Proposed Encryption and Key Generation Method Based on Geffe Generator, Genetic Algorithm, and DNA Coding

Asst. Prof. Dr. soukaena hassan hashem

Computer sciences/university of technology: Soukaena.hassan@yahoo.com

Abstract

After the evolution in the digital world, it has become difficult to the preservation of information in the channels of communication; for this reason using of encryption methods is employed to provide security and to ensure access of information properly to the authorized persons and to ensure that the information is not manipulated and changed by others. Encryption methods are divided into two types one type is depended on using two keys (Public and Private key) while the other type is depended on using one secret key founded in both sides (Sender and Receiver). The encryption methods which depend on using two keys (Public and Private keys) sometimes are not secure because the predication of secret key is possible by using special calculations on the public key, the unauthorized person do this calculation .For this reason the methods which depend on private key are used where only authorized persons to send and receive the information have the private key. In this paper a proposed encryption method is presented based on using the Geffe Generator, Genetic algorithm, S-box and DNA coding. Since these will be used to generate special key. In the proposed method present three types of keys will be used in the encryption which are (Specified, Generated, and Static) which make it difficult to know the real key and the real secret text characters.

Keywords: Encryption, Geffe Generator, Genetic Algorithm, DNA coding.

طريقه تشفير وتوليد مفتاح مقترحه تعتمد على المولد geffe والخوارزميات الجينيه والترميز DNA

الخلاصة

بعد التطور الحاصل في العالم الرقمي اصبح من الصعب المحافظة على المعلومات في قنوات الاتصال لذلك يتم اللجوء الى استخدام طرق التشفير لتوفير الامنية لها وضمان وصول المعلومات بشكل سليم الى الاشخاص المخولين وضمان عدم التلاعب بها وتغييرها من قبل الاشخاص الغير مخولين.طرق التشفير تنقسم الى قسمين قسم يعتمد على مفتاحين (مفتاح عام ومفتاح خاص) وقسم يعتمد على مفتاح واحد سري يكون لدى كل من الطرفين(المرسل والمستلم).ان طرق التشفير التي تعتمد على مفاتحين مفتاح عام ومفتاح خاص بعض الاحيان تكون غير امنة وذلك لانه من الممكن التنبأ بالمفتاح الخاص عن طريق حسابات للمفتاح العام يقوم بها الاشخاص الغير مخولين.لذلك يفضل استخدام طرق المفتاح الخاص حيث ان المفتاح الخاص يوجد فقط لدى الاشخاص المخولين لارسال واستلام المعلومات. في هذا البحث تم اقتراح طريقة تشفير بالاعتماد على مولد ال Geffe والخوارزمية الجينية وال S-box وترميز ال DNA. وبما أن هذه سوف تستخدم لتوليد مفتاح خاص. في الطريقة المقترحة سيتم استخدام ثلاثة أنواع من المفاتيح في التشفير والتي هي (محددة، مولده، وساكنة) مما يجعل من الصعب معرفة المفتاح الحقيقي والحروف النص السري الحقيقي.

الكلمات المفتاحيه: *تشفير والمولد geffe والخوارزميات الجينيه و والترميز بال DNA.*

1. Introduction

The rapid growth of computer networks allows large files, such as text, voice and video, to be easily transmitted over the Internet, and it is important to protect confidential data from unauthorized access [1]. Information security is a very important topic in data transfer. Any loss or threat to the transfer of information can be a big loss in the process of sending information. Encryption technology plays a key role in information security systems [2]. Encryption has been used primarily to prevent the disclosure of confidential information, but it can also be used to provide credibility of the message, check the integrity of incoming data, provide a digital equivalent of handwritten signature, and nonrepudiation. Non-denial confirms that the party deals cannot deny that the transaction is taken place. Encryption is the name for the study of procedures and algorithms and methods to encrypt and decrypt information is encrypted, where, cryptanalysis is to study ways and means to defeat or encryption techniques and compromise. With some encryption methods, (and this paper used the same idea with the amendment to use the same key to encrypt and decrypt information is encrypted (multiple secret keys)), this type of encryption is known as symmetric encryption, which is also known as one of the key or secret encryption key format. Another encryption is that it uses two keys: one key to encrypt and decrypt a different key. These systems are referred to as non-symmetric encryption, also referred to as public-private encryption, which is also necessary because one key is known to all and the other is kept secret [3].

2. Linear Feedback Shift Register (LFSR)

Stream cyphers are type of Symmetric Key cryptosystems that encrypts plaintext one bit/byte at a time. Pseudorandom number sequences (PNS) are sequences whose properties approximate the properties of sequences of random numbers. These are not truly random, because it is completely determined by a relatively small set of initial values. LFSRs, see figure (1), are used in a stream cypher to generate linear sequences of pseudorandom numbers. They require very less hardware and have high speed of operations. An n-stage LFSR is of maximum length if initial states repeat after every $(2n - 1)$ bits. The contents of the registers are moved by one position at each clock. The left-most bit fed to the register is the result of mod-2 addition of bits corresponding to the nonzero coefficients of considered primitive polynomial. The right most bit is used to form the pseudorandom number sequence. All initial states should not be ‘‘0’’s because the LFSR would remain locked-up in these states [4].

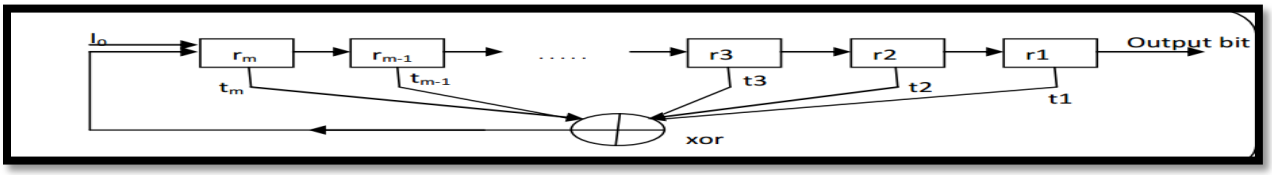


Figure (1): M-sequence Generation (LFSR) [5]

3. Geffe Generator

The Geffe generator [6] is defined by three maximum-length LFSRs whose lengths r1, r2, r3 are pair wise relatively prime, with nonlinear combining function, see equation (1).

$$F3(x1,x2,x3) = x1*x2 \oplus (1 \oplus x2)*x3 = x1*x2 \oplus x2*x3 \oplus x3 \ldots\ldots\ldots (1).$$

Figure (2) represents the Geffe generator.

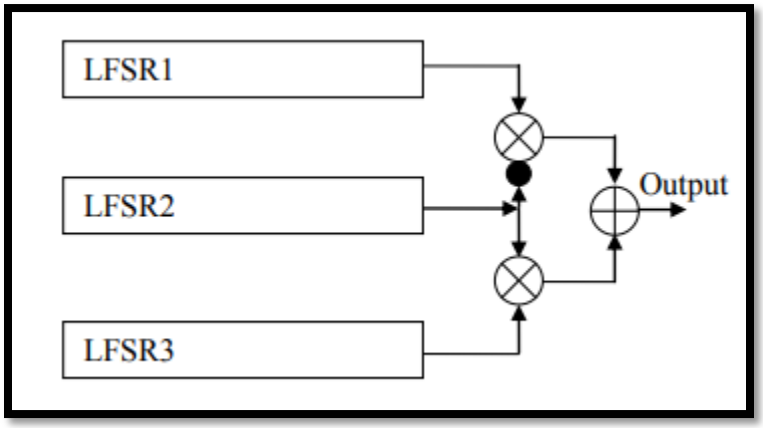


Figure (2): Geffe generator.

The keystream generated has period $(1^{r2} -1)(2^{r2} -1)(3^{r2} -1)$ and linear complexity, see equation (2).

$$LC=r1r2+r2r3+r3 \ldots\ldots (2).$$

The Geffe generator is cryptographically weak because information about the states of LFSR1 and LFSR3 leaks into the output sequence. Despite having high period and moderately high linear complexity, the Geffe generator succumbs to correlation attacks [6].

4. Genetic algorithm

Genetic Algorithm, similar to many other meta-heuristics, is an evolutionary population based algorithm. That is to say, a population of answers will evolve through the course of the optimization to move toward the optimality of a problem. Answers or individuals in GA are presented in chromosomes, which, incidentally, are the very strong suit of the algorithm. A chromosome is in fact an answer to the problem which is encoded to form a chromosome. The most prevalently applied chromosome is the binary chromosome. Each GA, consequently, needs to have a decoding function with the purpose of converting chromosome encoding to answers. Figure (3) presents a general flowchart of the algorithm.

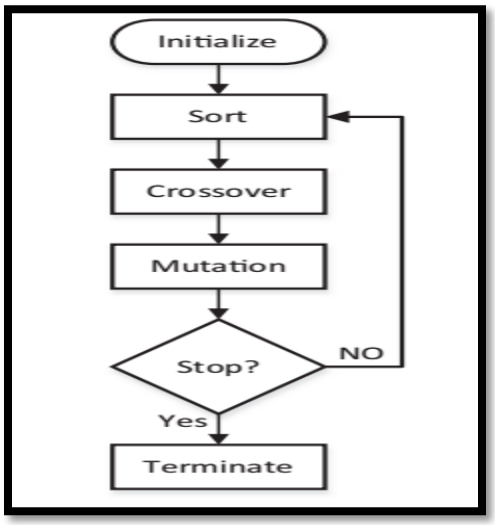


Figure (3): A general flowchart of the Genetic algorithm

GA will initialize by randomly producing chromosomes as many as the number of populations. In the case of binary chromosome, the cells of the chromosome will be filled with 0 or 1 by the same chance. Each and every chromosome will be decoded to an answer and consequently, their fitness value will be calculated. Fitness value, by definition, is the goodness of the answer according to the problem. Next, the population will be sorted based on the individuals' fitness value. Crossover and mutation are the two very important operators of the algorithm. In both, a selection function plays an essential role. Basically, the operators will change entering chromosomes with the hope of improving them, but choosing which chromosome to undergo the operations is by the selection function [7].

The function that is used for this purpose time and again is the roulette wheel function. This function operates in a way such that each and every member of the population has a chance to be selected, but the better the fitness value of a chromosome the more selection chance it will have. Crossover is a bi-chromosomal operator in the sense that it will work on two

chromosomes to output other(s). Two entering chromosomes will mix and produce one or two new chromosome(s) which are known by their offspring. In the case of the binary chromosome, one-cut crossover is most used. In one-cut crossover, both chromosomes will be broken from the same cell number and their parts will swap between the two, resulting in two different chromosomes that have characteristics of both entering chromosomes. Crossover is famous for being GA's optimality derive, swaying the population toward best answers. Mutation, unlike crossover, is not bi-chromosomal and does not serve the purpose of moving the population toward optimality. Its contribution to the algorithm is to keep it from local optima by radically changing the entering chromosomes. The single entering chromosome is changed by the operator harshly, without any reason, and randomly [7].

Last word about mutation is the extent that operator will change the entering chromosome. Mutation rate is the term for this behavioral factor of the algorithm. The pivotal step in the algorithm and certainly in the flowchart is deciding when to stop the evolution and be satisfied with the best answer in hand. There is no way GA can be sure of the optimal solution unless an optima is known to it in advance so there is a need for stopping strategies. In fact, there are different stoppage criteria. They can be as simple as a specific numbers of iterations or more involved by bringing the scaled improvements into equation [7].

5. DNA Computing

DNA is a polymer composed of monomers called nucleotides. Each nucleotide contains three components: a sugar, a phosphate group, and a base. Figure 1 illustrates the structure of a nucleotide. The sugar has five carbon atoms which are numbered from 1' to 5'. The Phosphate group is attached to the 5' carbon and the base is attached to the 1' carbon. There are four different types of bases: Adenine, Guanine, Cytosine and Thymine, abbreviated as A, G, C and T, respectively. What makes a nucleotide distinct from another is the base portion. So we can refer to every nucleotide as A, G, C or T nucleotides, depending on the type of base they have. Therefore it is possible to consider a DNA strand as a string over the alphabet {A, G, C, T}. For example ATTGCATGG is a DNA strand composed of 9 nucleotides. In every DNA computing experiment, there is a (test) tube containing lots of DNA molecules (strands). Each of the DNA molecules can be a potential solution to the problem. Also there are some biological operations available, which are performed on the DNA strands of the tube to perform computations. These operations differ according to the DNA computing model used [8].

6. Substitution Box (S-Box)

Substitution-boxes, or simply S-boxes, are used to increase confidentiality in substitution stage of most of cryptosystem approaches. The design of substitution box, or simply S-box, for secure cryptographic ciphers attracts a great deal of attention of most cryptographic researchers. Indeed, S-box is an important component of most

block ciphers and the good S-box ensures the nonlinearity and the confusion property [9]. In this paper S-box designing based on DNA codes is introduced in order to make the proposed text encryption method more secure and more efficient.

7. The Proposed Method

The encryption methods which depend on one private key to protect the information are secure but if the secret key is known by unauthorized persons this security is disappeared. In the proposed method the problem on depending on one secret key is solved by depending on multi secret keys.

In the proposed method multi secret keys are used in order to provide more security and to reduce the secret keys predication capability by the unauthorized persons. The security is achieved because every secret key is designed in special way and has special characters. Some of these keys are random based on some methods like Geffe Generator, Genetic algorithm to generate other keys and the other keys are static (tables) are found at both sides (Sending side and the Receiver side). At the beginning 3 specified seeds are used in the Geffe generator to output a key it's length is (16 bits) for a simple example, these 16 bits will be divided into two parts each part is 8 bits in order to be considered as initial population to the Genetic algorithm, then specified Genetic algorithm phases will be applied to generate a new population (children). After that the genetic algorithm result will be stored.

Then the secret text characters will be read. Then the secret text characters will be converted to numbers in the form (n1, n2) using table (1) where n1 represents the row and the n2 represents the column. Then the numbers will be converted to binary representation. Apply merging to every two 3 bits to get 6 bits with adding specified padding (2 bits). Then the binary representation will be converted to DNA codes using table (2). Apply swapping with DNA s-box codes using table (3).Convert DNA codes to binary representation using table (4). Apply XOR with first stored Genetic algorithm children. The XOR will be applied between the previous result and the second stored Genetic algorithm children. Finally the result will be send. Algorithm (1) and algorithm (2) and the figures (4, 5) illustrate the proposed method at the both sides.

Table (1) secret text characters will be converted to numbers in the form (n1, n2)

	0	1	2	3	4	5
0	a	b	C	d	e	f
1	g	h	I	j	k	l
2	m	n	O	p	q	r
3	s	t	U	v	w	x
4	y	z	0	1	2	3
5	4	5	6	7	8	9

Table (2) the binary representation will be converted to DNA codes

Binary Codes (2 bits)	DNA Symbols
00	**A**
01	**C**
10	**G**
11	**T**

Table (3) swapping with DNA s-box codes; take the intersection value of column and row of each one code

Column / Row	A	C	G	T
A	A	C	G	T
C	C	G	T	A
G	G	T	A	C
T	T	A	C	G

Table (4) Convert DNA codes to binary representation

DNA Symbols	Binary Codes (2 bits)
A	11
C	10
G	01
T	00

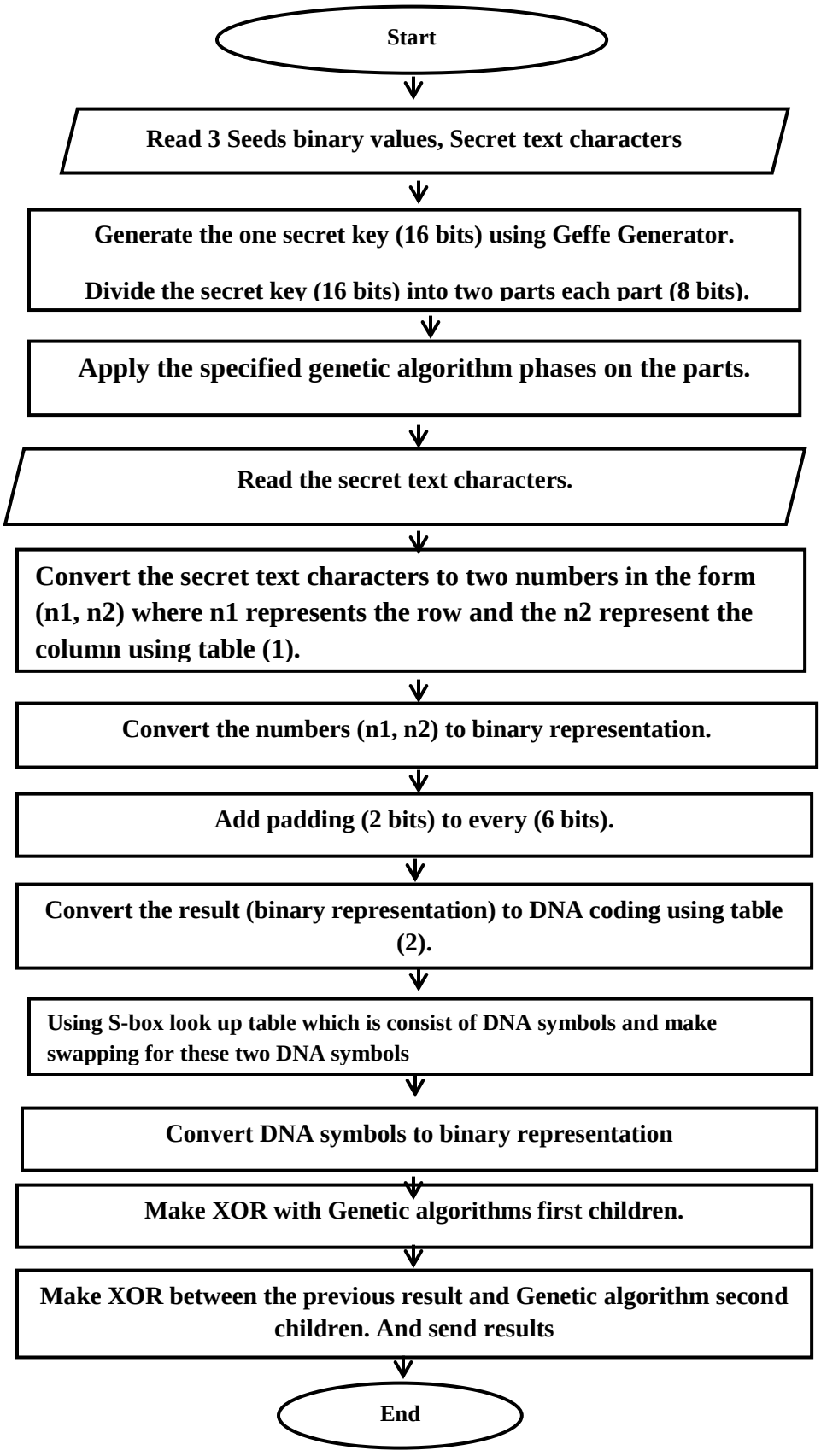


Figure (4): The proposed method at the Sender

Algorithm (1) the proposed method at the sender side

Input: Three seeds binary values, Secret text characters, the four tables.

Output: Binary representation.

Step1:- Begin

Step2:- Using the 3 specified seeds to generate the one secret key (16 bits for a simple example) using Geffe Generator.

Step3:- Divide the secret key (16 bits) into two parts each part (8 bits).

Step4:- Consider the keys as chromosomes in order to apply the specified genetic algorithm phases.

Step5:- Read the secret text characters.

Step6:- Convert the secret text characters to two numbers in the form (n1, n2) where n1 represents the row and the n2 represent the column using table (1).

Step7:- Convert the numbers (n1, n2) to binary representation.

Step8:- Add padding (2 bits) to every (6 bits).

Step9:- Convert step7 result (binary representation) to DNA coding using table (2).

Step10:- Using S-box look up table which is consist of DNA symbols and make swapping for these two DNA symbols where the first DNA symbol is used as determination to the row of the S-box lookup table and the other DNA symbol is used as determination to the column of the S-box look up table. The row determination DNA symbol is stay and the corresponding cell of that row is changed with the column determination using table (3).

Step11:- Convert DNA symbols to binary representation using table (4).

Step12:- Make XOR between Genetic algorithms first children.

Step13:- Make XOR between step9 result and Genetic algorithm second children.

Step14:- Send the step12 result.

Step15:- End.

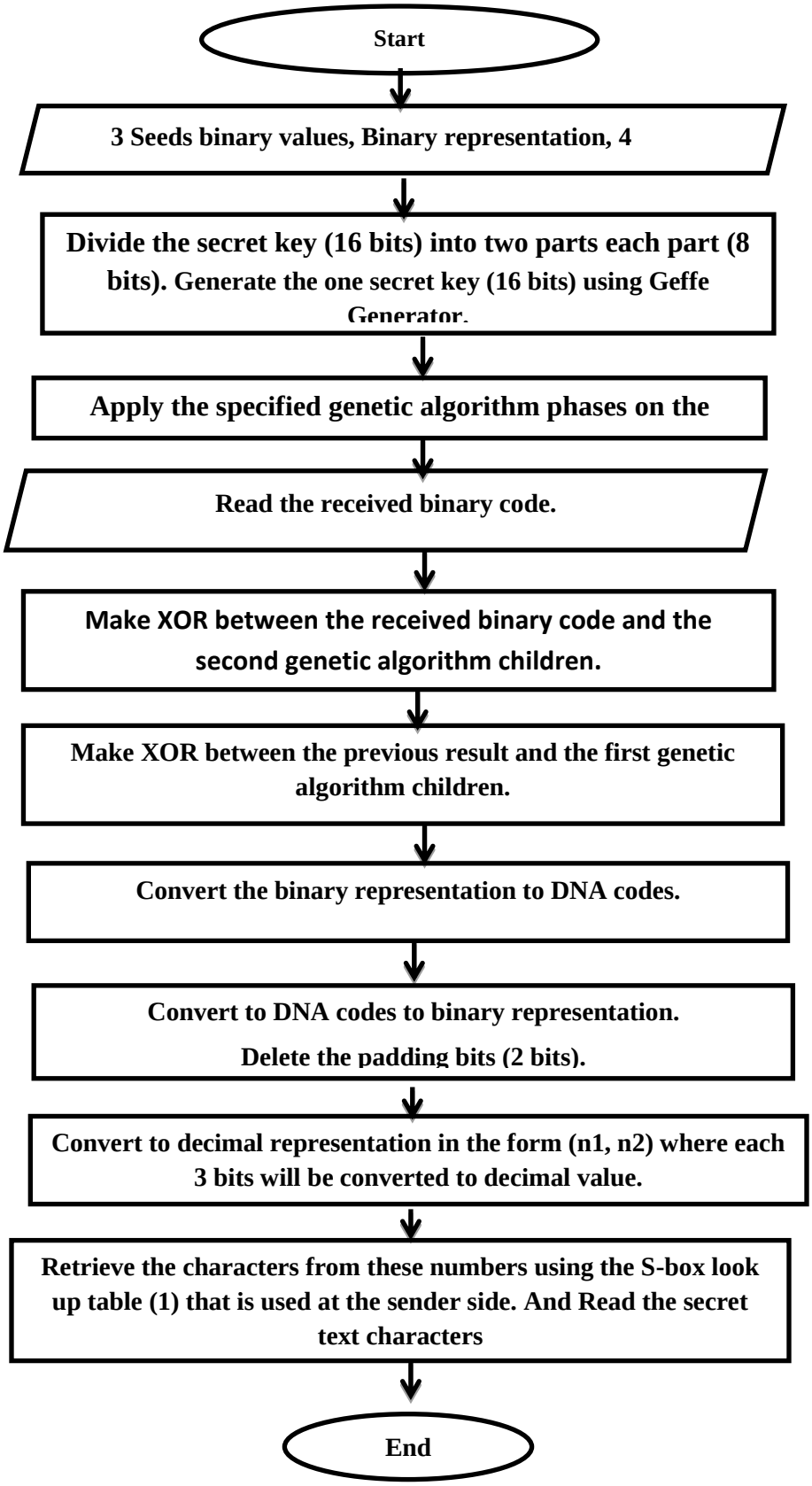


Figure (5): The proposed method at the Receiver Side

Algorithm (2) the proposed method at the receiver side
Input: Three seeds binary values, Binary representation, the four tables.
Output: Secret text characters.
Step1:- Begin Step2:- Using the 3 specified seeds to generate the one secret key (16 bits for a simple example) using Geffe Generator. Step3:- Divide the secret key (16 bits) into two parts each part (8 bits). Step4:- Consider the keys as chromosomes in order to apply the specified genetic algorithm phases. Step5:- Read the received binary code. Step6:- Make XOR between the received binary code and the second genetic algorithm children. Step7:- Make XOR between the step6 result and the first genetic algorithm children. Step8:- Convert the binary representation to DNA codes using table (4). Step8:- Making inverse S-box converting where each two DNA symbols will be swapped .The first DNA symbol will be as indication to the row and the second DNA symbols as indication to cell content to retrieve the corresponding column DNA symbol with the row DNA symbol using table (3). Step9:- Convert to DNA codes to binary representation using table (2). Step10:- Delete the padding bits (2 bits). Step11:- Convert to decimal representation in the form (n1, n2) where each 3 bits will be converted to decimal value. Step12:- Retrieve the characters from these numbers using the S-box look up table (1) that is used at the sender side. Step13:- Read the secret text characters. Step14:- End.

7.1 Example of the proposed method (16-bit for simple example)

At the beginning suppose that the three secret seeds for the Geffe generator are as following

Seed 1 *65=001000001*

Seed 2 *70=001000110*

Seed 3 *135=110000111*

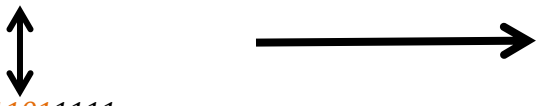
After applying the Geffe Generator the result will be as following (1000001111011111), These 16 bits will be divided into two parts each part is (8 bits).

Part1 =10000011

Part2=11011111

These two parts will be considered as the initial population to the genetic algorithm. In the example we suppose that the specified genetic algorithm phases are just the crossover to generate the new children. The crossover will be applied at the middle of each part, after the crossover the result will be as following:

*Part1 =1000*0011 *Part1 =11010011*



*Part2 =1101*1111 *Part2=10001111*

Then the secret text characters will be read, we suppose that the secret text is “cryptography". In order to be converted to numbers in the form (n1,n2) table (1) is used the output will be as following:

c=(0,2),r=(2,5),y=(4,0),p=(2,3),t=(3,1),o=(2,2),g=(1,0),r=(2,5),a=(0,0),p=(2,3),h=(1,1),y=(4,0).

Then the numbers will be converted to binary representation as following:

c=(000,010),r=(010,101),y=(100,000),p(010,011),t=(011,001),o=(010,010),g=(001,000),r=(010, 101),a=(000,000),p=(010,011),h=(001,001),y=(100,000).

Apply merging to every two 3 bits to get 6 bits with adding padding (2 bits) in order to get 8 bits which is the same length of the genetic algorithm stored keys key. Merging is:

000010 010101 100000 010011 011001 010010 001000 010101 000000 010011 001001 100000.

After Padding:

00000100 00101010 01000000 00100110 00110010 00100100 00010000

00101010 00000000 00100110 00010010 01000000.

Padding bits

Then Convert to DNA representing using table (2) ,the result will be as following

00000100 00101010 01000000 00100110 00110010 00100100

AACA AGGG CAAA AGCG ATAG AGCA

00010000 00101010 00000000 00100110 00010010 01000000.

ACAA AGGG AAAA AGCG ACAG CAAA

Apply swapping with DNA S-box look up table using table (3) ,the result will be as following.

AACA AGGA CCAA AGCT ATAG AGCC ACAA AGGA AAAA AGCT ACAG CCAA

Then these DNA codes will be converted to binary representation using table (4), the result will be as following:

1111101011010111101011111101100011001101110110101110111111010111111111111011000111010110101111

Binary codes =

11111010 11010111 10101111 11011000 11001101 11011010 11101111 11010111 11111111 11011000 11101101 10101111

Stored key=

11010011 11010011 11010011 11010011 11010011 11010011 11010011 11010011 11010011 11010011 11010011

First XORED result is =

00101001 00000100 01111100 00001011 00011110 00001001 00111100
00000100 00101100 00001011 00111110 01111100

Then the XOR result will be XORed with the second stored genetic algorithm children and the result will be as follow:

First XORED result is =

00101001 00000100 01111100 00001011 00011110 00001001 00111100
00000100 00101100 00001011 00111110 01111100

Second stored key =

10001111 10001111 10001111 10001111 10001111 10001111 10001111 10001111
10001111 10001111 10001111 10001111

The final result after XOR is =

10100110 10001011 11110011 10000100 10010001 10000110 10110011
10001011 10100011 10000100 10110001 11110011

This final result will be send.

7.2 Evaluation of the proposed method

In the proposed method there are three types of secret keys:

1) The specified keys (3 Primitive polynomials, 3 seeds which are used in Geffe generator and padding which can be 00, 01, 10, 11).
2) The Generated keys (Geffe Generator result) and the Genetic algorithm result.
3) The Static keys (the tables which are used like table, S-box look up table).

In the following table, table (5) the tested results are display, with randomness measures that are used in cryptography which are (Frequency test, Serial test, Poker test, Run test, Auto_Correlation (AC) test).

Table (5) randomness test of the three keys

Example No.	**LFSR polynomial**	**Seeds**	**Geffe Generator** **Result**	**Specified Keys Randomness Measures**	**Generated Keys**	**Static Keys**
Example 1	LFSR1=X^8+ X^6+X^5+X^4+ 1 LFSR2=X^8+ X^6+X^5+X^4+ 1 LFSR3=X^8+ X^6+X^5+X^4+ 1	Seed 1 =001000 001 Seed 2 =001000 110 Seed 3 =110000 111	10000011110 11111	Frequency Test=9.000 Serial Test=12.000 Poker Test= 15.200 Run Test=38.969 AC Test Move No. 1 to No. 10 all are Succeed Value average from 3.769 to 0.500.	Frequency Test=1.000 Serial Test= 3.000 Poker Test=7.200 Run Test= 8.375 AC Test Move No. 1 to No. 10 all are Succeed Value average from 3.267 to 0.000	Frequency Test= 18.375 Serial Test= 19.500 Poker Test= 21.367 Run Test= 8.583 AC Test Move No. to No. 10 most of them defeat Value from 6.868 to 7.511
Example 2	LFSR1=X^{16} +X^{14}+X^{13}+X^{11}+1 LFSR2=X^{19} +X^{18}+X^{17}+X^{14}+1 LFSR3=X^9+ X^5+1	Seed1= 0010101 0101010 101 Seed2= 1111000 0111100 001111 Seed3= 0100110 011	10010110111 10101110110 1001001100	Frequency Test= 0.125 Serial Test= 12.500 Poker Test= 5.400 Run Test= Succeed Value T0 = 8.250 AC Test AC Test Move No. 1 to No. 10 most of them succeed Value 0.043 and some defeat Value 4.172	Frequency Test= 0.500 Serial Test= Succeed Value T0 = 3.000 Poker Test= Succeed Value 2.400 Run Test= Succeed Value T0 = 3.000 AC Test Move No. 1 to No. 10 all are Succeed Value from 2.793 to 0.000	Frequency Test= 18.375 Serial Test= 19.500 Poker Test= 21.367 Run Test= 8.583 AC Test Move No. 1 to No. 10 some of them succeed Value 0.011 and some defeat Value 7.511

8. Conclusions

A deception and secure encryption method is proposed in this paper. The proposed method is better in security level since it depends on multi secret special keys and these keys are designed based on Geffe Generator, DNA, and S-box .The proposed method is fast and simple in implementation since it doesn't need any complex calculation. The private keys are kept secret since the secret keys are not used directly in encryption but will be processed using genetic algorithm. The attacker can't know the secret keys since not the original secret keys will be used to make the encryption.

References

[1] Ghassan M.H. ," Image Encryption Using Permutation and Hill Cipher ", Al-Rafidain University College For Sciences ,2012.

[2] Hasan M. Azzawi ," Enhancing The Encryption Process Of Advanced Encryption Standard (AES) By Using Proposed Algorithm To Generate S-Box", Journal of Engineering and Development, Vol. 18, No.2, March 2014, ISSN 1813- 7822.

[3] Qusay M. Jafar Alsadi ," Proposed Method for Text Encryption Using Two Secret Keys and One Secret Mathematical Equation ",Al-Rafidain University College For Sciences,2009.

[4] Maiya Dina, Saibal K. Pal a S.K. Muttoo b, Anjali Jainc , "Applying Cuckoo Search for analysis of LFSR based cryptosystem", Perspectives in Science (2016) 8, 435—439.

[5] Noor Dhia Kadhm Al-Shakarchy ," Randomly Steganography using LFSR and NLFSR generation", Journal of KerbalaUniversity , Vol. 11 No.1 Scientific . 2013.

[6] Hussein Ali Mohammed , "Frequency Postulate's Theoretical Calculation for the Sequences Produced by Modified Geffe Generator", Journal of Kerbala University , Vol. 12 No.2 Scientific ,2014.

[7] Ruholla Jafari-Marandi, Brian K. Smith "Fluid Genetic Algorithm (FGA)" , Journal of Computational Design and Engineering 4 (2017) 158–167.

[8] Ramin Maazallahi, Aliakbar Niknafs, Paria Arabkhedri , "A Polynomial-Time DNA Computing Solution for the N-Queens Problem" , Procedia - Social and Behavioral Sciences 83 (2013) 622 – 628.

[9] Akram Belazi , Ahmed A. Abd El-Latif b," A simple yet efficient S-box method based chaotic sine map" Optik - International Journal for Light and Electron Optics Volume 130, February 2017, Pages 1438-1444.