Journal of Information Hiding and Multimedia Signal Processing ©2026 ISSN 2073-4212
Ubiquitous International Volume **17**, Number **1**, March **2026**

Designing a New Text Encryption Approach Based on Genetic Algorithm

Riyadh Bassil Abduljabbar[1], Ahmad O. Almashhadani[1,*] and Hayder S. Radeaf[1]

[1]Department of Computer Engineering
University of Baghdad
Al-Jadriya, Baghdad, Iraq
riyadh.b@coeng.uobaghdad.edu.iq; ahmad.obaid@uobaghdad.edu.iq; haydersaadi@coeng.uobaghdad.edu.iq

*Corresponding author: Ahmad O. Almashhadani

Received November 14, 2025, revised February 7, 2026, accepted February 10, 2026.

Abstract. Today, data security is a major problem concerning organizations and individuals. The confidentiality of information is associated with using reliable and robust encryption algorithms in systems. Cyber-attacks on data and systems have become prevalent and sophisticated, and are increasing rapidly; hence, the need for developing robust encryption algorithms is crucial nowadays. This paper proposes a new encryption algorithm using dynamic symmetric key cryptography to encrypt text files. It utilizes a secret key for encryption and decryption processes, where the key's length is varied depending on the text size. This presented approach gives a trade-off between speed and security, making it suitable for various applications, such as secure messages and files. It utilizes the Genetic Algorithm (GA) only for creating a robust key, which reduces the complexity and computation overheads. The robustness and effectiveness of the proposed encryption algorithm were evaluated thoroughly by implementing several analysis tests, such as entropy, correlation, and the avalanche effect. The results of these tests demonstrate high randomness and uniformity in the created ciphertext and provide an important contribution to the enhancement of the cryptography field for protecting data.

Keywords: Genetic Algorithms, Text Encryption, Encryption Algorithm, Cryptography

1. **Introduction.** Today, the increased dependance on the Internet and communication systems has increased the risk of information disclosure. Cryptography is one of the significant aspects of communication security and becomes a major part for any computer and information security system [1]. Several security objectives, such as confidentiality, authentication, ,integrity, and digital signature are provided by cryptography [2]. with the exponentially increasing of data breaches and cyber threats, the need for strong and reliable encryption becomes a critical issue for organizations, bank systems, and individuals in securing their data. Symmetric key encryption is used efficiently to protect sensitive information from being disclosed by unauthorized persons. So, several researchers are developing encryption algorithms to enhance the data security [3]. Today, as data from several sectors, such as financial, healthcare, and industry, transition to digital format, it becomes more vulnerable to cyber threats. Symmetric encryption is one of the strategies that present robust protection against these growing cyber threats [4].

Symmetric encryption, also known as single key employed for encrypting and decrypting data [5] or conventional encryption, is used for encrypting data, in which the sender and receiver share one secret key to maintain the confidentiality of the message. Figure 1 shows the main parts in symmetric encryption scheme.

There are two requirements to apply symmetric encryption securely [6, 7]:

- A strong encryption algorithm such that an opponent who knows the algorithm and has access to one or more ciphertexts would be unable to decrypt ciphertext or discover the key.

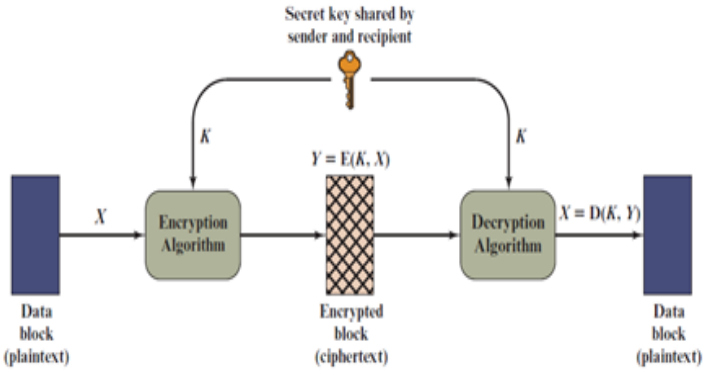

FIGURE 1. The symmetric encryption model [6].

- Distribute the secret key to the sender and receiver in a secure approach to maintain the confidentiality of the communications.

The Genetic Algorithms (GAs) are optimization methods based on the concept of genetics and natural selection. Several complex problems can be solved using GAs, such as optimization, encryption, image processing, and machine learning parameter tuning. Utilizing GA can reduce the overhead of time complexity [8], and it can also be applied to text and image data types. There are three most significant operators in GA: selection, crossover, and mutation [9]. Numerous problems are solved through modeling simplified genetic processes, and GAs have been applied effectively to create strong encryption keys and enhance encryption algorithms [10].

This research work presents a new encryption algorithm utilizing a dynamic symmetric key to secure texts. It uses one secret key for encryption and decryption processes, and the key's length depends on the length of the text. Since it provides a trade-off between speed and security, it is appropriate for various applications ranging from secure messages to file protection. It is based on the GA only for creating a robust key, which reduces the complexity and computation overhead. Contributing to developing resilient security solutions can be realized through using GA.

2. **The Genetic Algorithms (GAs) method.** Genetic Algorithms (GAs) are a type of search and optimization methodology based on the notion of natural selection and genetics. GAs are employed to solve complicated optimization problems by imitating the process of natural evolution. They operate on a population of potential solutions, which develop over successive generations through executing operators such as crossover, mutation, and selection. Figure 2 illustrates the GA genetic cycle [11].

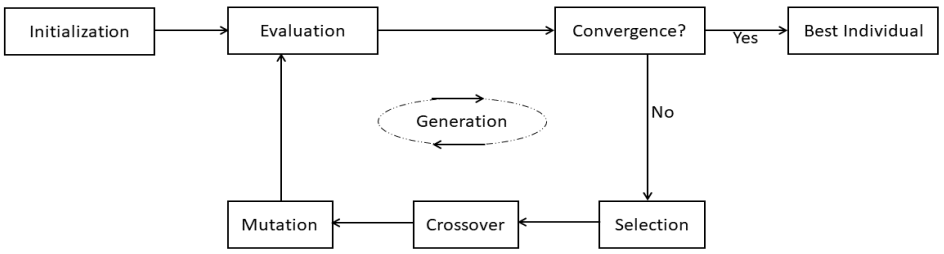

FIGURE 2. The sequence of operators and evaluation in GA.

GAs are heuristic algorithms utilized in many engineering applications, such as encryption techniques and optimization. The applications of GAs in encryption focus primarily on optimizing key generation and the design of encryption algorithms [12].

As classical encryption methods can be vulnerable to various attacks, GAs enhance security by generating robust keys using evolving solutions. These keys will be employed for securing the communications between the sender and the receiver. In addition to the key generation, GAs can be used to improve the traditional encryption algorithms by evolving the encryption schemes to develop their resistance to encryption attacks while maintaining efficiency [13].

3. **Related Work.** Various methods for data encryption using the genetic algorithm were proposed by Several researchers, as follows. Gaata et al [14] presented an approach to enhancing the security of big data during transmission by integrating a Modified Reverse Encryption Algorithm (MREA) with the genetic algorithm method. Two phases are required for this method: the first is the dynamic key generation using the genetic algorithm, and the second phase is the data encryption, which is performed by employing MREA. The data encryption robustness was enhanced by the combination of a genetic algorithm and MREA, which made it more challenging for unauthorized decryption processes. Since the proposed method provided low efficiency, it might be unsuitable for some applications.

Marjuni et al. [15] presented a method to secure the data concealment within digital images. They integrate three known approaches: the Advanced Encryption Standard (AES), GAs, and Least Significant Bit (LSB). The experiments that were implemented using standard images illustrated low efficiency for the method through getting low mean squared error (MSE) values and high peak signal-to-noise ratios (PSNR). However, using a genetic algorithm and LSB may increase the complexity and produce low performance. Zengin et al. [16] developed a new genetic encryption algorithm (GEA) inspired by DNA nature to enhance data security. The performance of GEA was compared with AES and the RSA algorithms. Introducing a robust encryption key using GEA might enhance the security features.

Sakr et al. [17] presented an encryption model that combines the properties of amino acids with genetic algorithms. Two encryption keys were generated to enhance the strength of the encryption. A robust encryption key was created using the genetic algorithm (GA). The second key is called the protein key, derived from converting the DNA sequence to a corresponding protein message. Although using amino acid sequences introduces new biology-inspired cryptographic methods, they increase the complexity of the encryption process. The experiments indicate the robustness of the method against some known attacks, such as ciphertext-only and brute-force attacks. However, the integration of biological concepts into encryption approaches may increase the implementation complexity and the computational overhead. In [18], Menon et al. combined a chaotic system with genetic algorithms to improve the text encryption process. The chaotic function is developed to encrypt input data by generating a pseudo-random sequence, whereas the genetic algorithm was employed to adjust the parameters of the chaotic map, thus enhancing the strength of the encryption data. Although this new method offers a large key space, i.e., making it more challenging to brute-force attacks, it may increase the complexity of computations. Moreover, a highly sensitive to the parameters of the chaotic map, which makes the method requires accurate tuning to get satisfying results.

Baagyere et al. [19] presented a multi-layer approach by combining the steganographic and cryptographic techniques using genetic algorithms, besides some properties of the Residue Number System (RNS), with a suitable fusing method. Firstly, the plaintext is encrypted using the genetic algorithm and RNS; then the ciphertext is embedded within images. The results of the simulation show that the proposed method can be robust to various forms of attacks. However, using a multi-layered scheme complicates the practical implementation and makes the performance sensitive to the selection of parameters. Akwukwuma et al. [20] designed and built a GUI (graphical user interface) application using the Python language to simplify the processes of text encryption and decryption. The AES algorithm is utilized to encrypt the plaintext with a block size of 128 bits. Pandey and Sharma [21] presented a novel encryption algorithm for images by combining three approaches: hybrid chaotic maps, elliptic curve cryptography (ECC), and genetic algorithms.

The proposed algorithm requires three phases: confusion-diffusion, encryption, and optimization. Combining the chaotic maps with ECC improves the security of the image encryption, making it stronger to brute force and statistical attacks. The genetic algorithm has been employed to optimize the image encryption parameters involved in the chaotic maps and ECC, or to improve the quality of the encrypted image. However, the performance can be sensitive to the parameters selected for the system, and the integration of hybrid algorithms, such as chaotic maps and ECC, increases the complexity of implementation.

4. **Methodology.** Securing sensitive information from unauthorized access has become an important task nowadays. One of the main effective techniques used to protect confidential data is encryption. The proposed method aims to enhance data security while maintaining high efficiency and low complexity. The GA is used to develop the performance for text encryption through the automated generation and selection process of optimal keys. The implementation of text encryption based on a genetic algorithm (GA) is illustrated in this section. The proposed algorithm starts by reading the text from a file and then converting the characters to binary format to prepare the data for the algorithm. Then, the steps of initializing the necessary parameters of GA are implemented.

Through using a fitness function to evaluate the generated keys based on their ability to conceal the plaintext, the GA effectively narrows down the search for the best key. The final (i.e., optimal) key is determined by its performance in terms of minimal correlation with the plaintext. Thereafter, this optimal key is used to perform the encryption process with a secure XOR. Besides, the decryption process that reverses the operation using the same key is included in the methodology, thus ensuring that the original plaintext can be accurately recovered. The encryption and decryption of the methodology demonstrate the algorithm's applicability in practical life, where data security is crucial. Figure 3 below illustrates the flow chart of the proposed method.

5. **Implementation.** The implementation of the presented method is indicated in this section. The implementation is conducted using MATLAB (R2024a), utilizing a structured approach to ensure the clarity of the process. The proposed algorithm consists of several significant stages:

- Reading the plaintext from a file
- Initializing GA parameters
- Executing the GA
- and encrypting and decrypting the text.

The following subsections outline the procedures in these stages.

5.1. **Reading the text from a file.** The algorithm begins by reading the text from a designated input file (such as Plain.txt). The plaintext is retrieved as character data and converted into an unsigned 8-bit integer format. This conversion is essential for subsequent binary transformations, which allow for effective encryption operations. Then, the characters are represented in binary format, simplifying the XOR operation used in the encryption process.

5.2. **Initialization the GA Parameters.** The keys' parameters are important for the GA optimization process. They are initialized accurately to make the GA operate effectively. These parameters are:

- Population Size: Population Size: This parameter determines the number of encryption keys (i.e., individuals) evaluated in each generation. It is determined by 100.
- Number of Generations: It specifies the number of iterations the GA executes. It is set to 50.
- Mutation Rate: It refers to the likelihood of introducing random changes to the keys to maintain the diversity in the population and explore new solution space. Set to 0.01.
- Key Length: The encryption key length is related to the length of the plaintext. The key must fit the length of the data being encrypted.

5.3. **Run GA for key optimization.** The main part of the implementation is running the GA over a predetermined number of generations. The steps executed in each generation are demonstrated below.

5.3.1. *Fitness Evaluation.* It is an important step that computes the performance efficiency of each solution (i.e., encryption key) in hiding the plaintext. The effectiveness of the key depends on its ability to minimize the correlation with the original plaintext. The minimum absolute correlation with the plaintext is determined by the fitness score [11]. Hence, each key in the population is evaluated based on its fitness score. The fitness function can be specified as:

P= [p_1, $p_2 \ldots$, p_n]: the plaintext (as a numeric vector)

K_i= [k_{i1}, $k_{i2} \ldots$, k_{in}]: the i-th candidate key in the population.

Corr (P, K_i): the Pearson correlation coefficient between plaintext and candidate key Then the fitness function f_i for the i-th candidate is:

$f_i = |Corr(P, K_i)|$

$$\mathrm{Corr}(P,K) = \frac{\sum_{j=1}^{n}(p_j - \bar{p})(k_{ij} - \bar{k}_i)}{\sqrt{\sum_{j=1}^{n}(p_j - \bar{p})^2}\sqrt{\sum_{j=1}^{n}(k_{ij} - \bar{k}_i)^2}} \tag{1}$$

$\bar{p}$: mean of plaintext values

$\bar{k}_i$: mean of the i-th candidate key

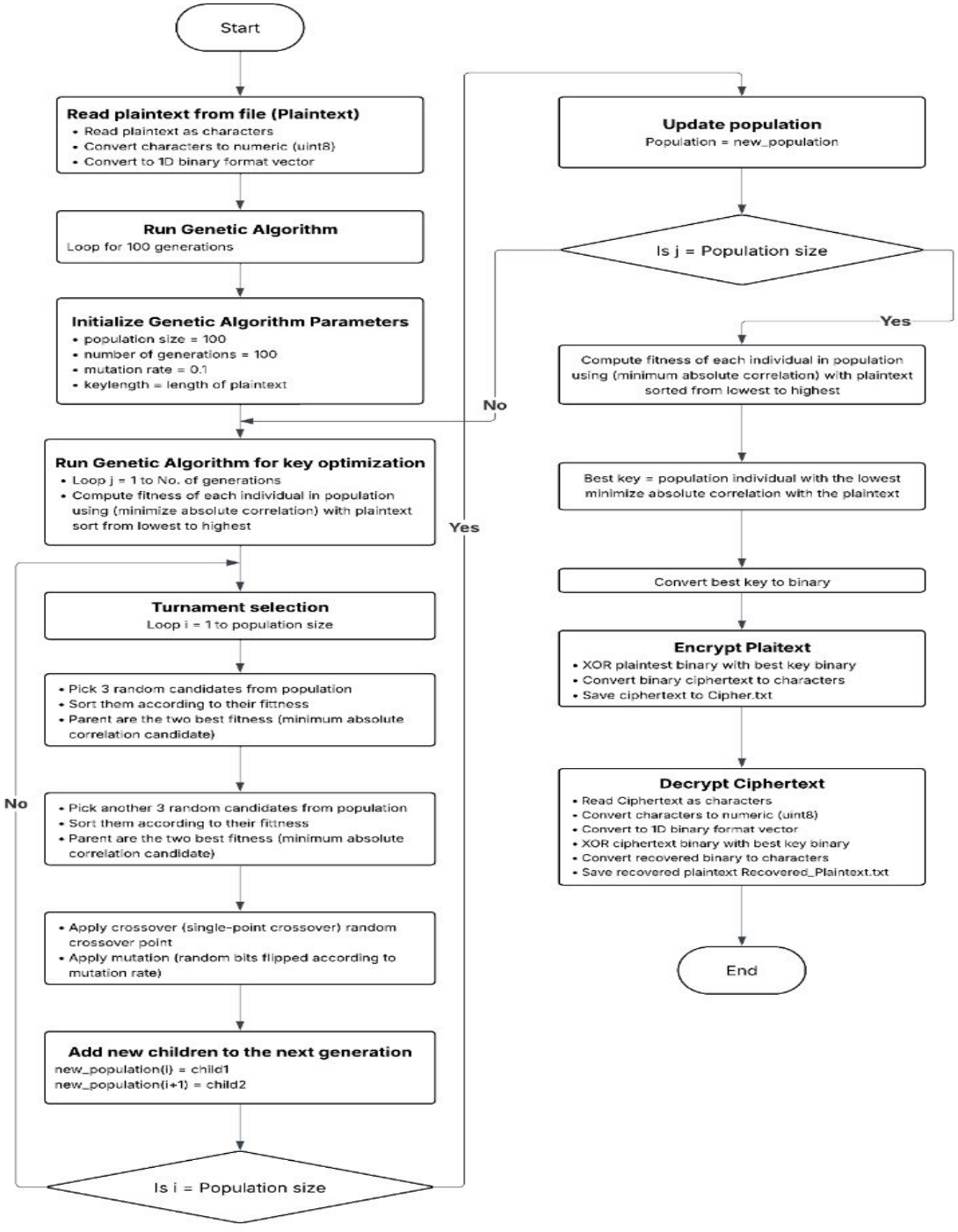

FIGURE 3. The flow chart of the encryption and decryption processes for the proposed algorithm.

5.3.2. *Selection.* This step determines the key from the current population that will be used to create the next generation. Choosing the right selection method is essential to balance the exploration of new solutions with the exploitation of the known best. We adopted the tournament method [11] to run the selection process because it is less susceptible to premature convergence and can easily adjust the selection pressure by changing the tournament size. Where a group of keys is randomly chosen, and then the one with the highest fitness is selected to create the next generation. The formula of the tournament approach is illustrated below:

If T represents the tournament size, and N is the number of tournaments required to select individuals, the process can be outlined as follows: For each tournament $i(where\, i = 1, 2, \ldots, N)$:

- Randomly select T individuals from the population.
- Find the individual ($winner_i$) with the highest fitness.

$$\text{winner}_i = \max_{j \in \text{Tournament}} \text{Fitness}(j) \tag{2}$$

Then collect all winners to form the new generation.

5.3.3. *Crossover.* Crossover is a genetic operator used to combine the genetic information of two parents to construct one chromosome (offspring). It is critical for producing diversity in the population and exploring the search for the best solutions. We applied the single-point crossover type to the selected keys to create offspring. This process develops the diversity by combining the properties of parent keys. Let p_1 and p_2 be the two parents represented in binary format, and assume c is the randomly selected crossover point. The offspring (o_1 and o_2) can be expressed as follows:

$$o_1 = p_1[0 : c] + p_2[c : \text{end}] \tag{3}$$

$$o_1 = p_2[0 : c] + p_1[c : \text{end}] \tag{4}$$

5.3.4. *Mutation.* Mutation is a random change in the chromosome (offspring) to get a new solution. It is used to maintain the diversity in the population and is usually applied with a low probability. The algorithm can explore the solution space and avoid convergence by applying different mutation strategies. The choice of mutation type significantly impacts the performance of the GA. We applied the bit flip mutation type that can be used for binary solutions where one or more random bits are selected and flipped. A bit is flipped from 0 to 1 or from 1 to 0 in a chromosome. If C represents a chromosome of length n binary, hence for each bit $C\,[i]$:

$$C[i] = \begin{cases} 1 - C[i] & \text{with probability } p \\ C[i] & \text{with probability } 1 - p \end{cases} \tag{5}$$

Where p is the mutation probability.

5.3.5. *Population update.* After selection, crossover, and mutation processes, the population update process occurs to ensure that the optimal individuals are retained while introducing a new generation. It is an important step where the current population is replaced or modified to form a new generation. Applying the update makes the GA explore a new solution space and enhance the fitness of the population by involving the best individuals. The strategy used in this paper to perform the update process is replacing the entire population with a new population. The steps of updating can be summarized as follows:

- Selecting parents from the current population.
- Performing the crossover and mutation processes to create offspring.
- Replacing the previous population with the new offspring.

5.3.6. *Select the best key.* After completing the defined number of generations, we identify and retain the key that has the minimum correlation to the plaintext based on the fitness values, as this key represents the best encryption key. Figure 4 shows the flow chart of the best key selection.

5.4. **Encrypting and decrypting the text.** The encryption process is carried out using the XOR (exclusive OR) operator between the binary representation of the plaintext and the selected optimal key. The encryption operation is summarized as indicated in the steps below:

- Convert the plaintext into a binary representation like ASCII code.
- Convert the selected best key into a binary format.
- Run the XOR logic between each bit of the plaintext and the corresponding bit of the best key.
- The producing binary ciphertext is transformed back into character format and saved to an output file named Cipher.txt.

The decryption process is used to convert the ciphertext into plaintext. Where the ciphertext is XORed with the same key used for the encryption process to recover the original plaintext, thereafter, the recovered plaintext is transformed back into character format and saved to a file named Recovered_Plaintext.txt.

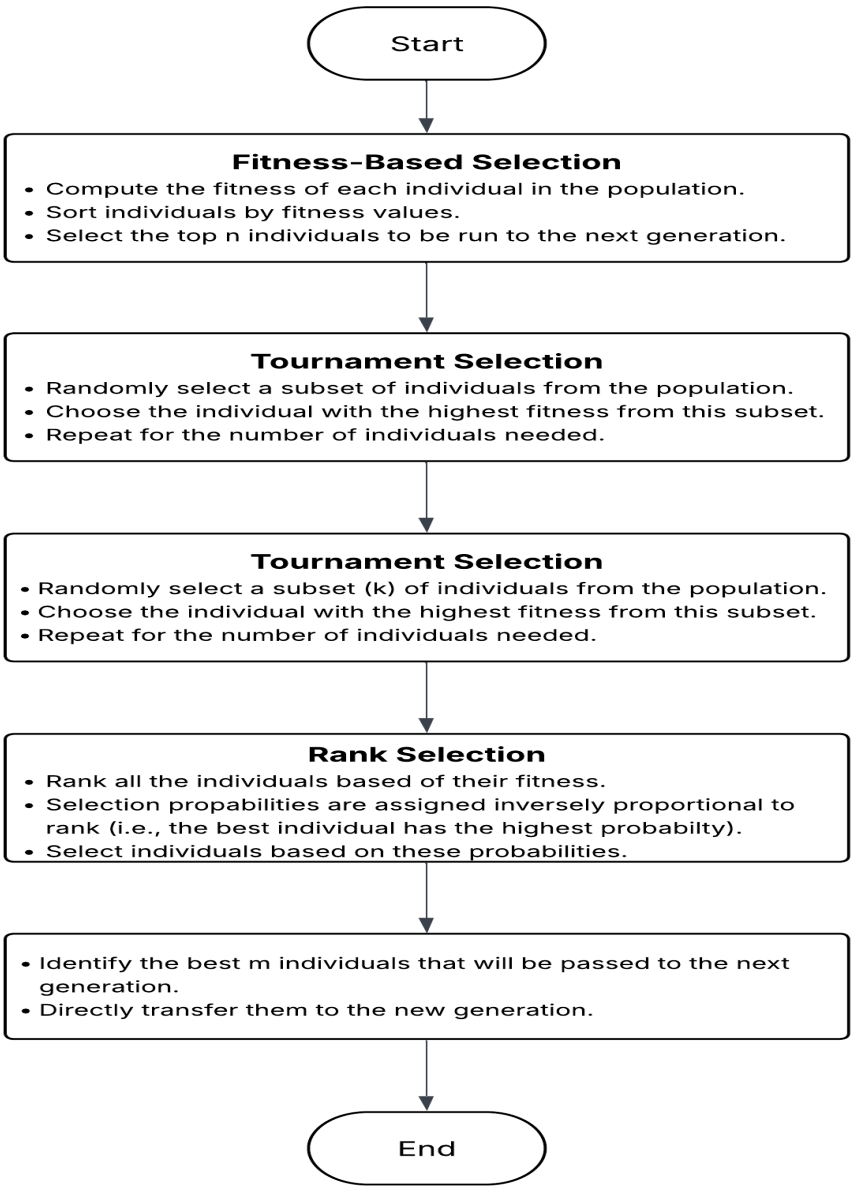

FIGURE 4. The flow chart of the best key selection

6. **Results and Discussion.** We will present and discuss the results of our experiments on encryption focusing on the text encryption. Several tests were implemented to show the effectiveness and the reliability of our method. Below the details and the results of these tests.

6.1. **Entropy Test.** The Shannon entropy is used to measure the randomness or unpredictability of the encrypted text, which is estimated based on the number of occurrences of zeros and ones inside the encrypted text. Shannon entropy is used to evaluate the encryption algorithm's strength by quantifying the ciphertext's randomness. The entropy is calculated as shown below [22].

$$H = -\sum_{i=1}^{N} (P_i \cdot \log_2(P_i)) \tag{6}$$

Where H is the entropy in bits per character of the encrypted text, and N: the total count of unique characters included in the encrypted text. P_i: represents the probability of the i_{th} character appearing in the encrypted text. A higher level of entropy signifies a greater degree of unpredictability and reflects

increased unpredictability and potentially improved security [23]. Table 1 shows our results of entropy calculations applied to some files with different sizes.

TABLE 1. The entropy results for some different encrypted files

Cypher text size	**Average entropy value**
1KB	6.5
3KB	6.54
5KB	6.54
10KB	6.56
100KB	6.56

The results show average entropy values of approximately 6.54 for the text files, indicating high levels of randomness in the ciphertexts.

6.2. **Correlation test.** The correlation test is a statistical method used to compute the randomness of encrypted text by assessing if there are any predictable patterns or linear relationships between the plain text and the ciphertext. A strong encryption algorithm should generate ciphertext that appears random, with no significant correlations between its components. The range of the correlation spans from -1 to 1. The correlation coefficient is calculated as illustrated below [24].

$$\text{Correlation coefficient} = \frac{\text{covariance}(P, C)}{\sigma(P) \cdot \sigma(C)} \tag{7}$$

Where:
$\sigma(P)$ is the standard deviation of plain text values.
$\sigma(C)$ is the standard deviation of encrypted text values.
Covariance: The covariance between encrypted text and plain text.

$$\text{Covariance}(P, C) = -\sum_{i=1}^{N} (C_i - \text{mean}(C)) \cdot (P_i - \text{mean}(P)) \tag{8}$$

TABLE 2. The correlation coefficient values for some random files

Size of plain text file	**Size of encrypted text file**	**Correlation coefficient**
1KB	1KB	0.0023
3KB	3KB	0.00094
5KB	5KB	-0.0196
10KB	10KB	0.0108
100KB	100KB	0.0032
250KB	250KB	0.0023

The results shown in Table 2 demonstrate that the values of the correlation coefficients are low (approximate zero) for different text file sizes, suggesting a good level of randomness (i.e., lack of association between the plain and encrypted text files). This implies that despite having knowledge of the ASCII values assigned to the characters in one of the files, it is difficult to infer the values of the characters in the other files.

6.3. **Avalanche effect.** The avalanche effect is an important property of encryption algorithms. It ensures that a small change in the input, such as a bit change in the plaintext, results in a significant and unpredictable change in the ciphertext (output). The avalanche effect can be evaluated by calculating the Hamming distance (number of differing bits) between the encrypted text and the modified encrypted text after changing one character in the plain text The percentage of different bits can be calculated relative to the total number of bits in the plaintext. The percentage of different bits can be calculated relative to the total number of bits in the plaintext [25].

$$\text{Avalanche effect} = \left(\frac{\text{Hamming Distance}}{\text{length}(P)}\right) \times 100 \tag{9}$$

Where:
P: is the total number of bits in the ciphertext.

$$\text{Hamming Distance} = \sum_{i=1}^{k} \text{XOR}(E_1[i], E_2[i]) \tag{10}$$

k: is the length of the encrypted text.
$E_1[i]\, and\, E_2[i]$ are the i_{th} bits of the encrypted and modified encrypted texts.

TABLE 3. The avalanche effect analysis

Plain text file with a single bit flip	**Cypher text (size)**	**Percentage avalanche effect**
1KB	1KB	50.3336%
3KB	3KB	50.5737%
5KB	5KB	50.5986%
10KB	10KB	50.6686%
100KB	100KB	50.7736%

The values of the avalanche effect in Table 3 are more than 50%, demonstrating that our method is robust in obscuring the relationship between the plaintext and ciphertext.

6.4. **Frequency test.** The frequency test is a technique used to analyze the frequency of letters or sets of letters in both plain and encrypted texts. It is an effective method against substitution ciphers, where each letter in the plaintext is replaced by another letter [26]. Figure 5 shows the frequency distribution for the letters (A to Z) in the plain text for a file size 250 KB where the number range in the horizontal axis ranges from 1 to 26, as illustrated below.

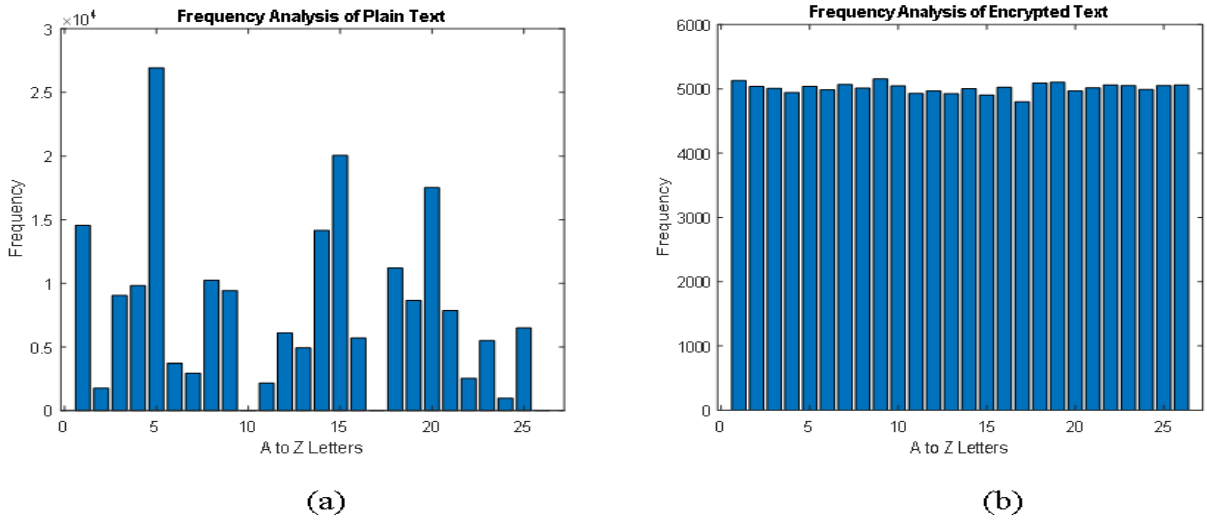

FIGURE 5. Frequency distribution of letters in (a) plain text, (b) encrypted text

Figure 5a demonstrates the frequency of each character in plain text. Different bar heights show which letters occur more often. In every natural language, some letters are more prevalent. While Figure 5b shows the frequency of each letter in the encrypted text. The frequencies are more consistent here than in raw text, indicating that the encryption process has masked the letter frequencies and hence makes it difficult to break the encrypted text through frequency analysis.

6.5. **Chi-square test.** An effective encryption algorithm generates a random output. The Chi-square is a method used to compare the distribution or frequencies of characters in ciphertext with the expected distribution of characters in a text. It assesses the randomness and uniformity of the ciphertext [27]. The Chi-square is calculated using the below equation [28].

$$x^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)}{E_i} \tag{11}$$

x^2: is the Chi-square value.
O_i: is the observed frequency of character i.
E_i: is the expected frequency of character i.
k: is the number of distinct characters (256 for ASCII range).

Table 4. The Chi-square results for different file sizes.

File Name	File Size (KB)	Chi-square test
Test_1	1	1.978
Test_2	3	1.847
Test_3	5	1.818
Test_4	10	1.765
Test_5	100	1.751

The result of a Chi-square test (1.7531) in Table 4 applied on various file sizes is relatively low, demonstrating that there is not a significant difference between the observed frequencies of characters in the encrypted files and the expected frequencies based on a uniform distribution. This indicates that the encrypted data displays a good level of randomness and uniform distribution, and the proposed algorithm is robust in producing secure ciphertext.

6.6. **Execution time.** Execution time refers to the amount of time it takes for an algorithm to convert plaintext into ciphertext (encryption) and to convert ciphertext back into plaintext (decryption). It's a critical metric for evaluating the efficiency and practicality of encryption methods, particularly in real-time applications. Several factors affect calculating the execution time, such as algorithm complexity, key length, and data size [29]. Table 5 shows the time values when executing the encryption for various sizes of plain text files. It is noted that when the file size increases, the encryption time also increases.

Table 5. The execution time results.

File name	File size (KB)	Encryption execution time (ms)
Test_1	1	203.221
Test_2	3	372.068
Test_3	5	652.411
Test_4	10	969.897
Test_5	100	4010.615

7. **Conclusion.** Due to the exponential increase in the number of cyberattacks and the continuous development of their tools, the need for new and robust security methods has become a significant issue. This research work presents a new encryption technique that employs Genetic Algorithms (GAs) to generate dynamic symmetric keys for securing textual data. The length of the key is adjusted based on the size of the input text, offering a flexible approach that maintains a substantial balance between encryption speed and data security. Due to its efficiency and adaptability, the proposed method is suited for encrypting confidential text files. Furthermore, the proposed approach depends on the GA only for creating a robust key, which reduces the complexity and computation costs. The effectiveness and

robustness of the proposed encryption algorithm were thoroughly assessed by applying six analytical tests: entropy, correlation, avalanche effect, frequency test, Chi-square test, and execution time. Our proposed approach has achieved a high level of randomness and uniformity in the generated ciphertext, thereby showing high resilience to potential cryptanalysis attacks. In future work, the limitations of the proposed method will be studied, such as developing the performance of execution time while maintaining high ciphertext security and increasing the complexity and time delay with increasing the text file size.

REFERENCES

[1] H. T. Elshoush, Mitigating Man-in-the-middle Attack in Online Payment System Transactions Using Polymorphic AES Encryption Algorithm, *J. Inf. Hiding Multim. Signal Process.*, vol. 14, no. 3, pp. 102–112, 2023.

[2] K. Sasikumar and S. Nagarajan, Comprehensive Review and Analysis of Cryptography Techniques in Cloud Computing, *IEEE Access*, vol. 12, pp. 52325–52351, 2024.

[3] H. A. A. Fursan Thabit, Ozgu Can, Asia Othman Aljahdali, and Ghaleb H. Al-Gaphari, Cryptography Algorithms for Enhancing IoT Security, *Internet of Things*, vol. 22, 2023.

[4] N. C. Nelakuditi, N. K. Namburi, J. Sayyad, D. V. Rudraraju, R. Govindan, and P. V. Rao, Secure File Operations: Using Advanced Encryption Standard for Strong Data Protection, *International Journal of Safety and Security Engineering*, vol. 14, no. 3, pp. 1007–1014, 2024.

[5] M. P. and M. Samreetha, A Review of Encryption and Decryption of Text Using the AES Algorithm, *International Journal of Scientific Research & Engineering Trends*, vol. 10, no. 4, pp. 400–404, 2024.

[6] W. Stallings, *Cryptography and Network Security Principles and Practice*, 8th ed., Pearson Publication, 2023.

[7] University of Babylon, *Classical Encryption*, [Online]. Available: https://www.uobabylon.edu.iq/eprints/publication_11_12227_49.pdf

[8] A. Putera, U. Siahaan, and R. Rahim, Dynamic Key Matrix of Hill Cipher Using Genetic Algorithm, *International Journal of Security and Its Applications*, vol. 10, no. 8, pp. 173–180, 2016.

[9] C. Chunka, R. S. Goswami, and S. Banerjee, An efficient mechanism to generate dynamic keys based on genetic algorithm, *Security and Privacy*, vol. 4, no. 5, pp. 1–10, 2021.

[10] C. Chunka, R. S. Goswami, and S. Banerjee, An efficient mechanism to generate dynamic keys based on genetic algorithm, *Security and Privacy*, vol. 4, no. 5, 2021, https://doi.org/10.1002/spy2.37.

[11] S. N. Sivanandam and S. N. Deepa, *Introduction to Genetic Algorithms*, Springer, 2008.

[12] M. Turčaník and M. Javurek, Cryptographic Key Generation by Genetic Algorithms, *Information & Security*, vol. 43, no. 1, pp. 54–61, 2019.

[13] M. I. Nazeer, G. A. Mallah, N. A. Shaikh, R. Bhatra, R. A. Memon, and M. I. Mangrio, Implication of genetic algorithm in cryptography to enhance security, *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 6, pp. 375–379, 2018.

[14] M. T. Gaata, Z. O. Ahmed, and R. S. Ali, Secure Big Data Transmission based on Modified Reverse Encryption and Genetic Algorithm, *Iraqi Journal of Science*, vol. 64, no. 1, pp. 439–451, 2023.

[15] A. Marjuni, N. Rijati, A. Susanto, D. Sinaga, P. Purwanto, Z. A. Hasibuan, and N. M. Yaacob, An optimization of advanced encryption standard key expansion using genetic algorithm and least significant bit integration, *Bulletin of Electrical Engineering and Informatics*, vol. 13, no. 6, pp. 4488–4497, 2024.

[16] M. Zengin and Z. Albayrak, Designing a new data encryption algorithm using a genetic code method, *Acta Polytechnica Hungarica*, vol. 19, no. 2, pp. 235–252, 2022.

[17] A. S. Sakr, M. Y. Shams, A. Mahmoud, and M. Zidan, Amino acid encryption method using genetic algorithm for key generation, *Computers, Materials & Continua*, vol. 70, no. 1, pp. 123–134, 2022.

[18] U. Menon, A. Hudlikar, and A. R. Menon, A Novel Chaotic System for Text Encryption Optimized with Genetic Algorithm, *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 10, pp. 34–40, 2020.

[19] E. Y. Baagyere, P. A. N. Agbedemnab, Z. Qin, M. I. Daabo, and Z. Qin, A Multi-Layered Data Encryption and Decryption Scheme Based on Genetic Algorithm and Residual Numbers, *IEEE Access*, vol. 8, pp. 100438–100447, 2020.

[20] V. V. N. Akwukwuma, F. O. Chete, M. N. Oshioluamhe, and A. E. Okpako, Text Encryption Using Advanced Encryption Standard (AES) Algorithm, *Journal of Science and Technology Research*, vol. 6, no. 2, pp. 214–228, 2024.

[21] K. Pandey and D. Sharma, Novel image encryption algorithm utilizing hybrid chaotic maps and Elliptic Curve Cryptography with genetic algorithm, *Journal of Information Security and Applications*, vol. 89, p. 103995, 2025.

[22] J. Bang, J. N. Kim, and S. Lee, Entropy sharing in ransomware: bypassing entropy-based detection of cryptographic operations, *Sensors*, vol. 24, no. 5, 2024.

[23] S. Yevseiev, S. Milevskyi, M. Melnyk, I. Opirskyy, M. Stakhiv, and R. Stakhiv, Entropy Method for Assessing the Strength of Encryption Algorithms, *Proc. 6th Int. Congr. on Human-Computer Interaction, Optimization and Robotic Applications (HORA-2024)*, July, 2024.

[24] M. Babaei, A novel text and image encryption method based on chaos theory and DNA computing, *Natural Computing*, vol. 12, no. 1, pp. 101–107, 2013.

[25] D. Upadhyay, N. Gaikwad, M. Zaman, and S. Sampalli, Investigating the Avalanche Effect of Various Cryptographically Secure Hash Functions and Hash-Based Applications, *IEEE Access*, vol. 10, pp. 112472–112486, 2022.

[26] M. Mohan, M. K. Kavitha Devi, and V. Jeevan Prakash, Security analysis and modification of classical encryption scheme, *Indian Journal of Science and Technology*, vol. 8, pp. 542–548, 2015.

[27] S. N. Al-Rekaby, M. A. A. Khodher, and L. K. Adday, A Hybrid Security System for Text Encryption and Steganography in Video Using Multi-Level Chaotic Maps, *International Journal of Safety and Security Engineering*, vol. 15, no. 3, pp. 521–532, 2025.

[28] Y. Shen, J. Huang, L. Chen, T. Wen, T. Li, and G. Zhang, Fast and Secure Image Encryption Algorithm with Simultaneous Shuffling and Diffusion Based on a Time-Delayed Combinatorial Hyperchaos Map, *Entropy*, vol. 25, no. 5, 2023.

[29] A. Lemma, M. Tolentino, and G. Mehari, Performance Analysis on the Implementation of Data Encryption Algorithms Used in Network Security, *International Journal of Computer and Information Technology*, vol. 4, no. 4, pp. 711–717, 2015.