-swarm. Then, the S_b^t, which represents the best particles overall the population, was selected through communication between sub-swarms.

$$P_{iD}^{t+1} = P_i^t + W \cdot V_i^t + c_1 U_1^t \left(P_{b_i}^t - P_i^t\right) + c_2 U_2^t \left(G_b^t - P_{iD}^t\right) + j\left(c_3 U_3^t \left(S_b^t - P_{iD}^t\right)\right) \quad (24)$$

Where j is the cooperative coefficient in [0, 1]. When $j = 0$, each sub-swarm iterates independently, and when $j = 1$, sub-swarms iterate with full information exchanged, calculated based on the following equation:

$$j = [\ (jmax - jmin) * \frac{g}{G} \quad (25)$$

To enhance convergence toward the real global solution and share good information through the evolutionary process, all particles are randomly regrouped based on a dynamic regrouping strategy. In this strategy, particles are randomly reassigned to each sub-swarm when these sub-swarms fall into local optima and fail to reach the optimal solutions. The equation below illustrates the appropriate timing for the regrouping process.

$$\boldsymbol{Fs} \geq \frac{\boldsymbol{SP}}{\boldsymbol{2}} \quad (26)$$

where Fs refers to the sub-swarms that failed to reach the optimal solutions and fell into local optima, and SP refers to the total number of sub-swarms in the population.

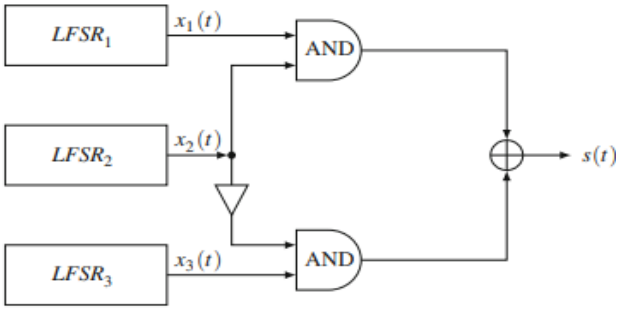


Figure. 3 Structure of Geffe generator

3.2.5. Optimization level

The final level of the proposed algorithm is initialized by merging all the sub-swarms into one large swarm. The best particle is selected from each sub-swarm to produce a new single, large population based on Eq. (23), and the same operations as in the first level are performed.

4. Details of nonlinear stream cipher systems

Stream ciphers are a fundamental category of ciphers designed to sequentially encrypt or decrypt individual symbols of the alphabet, typically represented as binary digits. The encryption process involves XOR-ing each bit or byte of the plaintext with a pseudo-random sequence of bits known as the keystream [39]. To generate a pseudorandom keystream with a long period, seed values are input into a generator. This keystream is then combined with the plaintext using an XOR operation. The Geffe generator is one of the numerous bit-oriented stream generators used to generate keystreams based on three maximal length LFSRs with lengths L_1, L_2, and L_3, which are pairwise relatively prime [40]. The combining function is:

$$f(x_1, x_2, x_3) = x_1 x_2 \oplus x_2 x_3 \oplus x_3 \quad (27)$$

In the Geffe generator, $LFSR_2$ serves as a selector, alternating the output between LFSR1 and LFSR3. The resulting keystream sequence, denoted as {s(t)}, exhibits a period of $T = (2^{L1} - 1)(2^{L2} - 1)(2^{L3} - 1)$, and the linear complexity of the keystream sequence is given by $L_C = L_1L_2 + L_2L_3 + L_3$. Fig. 3 presents the primary diagram of the Geffe generator, illustrating its overall structure and the interactions between $LFSR_1$, $LFSR_2$, and $LFSR_3$ [41].

5. Cryptanalysis nonlinear stream cipher based on proposed system

The cryptanalysis process of the proposed system is based on TL_MPSO, with the target being a nonlinear stream cipher. The traditional PSO algorithm often suffers from premature convergence, especially when applied to solve high-dimensional problems. To overcome this drawback and enhance the ability of PSO to tackle large search spaces, we propose a method to maintain the diversity of the population. In this section, we explain the proposed algorithm and how TL_MPSO can be employed to discover the keystream utilized in the Geffe generator. The primary objective is to effectively find the initial key values of the combined LFSRs. It's worth noting that all the approaches discussed in section 3 were designed to solve problems that required continuous optimization.

5.1 Particle representation and swarm initialization

Each particle represents a candidate solution, which is an (n-bit) binary key. Therefore, the proposed algorithm starts with an appropriate population size proportional to the length of the key to be attacked. The population is generated randomly, containing position and velocity for each particle. Applying the proposed algorithm to solve a stream cipher requires converting the position values into a binary representation. Eq. (5) converts a real vector of positions into a binary vector. The values of velocity are specified within lower and upper bounds to ensure that the elements do not exceed the specified range, as explained in section 3.1 and Eq. (4).

5.2 Objective function and evaluation process

The fitness function determines the quality of a solution, allowing for the comparison of solutions and the identification of the best representation. In this work, two stages of the objective function were utilized to evaluate the algorithm's elements (particles), each representing a candidate key. The first stage of the fitness function is applied in each iteration and is based on the correlation between the ciphertext and the output of each register.

During the first stage, a threshold function is employed to assess the candidate solution and determine whether it should proceed to the next stage. This threshold function depends on the count of zeros and ones. If the candidate solutions have zeros greater than or equal to 60%, they proceed to the next stage; otherwise, they remain in the same stage. This decision is influenced by the observation that in English language plaintext messages, the percentage of '0's typically accounts for approximately 60% of the entire text, although this value may vary depending on various plaintext sizes [16].

$$f_{fs} = No(z)/N$$
$$T_h = f_{fs} \geq 0.60 \tag{28}$$

In the second stage, the ciphertext is decrypted using the candidate keystreams that survived from the previous stage. Each solution is evaluated using frequency analysis, with the aim of comparing the frequencies in the decrypted text to those identified in the target literature. The difference between the most frequent bigram and unigram frequencies of the candidate plaintext and those of the corresponding bigram and unigram frequencies in normal language is used as the fitness function for this stage. The most commonly used equation for this purpose is given below [42]:

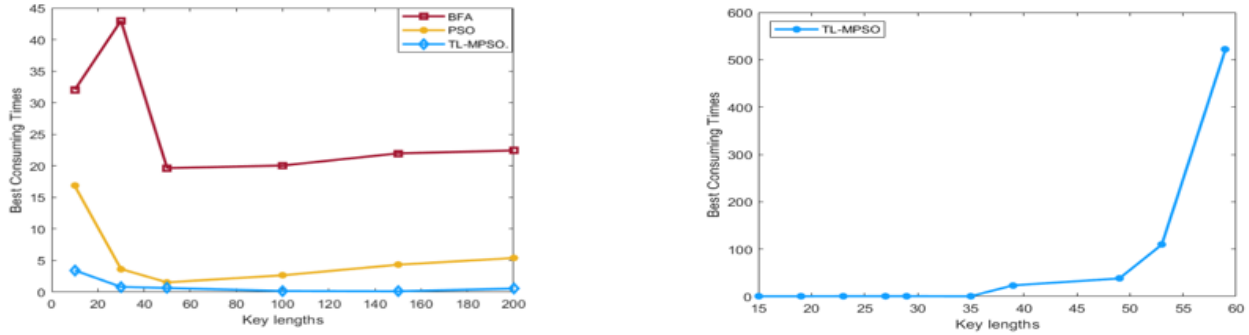
$$f_{cpt} = \alpha \sum_{1}^{L} |p_t(i) - C_{Pt}(i)|^u + \beta \sum_{1,1}^{n} |p_t(i,j) - C_{Pt}(i,j)|^b \tag{29}$$

where p_t, C_{Pt} respectively denote the known language statistics and the decrypted text statistics. u, b: refers respectively to uni-gram and bi-gram statistics, while α and β (with $\alpha + \beta = 1$) are weights assigning different priorities to uni-gram and bi-gram, n is the key used in decryption process and L is the size of the used alphabet. To evaluate particles, each candidate solution (particle) is applied to decrypt the current ciphertext and is evaluated based on Eqs. (15), (16), (21) and (22).

5.3Evolution process

The proposed algorithm enhances the diversity of the population and can conduct deep searches in the search space. In the first level of the proposed algorithm, the population is divided into predefined sub-swarms to cover multiple dimensions, and each particle evolves in its own tendency in the dimensional space according to two directions based on Eqs. (2), (15), (16), (21) and (22). The first direction is towards the best position of the current particle throughout iterations, which is called the local best position ($P_{b_1}^t$), while the second direction is towards the best particle's position within each sub-swarm, known as the global best position (G_b^t).

In the second level, the best particles of each sub-swarm (G_b^t) are used to create the population of the new sub-swarms based on Eq. (23). In the extraction



(a) (b)

Figure. 4 Attack Geffe generator: (a) The best consuming times of BFA, PSO and TL-MPSO to attack Geffe generator with keylength 15 and (b) Attacking Geffe cipher system based on 3 LFSRs with total lengths [15,19,23,27,29,35,39,49, 53,59} and 200 characters from ciphertext

Table 2. Applying proposed system, BFA and PSO to analysis Geffe cipher based on 3 LFSRs with total lengths {15 and 19} and ML = [10-200] with G=50 and N=30

PD	Methods		BFA		PSO			TL-MPSO		
	G	ML	AK	BTs	OS	BTs	Bgen	OS	BTs	Bgen
x^3+x^2+1 x^5+x^3+1 x^7+x^6+1	50	10	0.6346	32.03	0.6101	16.83	46	0.6346	3.44	19
		30	0.6231	42.98	0.6231	3.66	25	0.6231	0.83	11
		50	0.6061	19.64	0.6061	1.544	14	0.6061	0.65	9
		100	0.6062	20.05	0.6062	2.659	20	0.6062	0.18	7
		150	0.6069	21.97	0.6069	4.338	29	0.6069	0.136	4
		200	0.607	22.43	0.607	5.4	33	0.607	0.56	8
x^3+x^2+1 x^5+x^3+1 $x^{11}+x^9+1$	50	10	0.6346	21.47	0.6309	19.759	49	0.6346	3.6901	21
		30	0.6231	22.32	0.6214	8.932	36	0.6231	0.1382	2
		50	0.6061	23.73	0.6061	3.0467	27	0.6061	0.1581	3
		100	0.6062	27.63	0.6062	4.1789	30	0.6062	0.3162	7
		150	0.6069	31.54	0.6069	6.6652	35	0.6069	0.1273	2
		200	0.607	35.55	0.607	10.1791	42	0.607	0.6400	9

level, each particle evolves according to three directions, rather than two. The additional direction involves communication between all sub-swarms in the search space to update the learning sample, which is the best particle among all sub-swarms, referred to as the best swarm's position (S_b^t), based on Eq. (24).

In the last level, all sub-swarms are merged into a single large swarm, and each particle evolves in two directions, similar to the first level.

6. Experimental results

The aim of the experiments was to measure the performance of the TL-MPSO algorithm in finding the initial values of stream cipher systems with Geffe generators using ciphertext-only attacks and compare the results with traditional PSO and brute force attacks, as well as the results from previous works. All experiments were conducted on a Windows 11 laptop with an 11th Gen Intel(R) Core (TM) i5-11400H processor, 64-bit architecture, and 8 GB of RAM. The implementation of the cryptanalysis scheme was done in MATLAB. Two stopping criteria were used to halt the TL-MPSO cryptanalysis system: the first criterion was reaching the maximum number of generations, and the second was when the actual key value was found.

In this research, plaintext of varying sizes [10 to 400] was encrypted using a stream cipher based on the Geffe generator with three LFSRs of different lengths. The proposed algorithm employed a ciphertext-only attack to discover the encryption key within a reasonable timeframe. Firstly, Table 2 shows the results of using brute force attack, traditional PSO, and TL-MPSO to break ciphertext with lengths (10-200) using key lengths of (15 and 19). Secondly, the results of attacking systems with lengths (35, 39, and 53) using the proposed system are mentioned in Table 3. Fig. 4 presents the plotted curves of these results. Lastly, Table 4 illustrates the comparison between the proposed system and previous works. It's necessary to mention that all experiments were performed on text using the English alphabet, and spaces and all punctuation were removed.

Table 3. Applying proposed system to analysis Geffe cipher based on 3 LFSRs with total lengths (35,39 and 53) and ML = [10-200] with G={100,150,500} and N=70.

PD	ML	10	30	50	100	150	200
x^7+x^6+1 $x^{11}+x^9+1$ $x^{17}+x^{14}+1$	OS	0.601	0.6131	0.6061	0.6062	0.6069	0.607
	BTs	5.1719	0.600	0.8119	0.686	0.4340	0.583
x^9+x^5+1 $x^{11}+x^9+1$ $x^{19}+x^{18}+x^{17}+x^{14}+1$	OS	0.591	0.619	0.6061	0.6062	0.6069	0.607
	BTs	9.5	14.35	14.62	22.87	18.78	23.60
$x^{13}+x^{12}+x^{11}+x^8+1$ $x^{17}+x^{14}+1$ $x^{23}+x^{18}+1$	OS	0.577	0.605	0.592	0.602	0.600	0.6069
	BTs	65.91	62.39	83.45	100.85	96.65	109.59

Table 4. Results of attacking Geffe stream cipher by proposed system against previews works

GEFFE GENERATOR	PERIODS	RELATED WORK				PROPOSED SYSTEM		
		References	BTs	G	ML	BTs	G	ML
3,5,7	27,559	**[25]**	0.55	300	10	0.31	50	10
5,7,11	8,059,039	**[12]**	481	Na	400	0.18	50	20
7,9,11	132,844,159	**[20]**	1.6	16	400	0.328	50	30
		[16]	0.70	100	40			
7,9,13	531,571,327	**[12]**	877	Na	400	0.61	100	50
9,11,19	5.484131149×10^{11}		3693	Na	400	22.87	100	50
13,17,19	5.628758659×10^{14}	**[20]**	95.2	376	400	38.17	200	150
13,17,23	9.006029958×10^{15}		1168.8	4805	400	109.59	500	200
13,17,29	5.76385985×10^{17}		3925.2	25820	400	721.77	1000	200

Based on the results presented in Tables 2, 3, and 4, the proposed system demonstrated remarkable efficiency in recovering the initial key of the Geffe generator, which relies on 3 LFSRs with lengths of 15 and 19. The process required merely 10 characters and a matter of seconds. Even for key lengths of 27 and 35, the proposed system achieved impressive performance, needing only 30 and 50 characters, respectively, within optimal timeframes.

The TL-MPSO algorithm's achievement was noteworthy, successfully revealing the confidential initial values of the Geffe stream generated by the 3 LFSRs with a combined length of up to 59 characters. This accomplishment was realized within a reasonable timeframe, requiring only 200 characters from the encrypted text.

A comparison with prior research revealed the superiority of the TL-MPSO approach. For instance, Table 4 demonstrated that TL-MPSO outperformed earlier methodologies in pinpointing the correct initial value of the Geffe generator. This was achieved using fewer characters and within suitable time constraints. To illustrate, for a key length of 27 characters, TL-MPSO took only 0.32 seconds and 30 characters, whereas the previous methods [16] and [20] necessitated 0.7 and 1.6 seconds, along with 40 and 400 characters, respectively.

Furthermore, TL-MPSO exhibited efficiency in solving ciphers with key lengths of 53 and 59 characters, demanding roughly 200 characters and 109.5 seconds, and 721.7 seconds, respectively. In contrast, method [20] required 400 characters and 1168.8, 3925.2 seconds for the same tasks. These observations underscore the substantial efficiency of the proposed system, as indicated by its processing speed, fitness values, and the quantity of cipher characters required to successfully recover accurate secret keys.

7. Conclusions

A new technique for enhancement traditional PSO, called TL-MPSO has been proposed to cryptanalysis nonlinear stream cipher based on Geffe Generator. In TL-MPSO, the total generations

divided into three distinct levels called discovery, Extraction and optimization levels with a movement strategy for each particle is set up through the new inertia weight updating process. Where dynamic regrouping strategy was applied on proposed system to enhance converge toward real global solution and share good information through evolutionary process. To evaluate the efficiency and reliability of the TL-MPSO in solving nonlinear stream cipher, ciphertext only attack was used to cryptanalysis ciphertext encrypted using Geffe genereator based on three LFSRs with different polynomial degrees up to 29.

According to obtained results, TL-MPSO was able to solve the considered encrypted text in optimal time. Results from Table 2 illustrated that the proposed system able to find the correct initial key for different cipher length with average time less than one second. As per results shown in Table 3, the proposed algorithm successful to recover correct initial key for lengths up to 53 with only 200 characters from ciphertext. Where Table 4 showed that TL-MPSO algorithm able to solve Geffe generator with maximum period 9.006029958×10^{15} with about only two minutes with only 200 characters.

The comparison results from previews section proved that the TL-MPSO algorithm improves upon the traditional PSO algorithm and techniques of previews works where the proposed algorithm is simple but effective way that helps to prevent the swarm from becoming stuck in the local optima.

Based on the current research, there are several suggestions for future works. The initial recommendation involves the utilization of TL-MPSO to target alternative types of generators, for instance, the Bruer and Shrinking generators. This could provide valuable insights into the versatility and effectiveness of the TL-MPSO approach across different scenarios. The second suggestion work is extending TL-MPSO to cryptanalysis block cipher systems. Further study can be done by co-search between TL-MPSO and another optimization algorithm such as ACO. Lastly, implementation of TL-MPSO based on parallel computation to gain optimal time and efforts.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, Mohammed H. Ahmed, Asaad N. Hashim, and Khalid A. Hussein; methodology, Mohammed H. Ahmed; software, Mohammed H. Ahmed; validation, Mohammed H. Ahmed, Asaad N. Hashim, and Khalid A. Hussein; formal analysis, Mohammed H. Ahmed, Asaad N. Hashim, and Khalid A. Hussein; investigation, Asaad N. Hashim, and Khalid A. Hussein; resources, Mohammed H. Ahmed; data curation, Mohammed H. Ahmed and Khalid A. Hussein; writing—original draft preparation, Mohammed H. Ahmed; writing—review and editing, Mohammed H. Ahmed; visualization, Mohammed H. Ahmed; supervision, Asaad N. Hashim, and Khalid A. Hussein; project administration, Mohammed H. Ahmed; funding acquisition, Mohammed H. Ahmed, Asaad N. Hashim, and Khalid A. Hussein.

Acknowledgements

We thank Mustansiriyah University and the University of Kufa for their help with our research.

References

[1] S. Bhattacharya, “Cryptology and information security-past, present, and future role in society”, *International Journal on Cryptography and Information Security (IJCIS)*, Vol. 9, No. 1/2, pp. 13-39, June 2019.

[2] S. Sharma and Y. Gupta, “Study on cryptography and techniques”, *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, Vol. 2, No. 1, 2017.

[3] N. Munir, M. Khan, T. Shah, A. Alanazi, and I. Hussain, “Cryptanalysis of nonlinear confusion component based encryption algorithm”, *Integration*, Vol. 79, pp. 41-47, 2021.

[4] M. Ahmed, A. Shibeeb, and A. Mohammed, “Solve Polyalphabetic Cipher Based on Intelligent System”, In: *Proc. of International Conf. on Communication & Information Technology (ICICT)*, Basrah, Iraq, pp. 290-296, 2021.

[5] R. Mohammed, “Design a Lightweight Authentication Encryption Based on Stream Cipher and Chaotic Maps with Sponge Structure for Internet of Things Applications”, *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 1, pp. 532-547, 2023, doi: 10.22266/ijies2022.1031.08.

[6] M. Alenezi, H. Alabdulrazzaq, and N. Mohammad, “Symmetric Encryption Algorithms: Review and Evaluation Study”, *International Journal of Communication Networks and Information Security (IJCNIS)*, Vol. 12, No.2, pp. 256-272, 2020.

[7] R. R. Allah, H. Abdulkader, S. Elatif, W. Elkilani, E. A. Maghayreh, H. Dhahri, and A. Mahmood, “A Novel Binary Hybrid PSO-EO

Algorithm for Cryptanalysis of Internal State of RC4 Cipher", *Sensors*, Vol. 22, No. 10, p. 3844, 2022.

[8] M. Jain, V. Saihjpal, N. Singh and S. Singh, "An Overview of Variants and Advancements of PSO Algorithm", *Applied Sciences*, Vol. 12, No. 17, p. 8392, 2022.

[9] Q. Zhao and C. Li, "Two-Stage Multi-Swarm Particle Swarm Optimizer for Unconstrained and Constrained Global Optimization", *IEEE Access*, Vol. 8, pp. 124905-124927, 2020.

[10] Z. Zhan, J. Zhang, Y. Li and H. Chung, "Adaptive Particle Swarm Optimization", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 39, No. 6, pp. 1362-1381, 2009.

[11] H. A. Sharifi, "Cryptanalysis of Stream Cipher System Using Particle Swarm Optimization Algorithm", *Journal of Kerbala University*, Vol. 8, No.4, pp. 384-394, 2010.

[12] M. Din, A. Bhateja, and R. Ratan, "Cryptanalysis of Geffe Generator Using Genetic Algorithm", In: *Proc. of International Conf. on Soft Computing for Problem Solving: SocProS 2013*, Springer, New Delhi, Vol. 259, pp. 509-515, 2014.

[13] I. Polak and M. Boryczka, "Genetic Algorithm in Stream Cipher Cryptanalysis", In: *Proc. of International Conf. on Computational Collective Intelligence, ICCCI 2015*, Madrid, Spain, Vol. 9330, pp. 149-158, 2015.

[14] K. Pommerening, "Cryptanalysis of nonlinear feedback shift registers", C*ryptologia*, Vol. 40, No. 4, pp. 303-315, 2016.

[15] M. Din, S. Pal, S. Muttoo, and A. Jain, "Applying Cuckoo Search for analysis of LFSR based cryptosystem", *Perspectives in Science*, Vol. 8, pp. 435-439, 2016.

[16] S. A. Ageelee and R. Kadhum, "Cryptanalysis of nonlinear stream cipher cryptosystem based on improved particle swarm optimization", *International Journal of Applied Information Systems*, Vol. 19, No. 1, pp. 78-84, 2017.

[17] F. Ali, M. A. Safi, and A. Yousif, "Analyzing Cryptosystems by Using Artificial Intelligence", *ANJS*, No. 1, pp. 100-108, 2018.

[18] S. Sadkhan and B. Yaseen, "A DNA-Sticker Algorithm for Cryptanalysis LFSRs and NLFSRs Based Stream Cipher", In: *Proc. of International Conf. on Advanced Science and Engineering (ICOASE)*, Duhok, Iraq, pp. 301-305, 2018.

[19] M. Din, S. Pal, and S. Muttoo, "Analysis of RC4 Crypts Using PSO Based Swarm Technique", In: *Proc. of International Conf. on Harmony Search and Nature Inspired Optimization Algorithms: Theory and Applications, ICHSA-2018*, Springer Singapore, Vol. 741, pp. 1049-1056, 2019.

[20] M. Din, S. Pal, and S. Muttoo, "Applying PSO based technique for analysis of geffe generator cryptosystem", In: *Proc. of International Conf. on Harmony Search and Nature Inspired Optimization Algorithms: Theory and Applications, ICHSA 2018*, Springer Singapore, Vol. 741, pp. 741-749, 2019.

[21] I. Polak and M. Boryczka, "Tabu Cryptanalysis of VMPC Stream Cipher", *Tatra Mountains Mathematical Publications*, Vol. 73, No. 1, pp. 145-162, 2019.

[22] I. Polak and M. Boryczka. "Tabu Search in revealing the internal state of RC4+ cipher", *Applied Soft Computing*, Vol. 77, pp. 509-519, 2019.

[23] R. Jawad and F. Ali, "Using Evolving Algorithms to Cryptanalysis Nonlinear Cryptosystems", *Baghdad Science Journal*, Vol. 17, No. 2, 2020.

[24] G. Mishra, I. Gupta, S. Murthy, and S. Pal, "Deep Learning based Cryptanalysis of Stream Ciphers", *Defence Science Journal*, Vol. 71, No. 4, pp. 499-506, 2021.

[25] R. Jawad, "Proposed Hybrid Technique in Cryptanalysis of Cryptosystem Based on PSO and SA", *Iraqi Journal of Science*, Vol. 63, No. 10, pp. 4547-4558, 2022.

[26] M. Jain, V. Saihjpal, N. Singh and S. B. Singh, "An overview of variants and advancements of PSO algorithm", *Applied Sciences*, Vol. 12, No. 17, p. 8392, 2022.

[27] S. Lee, C. Cheng, C. Lin, and Y. Huang, "PSO-Based Target Localization and Tracking in Wireless Sensor Networks", *Electronics*, Vol. 12, No. 4, p. 905, 2023.

[28] H. Minh, S. Khatir, R. Rao, M. A. Wahab, and T. C. Le, "A variable velocity strategy particle swarm optimization algorithm (VVS-PSO) for damage assessment in structures", *Engineering with Computers*, Vol. 39, No. 2, pp. 1055-1084, 2023.

[29] Z. Chen, X. Li, Z. Zhu, Z. Zhao, L. Wang, D. Jiang, and Y. Rong, "The optimization of accuracy and efficiency for multistage precision grinding process with an improved particle swarm optimization algorithm", *International Journal of Advanced Robotic Systems*, Vol. 17, No. 1, 2020.

[30] H. Dai, D. Chen, and Z. Zheng, "Effects of Random Values for Particle Swarm Optimization Algorithm", *Algorithms*, Vol. 11,

No. 2, p. 23, 2018.
[31] D. Li, W. Guo, A. Lerch, Y. Li, L. Wang, and Q. Wu, "An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization", *Swarm and Evolutionary Computation*, Vol. 60, 2021.
[32] M. Zain, J. Kanesan, J. Chuah, S. Dhanapal, and G. Kendall, "A multi-objective particle swarm optimization algorithm based on dynamic boundary search for constrained optimization", *Applied Soft Computing*, Vol. 70, pp. 680-700, 2018.
[33] C. Qiu, "A Multi-swarm Particle Swarm Optimization with an Adaptive Regrouping Strategy for Feature Selection", In: *Proc. of International Conf. on Robotics and Automation Sciences (ICRAS)*, Wuhan, China, pp. 130-136, 2020.
[34] X. Xia, L. Gui, and Z. Zhan, "A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting", *Applied Soft Computing*, Vol. 67, pp. 126-140, 2018.
[35] A. Gad, "Particle swarm optimization algorithm and its applications: a systematic review", *Archives of Computational Methods in Engineering*, Vol. 29, No. 5, pp. 2531-2561, 2022.
[36] H. Zhou, H. Zhan, Z. Yang, and X. Wei, "AMPSO: Artificial Multi-Swarm Particle Swarm Optimization", *arXiv:2004. 07561*, 2020.
[37] X. Xia, L. Gui, and Z. Zhan, "A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting", *Applied Soft Computing*, Vol. 67, pp. 126-140, 2018.
[38] F. Ozsoydan and A. Baykasoğlu, "Quantum firefly swarms for multimodal dynamic optimization problems", *Expert Systems with Applications*, Vol. 115, pp. 189-199, 2019.
[39] F. Dridi, S. E. Assad, W. Youssef, M. Machhout, and R. Lozi, "The design and FPGA-based implementation of a stream cipher based on a secure chaotic generator", *Applied Sciences*, Vol. 11, No. 2, p. 625, 2021.
[40] B. Hameedi, A. Hattab, and M. Laftah, "A Pseudo-Random Number Generator Based on New Hybrid LFSR and LCG Algorithm", *Iraqi Journal of Science*, Vol. 63, No. 5, pp. 2230-2242, 2022.
[41] Y. Feng, H. Wang, C. Chang, H. Lu, F. Yang, and C. Wang, "A Novel Nonlinear Pseudorandom Sequence Generator for the Fractal Function", *Fractal and Fractional*, Vol. 6, No. 10, p. 589, 2022.
[42] T. Mekhaznia and A. Zidani, "Swarm intelligence algorithms in cryptanalysis of simple Feistel ciphers", *International Journal of Information and Communication Technology*, Vol.13, No. 1, pp. 114-138, 2018.