

Optimization-based pseudo random key generation for fast encryption scheme

Shihab A. Shawkat^{1,4}, Najiba Tagougui², Monji Kherallah³

¹National School of Electronics and Telecommunications of Sfax, University of Sfax, Sfax, Tunisia

²Department of Computer Science Engineering, Higher Institute of Computer Science and Multimedia, University of Sfax, Sfax, Tunisia

³Department of Physics, Faculty of Science, University of Sfax, Sfax, Tunisia

⁴Department of Quality Assurance and Academic Performance, University of Samarra, Samarra, Iraq

Article Info

Article history:

Received Oct 6, 2022

Revised Nov 2, 2022

Accepted Nov 13, 2022

Keywords:

Cryptography

Fitness function

Genetic algorithm

Information security

Text encryption

ABSTRACT

Data encryption is being widely employed to secure data in order to provide confidentiality, authenticity and data privacy. In proposed method a novel method for (encryption, decryption) message through optimization using genetic algorithm (GA) is presented which aims to maintain the security of the message via increasing the complexity of the key used in the encryption. The proposed method is named stream cipher randomization using GA (SCRGA). It is characterized by its secrecy by hiding the statistical properties for the input message. The GA increases the key diffusion in order to eliminate the likelihood of using the statistical analysis and cryptanalysis techniques to obtain the original text. A key (k) will be supplied as input to pseudorandom bit generator and then it produces a random 8-bit output, The SCRGA evaluates the candidate keys and it would select the one that achieves the best value of the fitness function by maximization three criteria considered in matching the input message and the key. The test results were based on the comparison of the proposed method with two other state-of-the-art methods, that the SCRGA outperformed the other two methods in terms of the execution time (encryption, decryption) and encryption rounds key size.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Shihab A. Shawkat

National School of Electronics and Telecommunications of Sfax, University of Sfax

Sfax, Tunisia

Email: shahab84ahmed@gmail.com

1. INTRODUCTION

In the modern usage, rapid data interchange has largely defined how the internet has developed, and the extensive and thorough use of devices by users has enabled them to communicate with one another and share information. As a result, it is concluded that it is important to protect the devices and the exchange of information between them [1]. Multimedia streams are currently simple to obtain thanks to the ubiquitous usage of the wireless networks and internet, the exploding expansion of consumer devices, and improvements in multimedia compression methods. In an effort to safeguard the aforementioned multimedia materials and grantee that any additional data is effectively hidden inside such digital files [2]. Genetic algorithm (GA) can consider one of the modern methods, as the importance of using it in solving complex problems has emerged, in addition to solving difficult problems in research operations, as well as solving coding issues and cracking the code [3]. General of how to produce new individuals that possess certain characteristics (desirable or undesirable) through modification, overlap or alteration that obtains the inherited groups with the aim of

forming new individuals. GA has been used in the present research to generate the stream cipher key, after reviewing some previous research that dealt with the uses of GA with encryption and decryption [4].

Video, text, audio, and images are the most common types of media. Despite the existence of numerous steganography methods, they are vulnerable to statistical, structural, and visual attacks. Texts with hidden data are anticipated to have greater entropy compared to those without in terms of text steganography [5]. Systems like stream ciphers are categorized as secret key systems since they only use one secret key for both decryption and encryption. These systems are the most popular and frequently utilized in encryption due to a number of crucial features, such as: (no errors in the event of occurrence, simplicity of usage in real applications, implementation speed. As a safe method of keeping information, stream cipher is one of the most significant relatively modern encryption techniques used in communication and storage systems [6]. No encryption technique is impenetrable by those who are determined to get inside. Using keys only once with stream cipher is the best approach to guarantee security. Each key must also be independent of the others used. In this manner, the rest of the system's security will be unaffected even in the case where attackers successfully crack one of the keys [7]. With the aid of a few algorithms depending on mutation, crossover, and selection, among other things, we will describe the fundamentals of GA, decryption, and encryption in this part. Under this idea, a vast method associated with generating secret keys for cryptography and random numbers is used.

- Generic algorithm

One of the general search algorithms depending on the mechanism of natural selection and the natural genetic system is the GA, a heuristic search algorithm that draws inspiration from Charles Darwin's theory of natural selection. It is intended for the inherited groups (adding or replacing certain genetic materials) in order to form individuals with good traits, and on this basis the GA selects the preferred solutions from a large number of solutions and makes some overlaps and alterations between these solutions in order to create better solutions [8]. The GA is a method of artificial intelligence and it is one of the modern methods, as the importance of using this method has emerged in solving complex problems (large in size and has a huge number of alternative solutions) in an appropriate time. The solution resulting from the application of the GA is often a near optimal solution, and this method, when applied, provides an intelligent search among a large number of alternative plans [9].

The method of GA in solving various problems depends on ideas deduced from genetics, which is generally concerned with how to produce new individuals that have certain characteristics (desirable or undesirable) through modification, interference or switching that obtains the inherited groups in order to form new individuals [10]. GA are optimization techniques that use an evolutionary process. The solution to the problem is a data structure known as chromosomes. The quality of the solution is evaluated with a function called the fitness function, and a series of initial solutions (a random community) is generated through a combination of processes similar to an evolutionary process. A certain value is chosen for the stop. Before proceeding to apply the GA, there is a necessity in encoding data-set in any of the formats, like bit encoding. We consider a group of the solutions for a problem and choose the set of optimal ones amongst them. Five phases have been considered in the GA [11]:

- a. Do initial population: a population, a set of individuals, is where the process starts. Every individual is a component of the solution to the problem you're trying to address. The genes can be described as a set of parameters or variables defining an individual. A chromosome (solution) consists of a string of genes. A string, or an alphabet, is used in a GA to represent an individual's set of genes. Binary values are typically utilized (string of 0 s and 1 s). We refer to genes on a chromosome as being encoded.
- b. Fitness function: which gauges an individual's fitness level (an individual's ability of competing with other individuals). Each individual receives a fitness score. Depending on its fitness score, an individual's probability of being selected for the reproduction has been determined.
- c. Selection: the purpose of the phase of the selection is to choose the most fit individuals and allow them to pass on genes to the following generation. Depending upon their scores of fitness, two sets of individuals (also referred to as the parents) have been selected. High fitness individuals are more likely to be selected for the reproduction.
- d. Crossover: the crucial stage of a GA is crossover. A random point of crossover from among genes is selected for every pair of parents before mating.
- e. Mutation: a few of newly produced offspring's genes have susceptibility to random mutation of low-probability. This suggests that a few bits in the bit string could be reversed.

This step overcomes the issue in the case where the value of the fitness at some point is stuck at local minima. None-the-less, it's stopped typically when certain condition of termination has been met. Some of the common conditions of termination are [12]; i) a predetermined number of the generations had passed, ii) a satisfactory solution was found, and iii) no improvements in the quality of the solution have happened for a specific number of the generations converging to a solution. The evolutionary cycle may be summarized

as; i) generation=0, ii) seed population, iii) while not (termination condition) do, iv) generation++, and v) compute the value of the fitness.

- Stream cipher

One byte or bit of the plain-text is encrypted at a time by a stream cipher, a type of encryption algorithm. The key represents an infinite stream of the pseudo-random bits. An implementation of the stream cipher must have an unpredictable pseudo-random number generator and never use one key twice to remain secured. Stream ciphers aim to come close to an idealized cipher, which is meant to use a completely random key, might be able to attain "perfect secrecy". It is intended to be completely resistant to brute force attacks [13]. The pseudorandom bit generator will receive a key (k) as input, and it will after that produce a random 8-bit output that is referred to as key-stream. The final keystream will be 1 byte in size, or 8 bits. Following a stream of pseudo-random numbers is how the stream cipher operates. In the stream cipher, every one of the digits of plain-text is encrypted individually with corresponding key-stream digit for producing a cipher-text stream digit. It is sometimes referred to as a state cipher since the encryption of every one of the digits depends upon the current state of the cipher. In reality, a digit is often a bit, and an XOR is used to combine them [14]. Bits are processed in a chain, one by one. High speed and simplicity of the hardware. A key and a lengthy initialization vector with no repetition are frequently coupled. If properly designed, it is equally secure. Usually really quick and simple. There are many advantages of using stream cipher, including [15]:

- Speed: this encryption form is usually faster than others, which include the block ciphers.
- Low complexity: it is easy to incorporate the stream ciphers into the modern programs, and the developers do not need complex hardware in order to make it happen.
- Ease of use: stream ciphers are tools of symmetric encryption; therefore, the companies are not forced to bother with the private and public keys. The mathematical concepts which underlie the modern stream ciphers allow the computers to decide the correct decryption key to be used.

Various organizations and people utilize stream ciphers because they are simple to implement. The majority of websites and web browsers use this technique. Making cryptanalysis more challenging is one of the advantages of using stream cipher, hence the number of the bits that have been selected for the keystream should be large so as to achieve this. It is also secure from brute force attacks because to the longer key. Stronger security is achieved, preventing any attack, the longer the key. To make cryptanalysis more challenging, keystream could be designed more effectively by having additional 1 s and 0 s. One of the significant advantages of the stream cipher is that it uses less code compared to a block cipher [16]. stream cipher divide the plaintext P into binary orders, P_1, P_2, P_3 , or sequential symbols, and encrypt each E_i using the K_i of the key sequence k_1, k_2, k_3 , as (1).

$$E_{ki}(P) = E_{k1}(P1)E_{k2}(P2)E_{k3}(P3) \quad (1)$$

The encryption process is as in (2).

$$C_i = E_{ki}(P_i) = P_i \oplus K_i \quad (2)$$

where C_i, P_i, K_i represent the binary orders of a key, the original and the cipher respectively and \oplus represents the XOR. The properties of the resulting randomness were relied upon by applying the conditions of approved randomness. Plaintext and key-stream produce the cipher text (an identical key-stream will be utilized for the decryption). Plain-text will undergo the XOR operation with the key-stream bit-by-bit and produce cipher text. Figure 1 for instance, if the next byte generated by the generator is plaintext: (1001 1001), key-stream: (1100 0011), cipher text: (0101 1010). The cipher text and key-stream give the original plaintext (by using the same keystream for the encryption). Cipher-text will undergo the XOR operation with the key-stream in a bit-by-bit manner and produce the original plaintext. Example: cipher text: (0101 1010), key stream: (1100 0011), plaintext: (1001 1001) [17].

- Hashing

It is a mathematical function whose input is a series of variable length data that represents the message data itself and may be applied to the key as well. The camouflaging function converts the variable length (random) of the entered data into a series of fixed length data whose length is usually less than the length of the data. The data entered. The output of a good hash function must be equally distributed, and it should also have pseudo-randomness, one-way characteristic, and collision resistance. Finding a message which maps to some specific message digest (one-way property) or two messages mapping to same message digest must be computationally efficient yet computationally impossible (collision-free property). Hash functions are employed to make sure that a message's integrity is preserved because of these characteristics [18]. Through it, we can distinguish the original message and accurately identify it, so that any changes in the message key, even if it is in one bit, will result in an entirely different fingerprint, it is not possible to derive two equal functions for two different messages, and it is also used to ensure that the message came from its

source without being exposed to any change during transportation. The simple hash function has been used. This function performs the XOR operation according as (3) [19]:

$$H_i = B_{i1m} \oplus B_{i2m} \quad (3)$$

where H_i (binary coding sequence in camouflage), M (number of binary coding in the second input), B_i (binary coding sequence in specified), the original and the cipher respectively and \oplus represents the XOR.

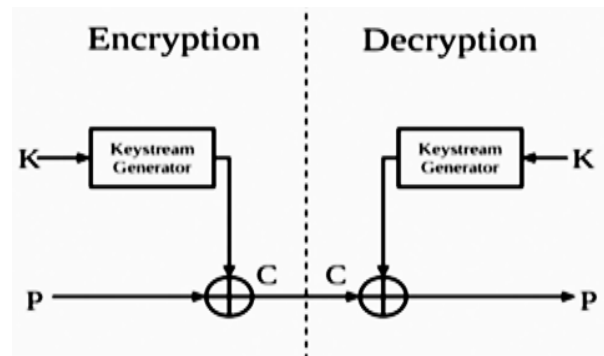


Figure 1. Diagram of stream cipher

Figure 2 represents the basic operation of hash function. Generally, integer contains message multiple of some fixed length which is padded onto it and padding contains the value of original message length that is in bits. Length field provides security measure which increases the trouble for an external attacker aiming to produce another message using same hash value [20]. Several of the most important works on multimedia and data privacy that make use of steganography and many associated technologies. The significance of healthcare services has recently come to light on a global scale, and it has become one of the most important research areas in academia and business. Yet, to guarantee seamless and error-free procedures, the best data privacy must be preserved. The secrecy of electronic data sent over the internet is covered in [21]. The use of GAs with pseudorandom sequences for encrypting and decrypting data streams as a novel strategy for e-security applications. The method by which the strings can combine and contrast their appealing traits at random. We choose two new random strings as strings to the right of randomly chosen point in the two strings are then exchanged. This process starts by selecting a random location in both strings. In this manner, information is exchanged between the strings, and pieces of two strings are joined and exchanged.

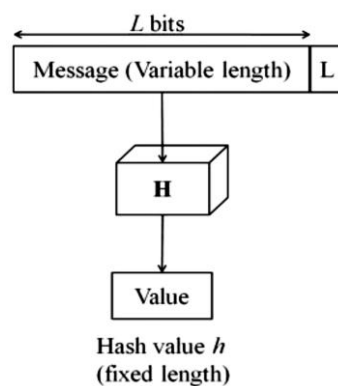


Figure 2. Block diagram of cryptographic hash function

To enhance both the security and the quality of the steg image, a more secure technique is put forth in [22]. GA can be defined as a semi-blind algorithm; therefore, it might choose a key that compromises security. Using the GA and optimal pixel adjustment process (OPAP), a more secure image hiding method. As a result, the security is implemented by applying image transformation utilizing both a user key and the GA key. The

hidden image's pixel locations are rearranged using the user key. The GA after that uses OPAP to choose the key that maximizes the stage-quality images and security. The approach employs an easy-to-use transformation technique that sharpens the contrast between the secret image and its altered counterpart. Data encryption is frequently used in [23] to assure data security. A group of optimization methods are known as GAs. Encrypting and decrypting data streams by means of GAs with pseudorandom functions. To enable the algorithm to be used with any type of .txt, the encryption process is applied to a binary file. We essentially utilized 5 keys in the algorithm, and we are now using blum shub pseudorandom number generator (PRNG).

According to Alhussain [24], it is demonstrated how to create a key exchange algorithm based on the GA's mutation and crossover operations and asymmetric key encryption. By permuting the key by a random permutation factor, the number of crossover points and mutation points determine the length of secret key. The combination of randomization and permutation makes the algorithm resilient and difficult to break. The text encryption algorithm based on the block cipher and chaotic maps has been suggested in [25] and decrypted for blocks of (88) bytes. The previously constructed nonlinear substitution S-box component. To create the key sequences utilized in decryption and encryption, a random key generator depending on tent map is employed, which depends on a 2D logistic map and a 2D cross-chaotic map. Pseudorandom numbers were generated using the multiplicative congruential generator approach in [26]. The aforementioned suggested algorithm has been tested on a variety of text files containing data like numeric values, plain text, and special characters. Have utilized a GA-based cryptography algorithm for data decryption and encryption.

According to Agbedemab *et al.* [27], a new text encryption and decryption approach that has been referred to as the GA residue numbers (GARN) is proposed. This three-layered text scheme has a high throughput rate and can encrypt both smaller and larger messages. It can encrypt and decrypt any character or symbol with the use of GA and some inherent properties from residue number system (RNS).

The mutation and crossover operators of a GA are used for generating a new session key in [28], which developed a cryptographic algorithm depending on session key for each image encryption. Uses 64-bit plain text and needs an 80-bit key, with the remaining 16 bits of the key being added randomly. The 64-bit key is created using a symmetric hexadecimal key. To protect diagnosis data in medical images, [29] introduces a new data encryption model hybridization. The suggested model has been created through combining the suggested hybrid encryption technique with either 2D discrete wavelet transform 1 level (2D-DWT-1 L) or 2D discrete wavelet transform 2 level (2D-DWT-2 L) steganography. The advanced encryption standard (AES) and Rivest-Shamir-Adleman (RSA) algorithms are strategically combined to create a hybrid encryption scheme, which secures the diagnosis data that will be incorporated with RGB channels of a medical cover image. The application of an adaptive GA for optimal pixel adjustment process (AGA-OPAP) that enhances features for imperceptibility and data hiding.

2. STREAM CIPHER RANDOMIZATION USING GENETIC ALGORITHM

In order to obtain more confidential encryption, faster implementation, and cost capturing, a hybrid algorithm has been proposed to take advantage of the genetic properties to generate a key and verify its randomness using approved methods. Instead of regenerating the key at the receiving party, a new method was used to hide the key inside the ciphertext before sending it, and the integrity of the key you used was confirmed using camouflage. In the work structure of the proposed method, the GA was relied on to avoid the disadvantages caused by the traditional methods. Initially, a random generation of the key was generated using the rand function, noting that the key length must be the length of the text to be encrypted, then the randomness is measured for each member of the generation according to the approved conditions.

2.1. Encryption phase

In the encryption, the algorithm hides the key inside the cipher text after the generation of the key, the encryption starts by XORing the resulted key with the text to be hired as can be listed in the following steps:

- Message encoding

For generating the key, we need first to check the length of the message that need to be encrypted. For the word “abc” the American standard code for information interchange (ASCII) code for the “a” is 097, “b” is 098 and for “c” is 099. Its noteworthy to mention that the message encoding considered is the binary code of each ASCII number corresponding to the letters in the message. Thus, the “abc” message is first transformed into “097 098 099” and then it is transformed to “01000001 01000010 01000011”. The length of the message now is 3×8 which is 24-bit length that represents the key-length at same time. The key will be randomly generated whose values are uniformly distributed between 0 and 1. Table 1, shows the ASCII code and its binary representation of the input message.

- Generate a random key: the same length of the input message.
- Check the generated key quality

Table 1. Message ASCII code and its binary encoding

letter	ASCII code	Binary	Letter	ASCII code	Binary
a	097	01100001	A	065	01000001
b	098	01100010	B	066	01000010
c	099	01100011	C	067	01000011

If the quality of the generation key according as (4) was good enough i.e., the get obtains the maximum value of (4), that would suggest the key wouldn't need to go through the key-optimization-process. It will be used right away in the encryption process. However, if the key failed to satisfy the equations mentioned above, then it needs to undergo the key-optimization-process. In the optimization process, the GA would be used to optimize (4). In order to find the value of (4) assume the count of 0 s is represented by x while the count of 1 s is represented by y . Also, we have n that represents a threshold value. Then, in order to find the objective function of the input key, we have to calculate f_1 , f_2 , and f_3 and add them together as shown in (4). Thus, we first calculates the number of 1 s and 0 s in the key and returns the value of f_1 either zero in the case of unequal or one in the case of equality while the second function starts when we enter the value of n that represents the number of block units required and $n=8$ and search for these n within the key string and returns values either zero in the absence of a block of size n or one in the case of a block of size n and also returns either zero in the absence of the presence of a gap of measure $n-1$ or one in the case of a gap of measure $n-1$.

$$f(x) = \sum_{i=1}^m f_i, m=1,2,3 \quad (4)$$

$$f_1 = f(x) = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases} \quad (5)$$

$$f_2 = f(x) = \begin{cases} 1, & x \geq n \\ 0, & x < n \end{cases} \quad (6)$$

$$f_3 = f(x) = \begin{cases} 1, & y \geq n - 1 \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

– Key optimization process

In this step, the evolutionary optimization is applied in order to generate a stronger key than the one resulted in the last step. The same criteria used in the last step, used here in the evolutionary process as an objective function i.e., in (4). However, this time it is used iteratively. The process starts from generating four new random solution. Thus, we can say that the population equals 4 solutions. Each single solution of these four candidate solutions is evaluated simultaneously using (4) until we find the optimum or the near optimum solution after several generations. The optimization continues until a satisfactory value of the fitness function is obtained for anyone one of the candidate solutions. It is expected that during the consecutive generations, the solutions highly ranked will be used for the message encryption.

– The initial population is first evaluated using the fitness function

This fitness function represents a maximization function that means that solutions having the highest fitness value is passed to the next generation. The best couple of solutions in the population is selected for the crossover and the mutation. Again, the fitness function is applied on the resulted new solutions. At this point the solution that achieves higher fitness than its own parent or any other individual in the population, then parents are substituted by their children. Now the previous step's output will work as an input of the operation of mutation. After mutation we will get the final key that would be utilized in the process of the encryption. Figure 3 shows the process of the key generation using GA to be utilized in encryption. Key length must first be determined so as to decide the GA working parameters before starting optimization process. The proposed parameters for the GA process as follow; i) generations=determined by reaching to the best solution according to the fitness function, ii) initial population=4, iii) selection=roulette-wheel, iv) crossover=one-cut point crossover, and v) mutation=uniform with 0.001 probability. Encryption: the text to be encrypted is XORed with the random key generated either from step 4 or 3.

– After the completion of the XOR operation, the hash function is calculated for the key.

– After encryption

After the encryption a frame will be generated that has the following elements; i) the generated key and the encrypted message with zero padding among its bits, ii) the hash no. That is generated from the secure hash algorithm 1 (SHA-1) that takes an input then produces a 160-bit (20-byte) hash value that is referred to as message digest—usually rendered as 40 hexadecimal digits, iii) zero-bit added after the hash,

iv) key location generated from the random permutation with two zeros padding among its bits, and
v) finally, the size of the message character is appended as a last part of the frame. All the steps explained earlier, are summarized in Figure 4.

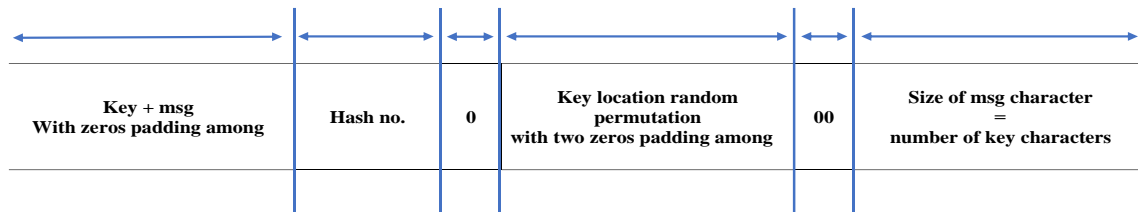


Figure 3. Encrypted message frame

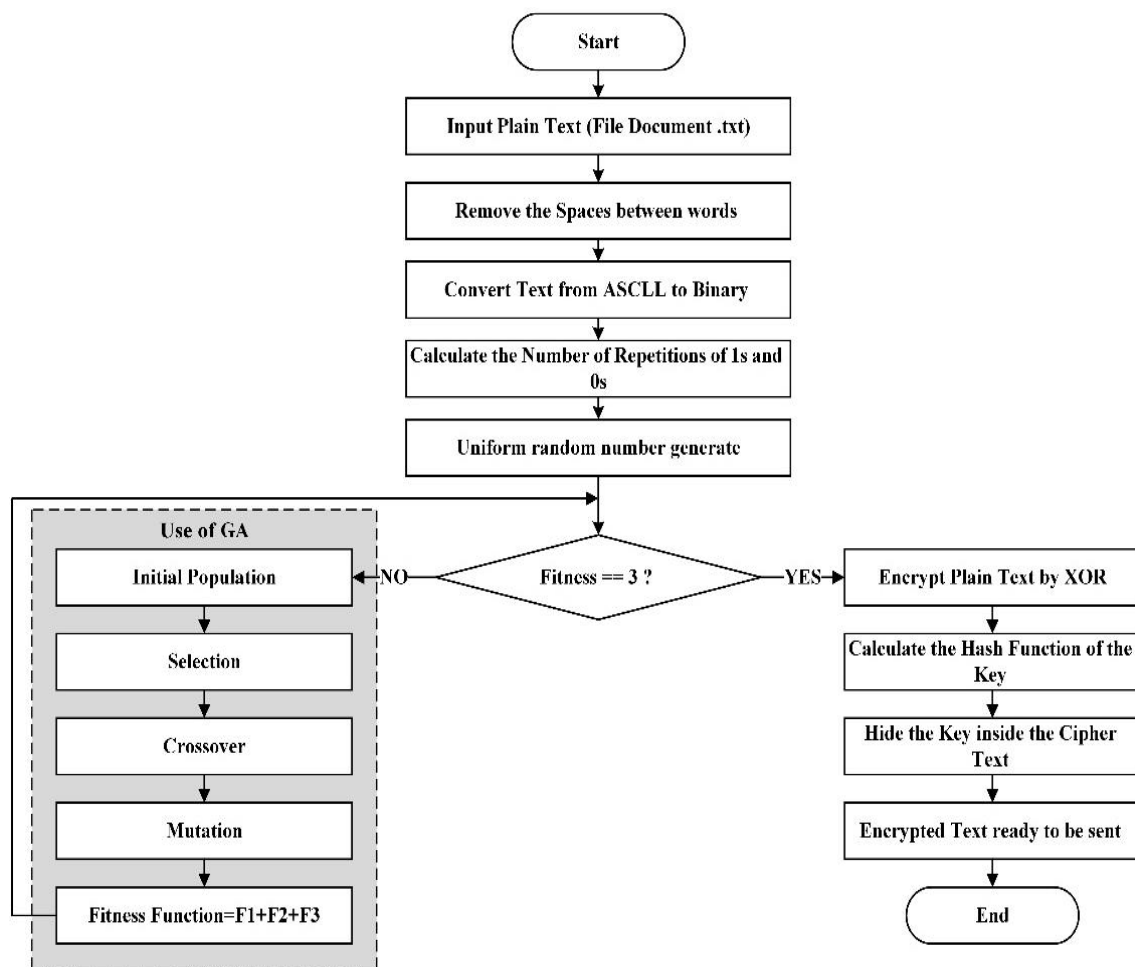


Figure 4. Encryption process of proposed method

2.2. Decryption phase

The decryption here is the process of extracting the original message from the frame illustrated in Figure 5. The decryption starts first from splitting the frame of 1-D array of elements and retrieve the last element that represents the length of the message. Afterwards, the key location is extracted with removing the appended two zeros. Then all other zeros added to the secret message are all removed. At this point the message location must be found by extracting only key by its location, extracting the secret message by its location and finally extracting the hash number from the frame. It is very important to convert the binary array of key the message into decimal numbers.

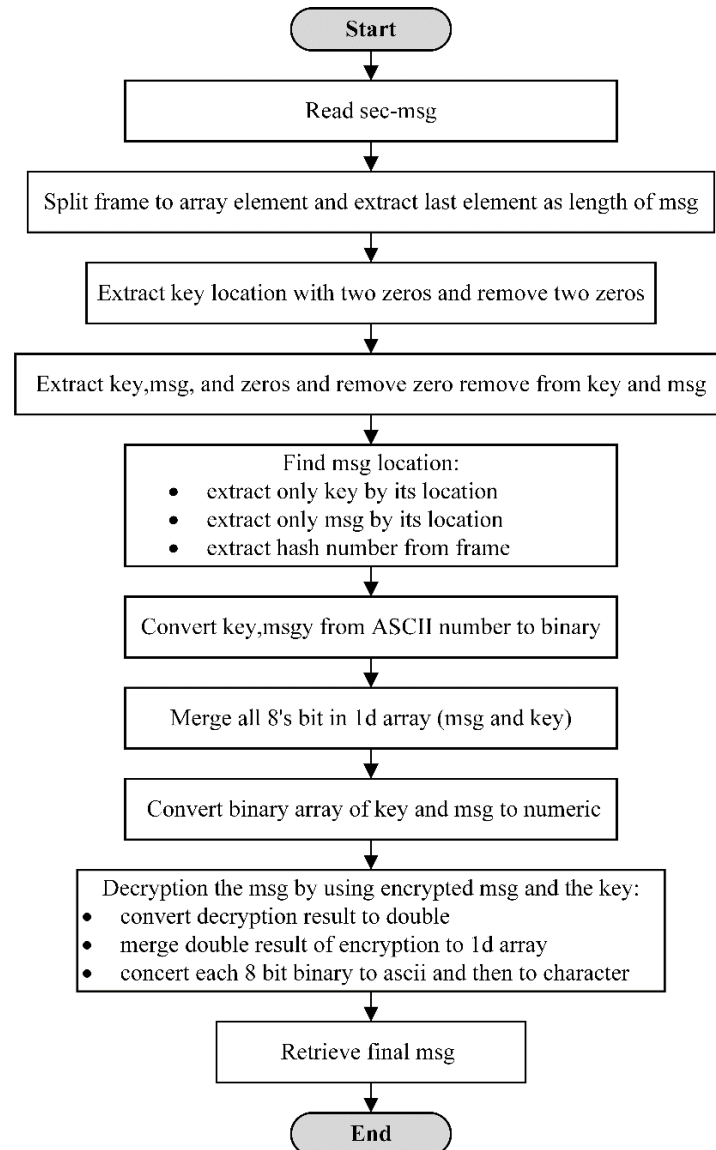


Figure 5. Decryption process of proposed method

3. RESULTS AND DISCUSSION

The simulation was done using MATLAB® R2021b and a CORETM i5 microprocessor laptop. The simulated results are shown in Table 2 with their average times of execution. The proposed scheme was tested on different types of text files containing (letters, numbers and symbol) characters. The proposed algorithm has the ability to encrypt and decrypt any character and size from ASCII table. Its relating ciphertext utilizing the proposed algorithm and the key.

3.1. Suggested method graphic user interface

The developing a simulation environment is very vital in order to verify the proposed algorithms and its performance. Experiments confirm that the developed approach allows us to receive a more readable and easier-to-interpret structure in the form to obtain much lower learning and testing errors compared to other techniques. The proposed systems integrate three modules (normal method, GA, my proposed) that together constitute the experimental application is encrypting and embedded, then decoding and extraction text using encryption techniques. This frame is represented in Figure 6.

3.2. Experimental results

The proposed algorithm has the ability to encrypt and decrypt any character and size from ASCII table. Its relating ciphertext utilizing the proposed algorithm and the key. We note in Table 2 that the

encryption and decoding time of texts are good and are not subject to any rule because they depend in text encryption on the process of creating the key using the GA. We also note that the decoding time is less than the encoding time because in the decoding process the GA is not used. In Table 2 the results of the statistical parameters that have been obtained after the use of various textual data files that possess letter, special characters of different sizes. This table has the text size measured in bytes and stored as a txt file. It also has the encryption, decryption execution time measured in seconds. Besides, it contains the encryption round number and finally the key size measured in bits. The number of symbols considered is randomly chosen between 1 and 15 symbols. They represent different text file sizes ranges between 10 bytes to 1,000 byte.

The screenshot shows a software interface for testing encryption and decryption. The input text is "I live in Iraq". The text is converted to binary. A random key is generated. The user selects the "My Proposed" method. The interface displays the XOR product, the product of the hash function, the encrypted message, the key locations to add, the final encrypted text, the encryption time (0.0079962 Second), the decrypted text, and the decryption time (0.000159 Second). The round key GA is 3. A "Clear all" button is present at the bottom right.

Figure 6. Experimental application of our proposed method

Table 2. Results of statistical parameters obtained from applying on different types of text files containing (letters, numbers and symbol) with different text sizes

No.	Text size (byte) .txt	Encryption execution time (s)	Decryption execution time (s)	Encryption rounds	Key size (bits)
1.	10	0.0053265	0.0001729	1	80
2.	20	0.0079821	0.0001089	2	160
3.	30	0.023734	0.0001479	3	240
4.	40	0.024326	0.0001492	3	320
5.	50	0.042311	0.0001541	5	400
6.	60	0.048751	0.0001661	5	480
7.	70	0.053775	0.0001998	5	560
8.	80	0.065280	0.0002210	6	640
9.	90	0.027827	0.0002599	6	720
10.	100	0.061645	0.0009064	7	800
11.	150	0.066103	0.0004002	4	1200
12.	200	0.12407	0.0004949	6	1600
13.	256	0.17085	0.001397	6	2048
14.	512	0.25468	0.0010366	10	4096
15.	1000	0.54936	0.00207878	12	8000

As can be realized in Table 2 that there is a direct correlation between the text file size with the encryption execution time. For instance, if the file size equals to 10 bytes only, the encryption execution time would equal to 0.0053265. Similarly, when the file equals to 1,000, once can notice that the encryption execution time has increased according to the file size. When it comes to the decryption execution time, another scenario can be observed. It is noticed that there is no direct relationship between the file size and the decryption execution time. For example, in the first test when the file size equals to 10, the decryption time

equaled to 0.0001729, as for the next text when the file size was increased to 20 there was a similar decryption time to the file when it was only 10 bytes which was almost equal to 0.0001. Until we reach to the file size that equals to 70, the decryption time still almost the same. It has increased slightly when the file size equaled to 70, until it reached its highest decryption time when the file size equaled to 1,000 byte. In all cases when the decryption time is compared to the encryption time, it can be clearly perceived that the encryption time is much higher in all cases. Figure 7 shows the relationship between file size on both encryption and decryption execution time.

The execution speed of the encryption and decoding algorithm was measured to ensure its speed and the results were as shown in the Table 2. A decent cryptosystem must be sensitive at secret keys this implies ciphertext created utilizing somewhat diverse keys ought to be altogether different. The ciphertext with inaccurate key does not demonstrate any data related with plaintext, hence the proposed algorithm is sensitive to secret key.

Thus, in Figure 7, once can notice in both Figures 7(a) and (b) there is an increase in the time required for the encryption due to the need to calculate the key using the proposed method. However, the extra time need at the encryption is insignificant. When it comes to the decryption, the time required for the decryption is less than that needed for the encryption because there is no need to calculate the key again.

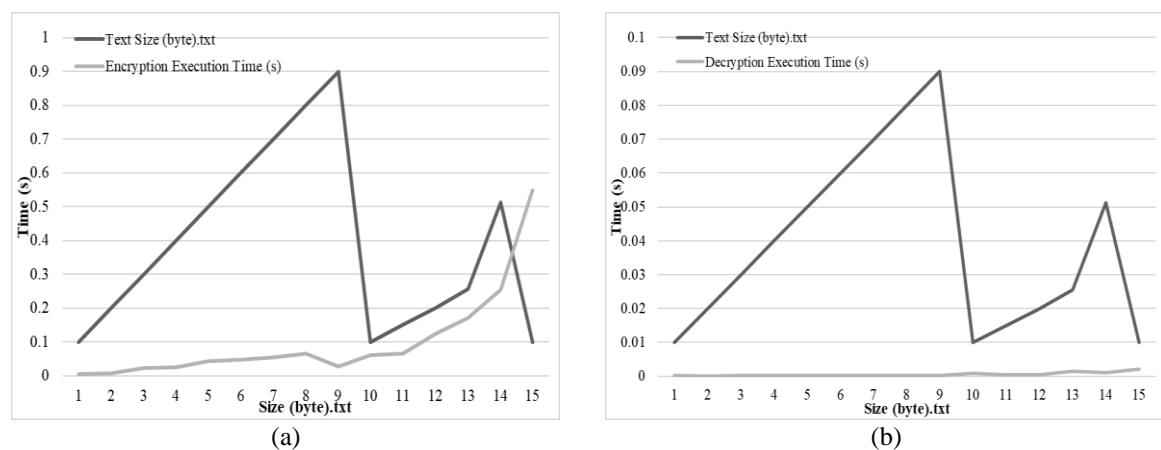


Figure 7. Relationship between file size on both (a) encryption and (b) decryption times

In Table 2, we note that the generated key goes through the number of rounds in our proposed method. Every time the number of rounds varies according to the text size and the number of rounds, because the process of generating or creating the key is done using the GA and the randomness of the key generated in each implementation is completely different from the one before it completely. This is what it distinguishes the proposed method. As well as the length of the key, the length of the text entered is $2n$ and a different number. Depending on the ideal result that you reach when using the GA and compared to the previous algorithms.

3.3. Comparing the results of our proposed techniques with other results

The performance of our model was compared with another technique developed by stream cipher and GA was also compared with another approach developed by reference [28], [29]. However, this reference [28] doesn't include any representation of its results in a form of table or figures. However, the results showed that the proposed scheme performs favorably better, reveals the higher performance of our proposed model. The runtimes of different text files are shown in Table 3.

Table 3. The performance results against of our model was compared with other techniques developed the of all different files

Comparison between other method	Text size (byte).txt	Parameter			
		Encryption execution time (s)	Decryption execution time (s)	Encryption rounds	Key size (bits)
[28] (2020)	80	---	---	5	640
Proposed method	80	0.035176	0.0005223	4	640
[29] (2021)	250	12.97	14.01	---	---
Proposed method	250	0.15842	0.0007725	10	2000

4. CONCLUSION AND FUTURE WORK

This paper introduced a new text encryption and decryption technique using streamlined encryption and GA is characterized by the possibility of implementing it in the case of the availability of the applicable system. In addition to the difficulty of obtaining the encryption key within the ciphertext designed for optimal competence, the algorithm was implemented using the MATLAB program. Various parameters such as text and key lengths, calculation of encryption and decryption time, number of attempts to reach perfect encryption were tested. The simulated results show that the proposed scheme is very chaotic and robust and the throughput rate of the scheme (runtime) competes very favorably with existing similar schemes. The proposed method is also characterized by confidentiality due to the randomness of the key, which leads to hiding the statistical properties of the plaintext language and knowing part of the key sequence is not helpful in knowing all sequences are non-recursive, as in the well-known linear and nonlinear shift registers. That is why the method is characterized by its stability in front of the known clear text attack. In future, it is also possible to rely on other intelligent techniques to generate the key, such as the use of neural networks. In addition to the possibility of integrating more than one encryption algorithm and benefiting from the GA by generating the key to it, and the use of the well-known camouflaging function that achieves the required specifications.




REFERENCES

- [1] S. S. Reka and T. Dragicevic, "Future effectual role of energy delivery: a comprehensive review of Internet of Things and smart grid," *Renewable and Sustainable Energy Reviews*, vol. 91, pp. 90–108, Aug. 2018, doi: 10.1016/j.rser.2018.03.089.
- [2] M. U. Hassan, M. H. Rehmani, and J. Chen, "Differential privacy techniques for cyber physical systems: a survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 746–789, 2020, doi: 10.1109/COMST.2019.2944748.
- [3] Y. Zhang, P. Li, and X. Wang, "Intrusion detection for IoT based on improved genetic algorithm and deep belief network," *IEEE Access*, vol. 7, pp. 31711–31722, 2019, doi: 10.1109/ACCESS.2019.2903723.
- [4] G. B. -Orgaz, S. S. -Sanz, and D. Camacho, "A multi-objective genetic algorithm for overlapping community detection based on edge encoding," *Information Sciences*, vol. 462, pp. 290–314, Sep. 2018, doi: 10.1016/j.ins.2018.06.015.
- [5] S. M. R. -Larmer, "Cover text steganography: N-gram and entropy- based approach," *2016 KSU Conference on Cybersecurity Education, Research and Practice*, pp. 1–8, 2016.
- [6] J. A. Sarumi, "A review of encryption methods for secure data communication," in *Proceedings of the 30th iSTEAMS Multidisciplinary & Inter-tertiary Research Conference*, May 2022, pp. 63–82, doi: 10.22624/AIMS/iSTEAMS/LASUSTECH2022V30P7.
- [7] D. Puthal, X. Wu, N. Surya, R. Ranjan, and J. Chen, "SEEN: a selective encryption method to ensure confidentiality for big sensing data streams," *IEEE Transactions on Big Data*, vol. 5, no. 3, pp. 379–392, Sep. 2019, doi: 10.1109/TBDATA.2017.2702172.
- [8] L. Zhang, P. H. Aaen, T. Dhaene, A. Sahu, and V. Devabhaktuni, "An introduction to evolutionary optimization for microwave engineering," *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–17, 2015, doi: 10.1002/047134608x.w8286.
- [9] A. F. Ali and M. A. Tawhid, "A hybrid particle swarm optimization and genetic algorithm with population partitioning for large scale optimization problems," *Ain Shams Engineering Journal*, vol. 8, no. 2, pp. 191–206, Jun. 2017, doi: 10.1016/j.asej.2016.07.008.
- [10] P. Kora and P. Yadlapalli, "Crossover operators in genetic algorithms: a review," *International Journal of Computer Applications*, vol. 162, no. 10, pp. 34–36, Mar. 2017, doi: 10.5120/ijca2017913370.
- [11] S. Sarkar, A. Lohani, and J. Maiti, "Genetic algorithm-based association rule mining approach towards rule generation of occupational accidents," in *Computational Intelligence, Communications, and Business Analytics*, Singapore: Springer, 2017, pp. 517–530, doi: 10.1007/978-981-10-6430-2_40.
- [12] A. Kumar and M. K. Ghose, "Overview of information security using genetic algorithm and chaos," *Information Security Journal: A Global Perspective*, vol. 18, no. 6, pp. 306–315, Dec. 2009, doi: 10.1080/19393550903327558.
- [13] Y. Zhang, "The image encryption algorithm based on chaos and DNA computing," *Multimedia Tools and Applications*, vol. 77, no. 16, pp. 21589–21615, Aug. 2018, doi: 10.1007/s11042-017-5585-x.
- [14] G. Millerioux and P. Guillot, "Self-synchronizing stream ciphers and dynamical systems: state of the art and open issues," *International Journal of Bifurcation and Chaos*, vol. 20, no. 09, pp. 2979–2991, Sep. 2010, doi: 10.1142/S0218127410027532.
- [15] D. Xiao and F. Y. Shih, "Using the self-synchronizing method to improve security of the multi chaotic systems-based image encryption," *Optics Communications*, vol. 283, no. 15, pp. 3030–3036, Aug. 2010, doi: 10.1016/j.optcom.2010.03.063.
- [16] Z. Zhang *et al.*, "High-efficiency parallel cryptographic accelerator for real-time guaranteeing dynamic data security in embedded systems," *Micromachines*, vol. 12, no. 5, pp. 1–24, May 2021, doi: 10.3390/mi12050560.
- [17] R. Asaad, S. Abdulrahman, and A. Hani, "Partial image encryption using RC4 stream cipher approach and embedded in an image," *Academic Journal of Nawroz University*, vol. 6, no. 3, pp. 40–45, 2017, doi: 10.25007/ajnu.v6n3a76.
- [18] A. Neelima and K. M. Singh, "Perceptual hash function for images based on hierarchical ordinal pattern," in *Handbook of Multimedia Information Security: Techniques and Applications*, Cham: Springer International Publishing, 2019, pp. 267–287, doi: 10.1007/978-3-030-15887-3_11.
- [19] J. S. Teh, K. Tan, and M. Alawida, "A chaos-based keyed hash function based on fixed point representation," *Cluster Computing*, vol. 22, no. 2, pp. 649–660, Jun. 2019, doi: 10.1007/s10586-018-2870-z.
- [20] P. Panagiotou, N. Sklavos, E. Darra, and I. D. Zaharakis, "Cryptographic system for data applications, in the context of internet of things," *Microprocessors and Microsystems*, vol. 72, pp. 1–12, Feb. 2020, doi: 10.1016/j.micpro.2019.102921.
- [21] A. Almarimi, A. Kumar, I. Almerhag, and N. Elzoghbi, "A new approach for data encryption using genetic algorithms," *Advances in Intelligent Systems and Computing*, vol. 167, pp. 783–791, 2012.
- [22] O. Banimelhem, L. Tawalbeh, M. Mowafi, and M. A. -Batafi, "A more secure image hiding scheme using pixel adjustment and genetic algorithm," *International Journal of Information Security and Privacy*, vol. 7, no. 3, pp. 1–15, Jul. 2013, doi: 10.4018/jisp.2013070101.
- [23] S. Dutta, T. Das, S. Jash, D. Patra, and P. Paul, "A cryptography algorithm using the operations of genetic algorithm & pseudo random sequence generating functions," *International Journal of Advances in Computer Science and Technology*, vol. 3, no. 5,




- pp. 325–330, 2014.
- [24] A. H. Alhussain, “Key exchange based on genetic algorithm,” *International Journal of Scientific Research in Science, Engineering and Technology*, vol. 1, no. 1, pp. 57–61, 2015.
 - [25] E. A. Albhrany, L. F. Jalil, and H. H. Saleh, “New text encryption algorithm based on block cipher and chaotic maps,” *International Journal of Scientific Research in Science, Engineering and Technology*, vol. 2, no. 2, pp. 67–73, 2016.
 - [26] P. Srikanth, A. Mehta, N. Yadav, S. Singh, and S. Singhal, “Encryption and decryption using genetic algorithm operations and pseudorandom number,” *IJCSN - International Journal of Computer Science and Network*, vol. 6, no. 3, pp. 455–459, 2017.
 - [27] P. A. N. Agbedemnab, E. Y. Baagyere, and M. I. Daabo, “A novel text encryption and decryption scheme using the genetic algorithm and residual numbers,” in *Proceedings of 4th International Conference on the Internet, Cyber Security and Information Systems 2019*, 2019, vol. 12, pp. 20–31, doi: 10.1109/AFRICON46755.2019.9133919.
 - [28] M. Gupta, K. K. Gupta, and P. K. Shukla, “Session key based fast, secure and lightweight image encryption algorithm,” *Multimedia Tools and Applications*, vol. 80, no. 7, pp. 10391–10416, Mar. 2021, doi: 10.1007/s11042-020-10116-z.
 - [29] R. Denis and P. Madhubala, “Hybrid data encryption model integrating multi-objective adaptive genetic algorithm for secure medical data communication over cloud-based healthcare systems,” *Multimedia Tools and Applications*, vol. 80, no. 14, pp. 21165–21202, Jun. 2021, doi: 10.1007/s11042-021-10723-4.

BIOGRAPHIES OF AUTHORS






Shihab A. Shawkat    received the B.Sc. degree in Computer Science from University of Tikrit in 2007 and M.Sc. Degree in Computer Science from Mansoura University in 2017. Currently a Ph.D student in National School of Electronics and Telecommunications of Sfax, University of Sfax, Sfax, Tunisia and. Mr. Shihab A. Shawkat worked as a teacher during the period from 2008 to 2019 in Directorate of Education in Salah Al-Din, Ministry of Education, Iraq. He has recently started working at the University of Samarra at the end of 2019 till now. His research interest lies in computer science, information security, image processing, and AI. He can be contacted at email: shahab84ahmed@gmail.com.



Najiba Tagougui    is actually Assistant Professor at the Higher Institute of Computer Sciences and Multimedia of Sfax, Sfax University Tunisia from where she graduated in Computer Sciences in 2005. She obtained a master degree in News technologies of dedicated computer systems in 2007 and a Ph.D in Computer Systems Engineering in 2014 at the National Engineering School of Sfax. Her research interest includes applications of intelligent methods to pattern recognition. She focuses her research on intelligent pattern recognition, especially online handwriting recognition and image caption generation. She can be contacted at email: najiba.tagougui@isims.usf.tn.



Monji Kherallah    received the Ing. Diploma degree, the Ph.D and HU in electrical engineering, respectively in 1989, 2008 and 2012, from University of Sfax (ENIS). For fourteen years ago, he was an engineer in Biotechnologie Center of Sfax. Now he is an associate professor in Faculty of Science of Sfax and member in Research Group of Intelligent Machines: REGIM-Lab. His research interest includes the handwritten documents analysis and recognition. The techniques used are based on intelligent methods, such as neural network, logic fuzzy, and genetic algorithm. He is one of the developers of the ADAB-Database (used by more than 50 research groups from more than 10 countries). He is co-organizer of the online Arabic handwriting recognition competitions at ICDAR 2009 and ICDAR 2011. He has more than 70 papers, including journal papers and book chapters. He is a member of IEEE and IEEE AESS Tunisia Chapter Chair, 2010 and 2011. He is reviewer of several international journals. He can be contacted at email: monji.kherallah@fss.usf.tn.