A Cryptanalytic Attack on Vigenère Cipher Using Genetic Algorithm

S. S. Omran
College of Elec. & Electronic Techniques
Foundation of Technical Education
Safaa_Omran@yahoo.com

A. S. Al-Khalid
College of Elec. & Electronic Techniques
Foundation of Technical Education
mudariben@yahoo.com

D. M. Al-Saady
Foundation of Technical Education
alsaady_dalal@yahoo.com

ABSTRACT**-** With the exponential growth of networked system and its applications such as e-commerce, the demand for effective internet security is increasing. Cryptology is the science and study of systems for secret communication. It consists of two complementary fields of study: cryptography and cryptanalysis.

In this paper the cryptanalysis of a poly alphabetic substitution cipher (Vigenère cipher) by applying genetic algorithms is presented. The applicability of genetic algorithms for searching the key space of encryption scheme is studied.

The frequency analysis is used as an essential factor in the objective function.

Keywords: Genetic algorithms, Vigenère cipher, Cryptanalysis, key space.

I: INTRODUCTION

The demand for effective internet security is increasing exponentially day by day. Businesses have an obligation to protect sensitive data from loss or theft. Such sensitive data can be potentially damaging if it is altered, destroyed, or if it falls into the wrong hands. So they need to develop a scheme that guarantees to protect the information from the attacker. Cryptology is at the heart of providing such guarantee.

Cryptology is the science of building and analyzing different encryption and decryption methods. Cryptology consists of two subfields; Cryptography & Cryptanalysis. Cryptography is the science of building new powerful and efficient encryption and decryption methods. It deals with the techniques for conveying information securely. The basic aim of cryptography is to allow the intended recipients of a message to receive the message properly while preventing eavesdroppers from understanding the message. Cryptanalysis is the science and study of method of breaking cryptographic techniques i.e. ciphers. In other word it can be described as the process of searching for flaws or oversights in the design of ciphers [1-3].

Many researches in the field of cryptanalysis are interested in developing automated attacks on encryption algorithms (ciphers). When analyzing ciphers it is advantageous that a proposed attack will run without human intervention, finishing either when the message has been successfully decrypted or the key has been determined[4]. Previous work in this area[5-7] has shown that genetic algorithms can provide successful automated attacks on poly alphabetic substitution cipher. The purpose of this paper is to Cryptanalyse a poly substitution cipher and guesses the key size. One of these types is the Vigenère cipher. The Vigenère ciphers is consisted of finding the keyword and then dividing the message into many simple substitution cryptograms[5,8-11].

In 1994 Clark for the first time presented a genetic algorithm approach for the cryptanalysis of poly substitution ciphers using genetic algorithm [4]. He has explored the possibility of random type search to discover the key for a poly substitution cipher.

In this paper, a certain method has been used to find the length of the key, and by using genetic algorithms finding the key itself and to break the poly substitution cipher i.e. Vigenère cipher.

II: VIGENERE CIPHER

The Vigenère code is a method of encoding text by replacing each character of the plain-text with another letter determined by adding of the plain text character to the index number of an arbitrarily chosen code word. The encoding is often done using a table of rows of the alphabet, shifted according to the indices of the letters in the code word.

Table (1) shows the assigned number for each letter of the alphabet.

Table 1: Mapping letters to integers and back

a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12
n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

For example, to encode the sentence **"example message"** using the code word **"test"**, each letter of the plain text and of the code word is represented by its index number. The code word indices are added to the text indices mod 26, as shown in equation (1) [5, 8, 10] and Fig.1 [12-15].

$$E(k,m) = m + k \text{ (mode 26)} \qquad \text{.......(1)}$$

978-1-61284-931-7/11/$26.00 ©2011 IEEE

Also The intersection position can been used to encrypt the letters as shown in Table (2) [12, 16-18].

The cryptanalyst uses the fact that in most English texts the frequencies of letters are not equal. For example, e letter occurs much more frequent than x. These frequencies have been tabulated in Fig.2 [2,5,8-10].

Plain text

e	x	a	m	p	l	e	m	e	s	s	a	g	e
4	23	0	12	15	11	4	12	4	15	15	0	6	4

Code word

T	E	S	T
19	4	18	19

key

T	E	S	T	T	E	S	T	T	E	S	T	T	E
19	4	18	19	19	4	18	19	19	4	18	19	19	4

Cipher text

23	1	18	5	8	15	22	5	23	22	10	19	25	8
X	B	S	F	I	P	W	F	X	W	K	T	Z	I

Fig.1 Vigenère cipher encryption example

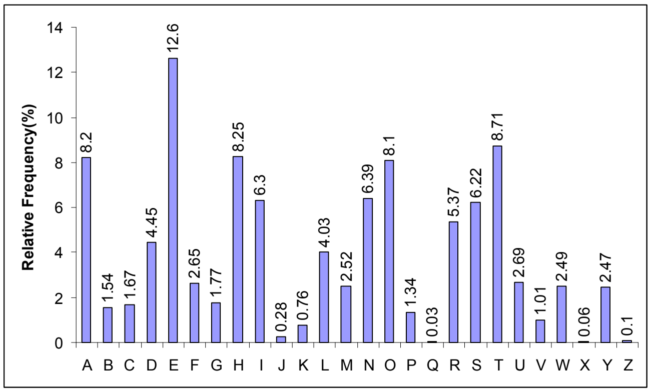


Fig. 2 Relative frequency of letters in English language

III. GENETIC ALGORITHMS

The genetic algorithm is based upon Darwinian evolution theory. The genetic algorithm is modeled on a relatively simple interpretation of the evolutionary process however, it has proven to be a reliable and powerful optimization technique in a wide variety of applications. Holland [19] in 1975, proposed the use of genetic algorithms for problem solving. Goldberg [20] was also pioneers in the area of applying genetic processes to optimization. Over the past twenty years numerous applications and adaptations of genetic algorithms have appeared in the literature.

During each iteration of the algorithm, the processes of selection, reproduction and mutation each take place in order to produce the next generation of solution.

Genetic Algorithm begins with a randomly selected population of chromosomes represented by strings. The GA uses the current population of strings to create a new population such that the strings in the new generation are on average better than those in current population (the selection depends on their fitness value).

Table 2: Vigenère tableau

	A B C D E F G H I J K L M N O P Q R S T U V W XYZ
A	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
B	B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
C	C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
D	D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
E	E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
F	F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
G	G H I J K L M N O P Q R S T U V W X Y Z A B C D E F
H	H I J K L M N O P Q R S T U V W X Y Z A B C D E F G
I	I J K L M N O P Q R S T U V W X Y Z A B C D E F G H
J	J K L M N O P Q R S T U V W X Y Z A B C D E F G H I
K	K L M N O P Q R S T U V W X Y Z A B C D E F G H I J
L	L M N O P Q R S T U V W X Y Z A B C D E F G H I J K
M	M N O P Q R S T U V W X Y Z A B C D E F G H I J K L
N	N O P Q R S T U V W X Y Z A B C D E F G H I J K L M
O	O P Q R S T U V W X Y Z A B C D E F G H I J K L M N
P	P Q R S T U V W X Y Z A B C D E F G H I J K L M N O
Q	Q R S T U V W X Y Z A B C D E F G H I J K L M N O P
R	R S T U V W X Y Z A B C D E F G H I J K L M N O P Q
S	S T U V W X Y Z A B C D E F G H I J K L M N O P Q R
T	T U V W X Y Z A B C D E F G H I J K L M N O P Q R S
U	U V W X Y Z A B C D E F G H I J K L M N O P Q R S T
V	V W X Y Z A B C D E F G H I J K L M N O P Q R S T U
W	W X Y Z A B C D E F G H I J K L M N O P Q R S T U V
X	X Y Z A B C D E F G H I J K L M N O P Q R S T U V W
Y	Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
Z	Z A B C D E F G H I J K L M N O P Q R S T U V W X Y

Three processes which have a parallel in biological genetics are used to make the transition from one population to the next (selection, crossover, and mutation) as shown in Fig.3

The selection process determines which string in the current will be used to create the next generation. The crossover process determines the actual form of the string in the next generation. Here two of the selected parents are paired. A fixed small mutation probability is set at the start of the algorithm [7].

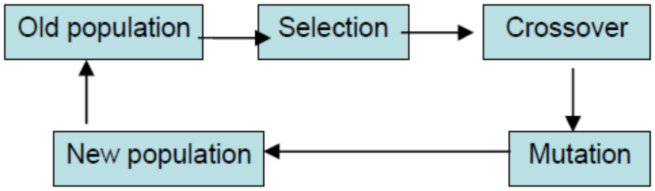


Fig. 3 The basic genetic algorithm cycle

IV. FITNESS MEASURE

The technique used to compare candidate keys is to compare n-gram statistics of the decrypted message with those of the language (which are assumed known).

Equation (2) is a general formula used to determine the suitability of a proposed key (k)[5,6, 21].

$$C_k \approx \alpha . \sum_{i\in A} \left|K^{u}_{(i)} - D^{u}_{(i)}\right| + \beta \sum_{i,j\in A} \left| K^{b}_{(i,j)} - D^{b}_{(i,j)} \right| + \gamma \sum_{i,j,k\in A} \left|K^{t}_{i,j,k} - D^{t}_{i,j,k}\right| \quad ...(2)$$

Here, *A* denotes the language alphabet i.e., for English, [A. . . Z], K and D denote known language statistics and decrypted message statistics, respectively, and the indices u, b and t denote the unigram, bigram and trigram statistics, respectively. The values of α, β and γ allow assigning different weights to each of the three n-gram types, in which case that (α+β+γ=1). In our work we put β and γ to zero and α =1, so we used the frequency of unigram only.

V. FINDING THE KEY LENGTH

To find the key length, a certain method is used, as following[2]:

- Writing the cipher text as a row on a long strip of paper, and again on another long strip.
- Displacing the cipher text by a certain number of places.
- Corresponding to the original cipher text with the displaced cipher text in each displacement and counting the total number of coincidence.
- The best guess for the key length equals to the number of places that gives the most corresponding numbers of coincidences.

VI. PROPOSED ALGORITM

The following is an outline of the proposed algorithm [5]:

1. Input to the algorithm the cipher text, the key size and the relative character frequencies.
2. Initialize the algorithm parameters: maximum number of iterations (M).
3. Generate randomly a population of 20 keys each one having the same known key length.
4. Decrypt the cipher text by the 20 generated keys.
5. Calculate the fitness of each key from every decrypted text using the formula shown in equation 1.
6. Sort the keys based on the increased fitness values.
7. for 1 to (M) do the followings:
   a. Choose 10 pairs from the 20 keys by using roulette wheel selection method.
   b. For 1 to (10 pairs) do
      i. Apply crossover to get children
      ii. Generate random number from 2 to (key size -1)

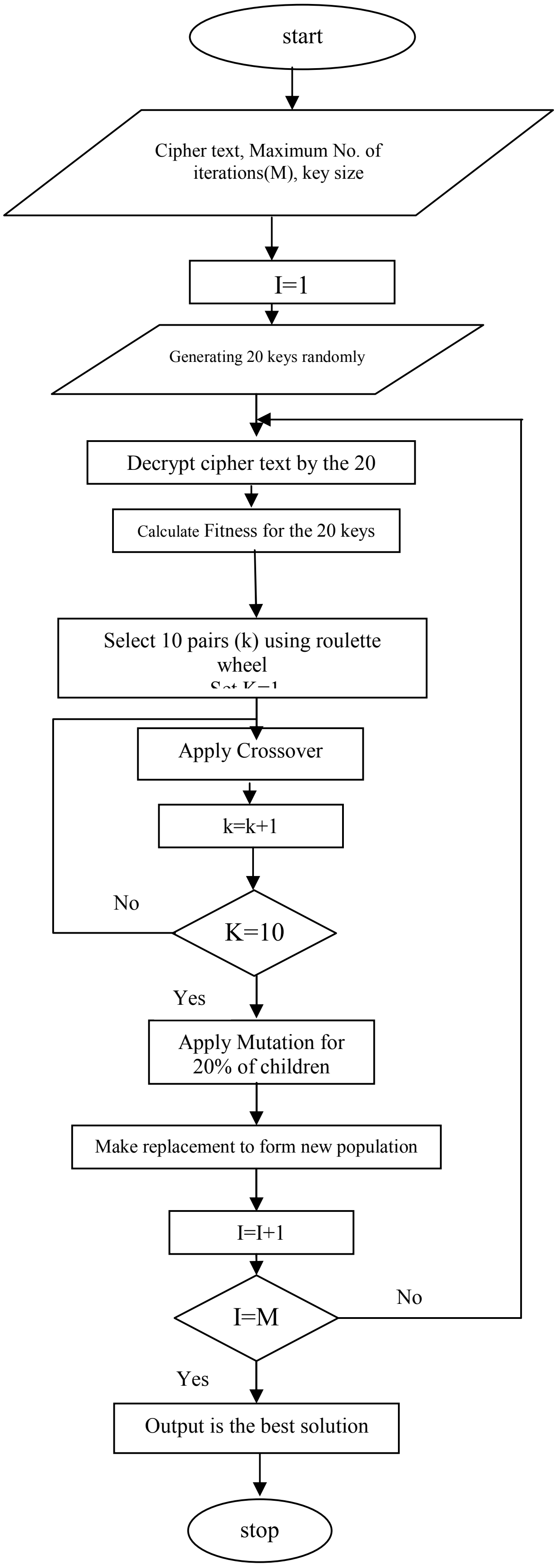

start
Cipher text, Maximum No. of iterations(M), key size
I=1
Generating 20 keys randomly
Decrypt cipher text by the 20
Calculate Fitness for the 20 keys
Select 10 pairs (k) using roulette wheel
Set K=1
Apply Crossover
k=k+1
No
K=10
Yes
Apply Mutation for 20% of children
Make replacement to form new population
I=I+1
No
I=M
Yes
Output is the best solution
stop

Fig.4 Flow chart of the proposed algorithm

iii. Swap the parts of parents.

c. Select randomly 20% from the children for mutation process.

d. Generate random position number between (1 to key size) and mutate the letter in that position.

e. Calculate the fitness for the new 20 children.

f. Sort the 40 keys based on increased fitness values (20 parents and 20 children).

g. Choose best 20 keys (new population).

8. Output is the best solution.

Figure 4 shows the flow chart of proposed algorithm.

VII. IMPLEMENTING THE ATTACK FOR VIGENERE CIPHER BY USING GENETIC ALGORITHMS

The proposed approach that is based on genetic algorithms for breaking the Vigenère cipher is now stated as follows: Generating 20 independent keys to represent the population size. The first generation is generated randomly using random number generator. Then the cipher text is decrypted using each permutation as a key, enabling us to assign a measure of fitness by using equation (1) to each candidate key. Pairs of candidate keys are then selected for producing offspring after applying a method of crossover to each pair.

The Roulette wheel selection is used in this paper. Each member of the population is allocated a section of an imaginary roulette wheel. Unlike a real roulette wheel the sections are of different sizes, proportional to the individual's fitness, such that the fittest candidate has the biggest slice of the wheel and the weakest candidate has the smallest. The wheel is then spun and the individual associated with the winning section is selected. The wheel is spun as many times as is necessary to select the full set of parents for the next generation [22].

Using this technique it is possible that one or more individuals is selected multiple times[20, 21]**.** Fig. 5 shows the Roulette wheel selection [22-25].

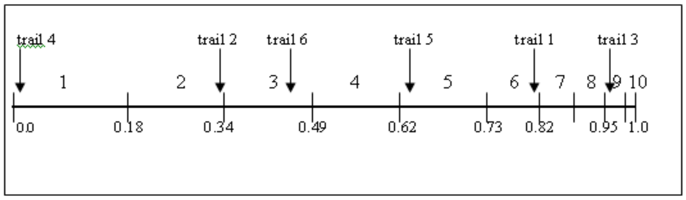
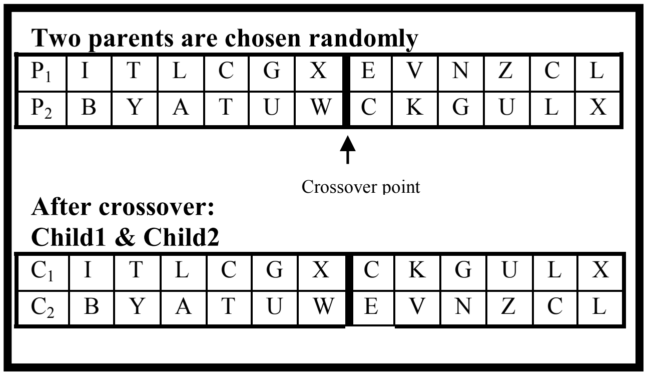


Fig.5 Roulette wheel selection

The crossover method used in this paper is a single point crossover, where the two mating chromosomes are cut once at corresponding points and the sections after the cuts are exchanged. Here, a cross-site or crossover point is selected randomly along the length of the mated string and letters next to the cross-site are exchanged. If appropriate site is chosen, better children can be obtained by combining good parents else it severely hampers string quality.

Fig. 6 shows single point crossover for key length of 12 letters. The crossover point can be chosen randomly[5, 23].



Two parents are chosen randomly

P_1	I	T	L	C	G	X	E	V	N	Z	C	L
P_2	B	Y	A	T	U	W	C	K	G	U	L	X

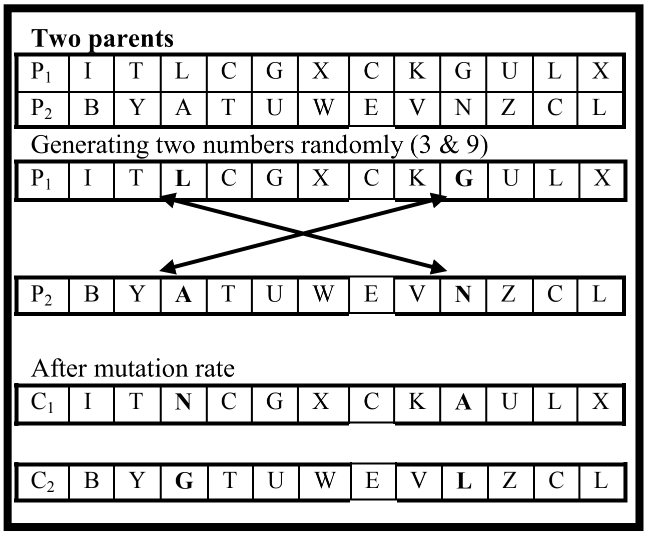
After crossover:
Child1 & Child2

C_1	I	T	L	C	G	X	C	K	G	U	L	X
C_2	B	Y	A	T	U	W	E	V	N	Z	C	L

Fig.6 Applying crossover between two parents

After crossover, some keys are subjected to mutation. Mutation prevents the algorithm to be trapped in a local minimum.

Two numbers are generated randomly. The range of numbers is between (1 to the key size). The character in position 3 in P_1 will be moved to position 9 in P_2 and vice versa. Fig. 7 shows the mutation process[1,5,22].



Two parents

P_1	I	T	L	C	G	X	C	K	G	U	L	X
P_2	B	Y	A	T	U	W	E	V	N	Z	C	L

Generating two numbers randomly (3 & 9)

P_1	I	T	**L**	C	G	X	C	K	**G**	U	L	X
P_2	B	Y	**A**	T	U	W	E	V	**N**	Z	C	L

After mutation rate

C_1	I	T	**N**	C	G	X	C	K	**A**	U	L	X
C_2	B	Y	**G**	T	U	W	E	V	**L**	Z	C	L

Fig.7 Mutation process

VIII. RESULTS

In this paper different mutation rates have been used for key size 8 and 12 letters and for population size of 20. Fig.8 shows that the number of correct letters for mutation rate of 0.2 gives best results.

It is clear that we got 9 correct letters of the 12 letter key size and 6 correct letters of the 8 letter key size.

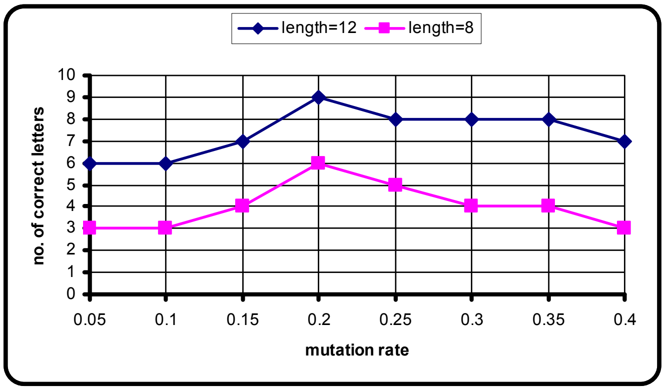


Fig.8 Effect of mutation rates

Then the algorithm was run for mutation rate 0.2 and population size 20 for different lengths of key and for different number of generations.

It is clear from Fig.9 that the number of correct letters for the used keys increases as the number of generation was increased, but still we could not get the whole correct letters of the used keys.

Table (3) shows the number of correct letters and fitness values for different lengths of key after 100 generation.

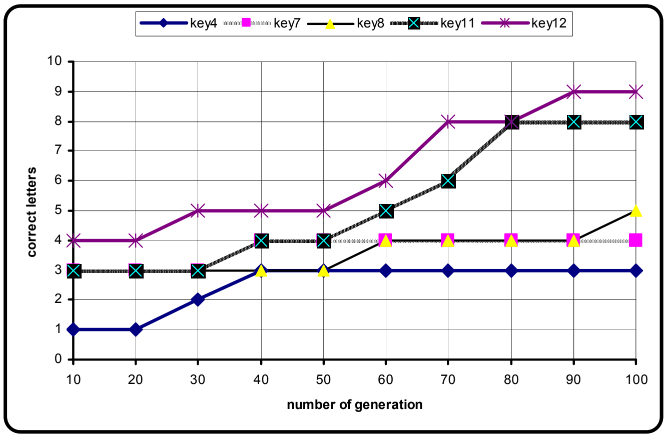


Fig.9 Different number of generation

Table 3: Different lengths of key for population size 20

Length of key	The Key	Fitness value	Correct letters
4	COST	77.4536	3
4	TEST	53.652	2
7	COLLEGE	71.3576	4
8	COMPUTER	76.3951	6
10	SIMULATION	72.8428	6
10	DEPARTMENT	79.3976	7
11	DEVELOPMENT	77.7785	7
12	SUBSTITUTION	82.5092	9

Then the algorithm was run for different number of population rate 0.2 and for 50 generation. Fig.10 shows that the whole correct letters was obtained for population size (60-100). Table (4) shows the fitness values for different lengths of key.

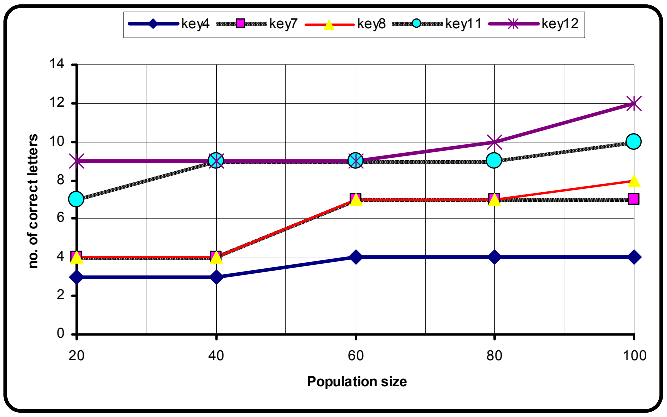


Fig.10 Different population size and correct letters for mutation rate 0f 0.2 and 50 generation

Table 4: Different lengths of key for population size 100

Length of key	Fitness value	Correct letters
7	90.9995	7
4	90.9995	4
8	90.9995	8
12	90.9995	12
11	89	10

IX. CONCLUSIONS

In this paper a genetic algorithm attack on a simple cryptographic cipher, called poly-alphabetic substitution, was implemented successfully.

The algorithm was implemented using the MATLAB program. Different parameters were tested such as the mutation rates, the lengths of key, the number of population. It is apparent from the results that the different mutation rates have been regarded where the mutation rate 0.2 gives highest fitness that gives the highest number of correct letters. This is evident in Fig.8. Besides that increasing the number of population above 20 was helpful in retrieving the original key. This is evident in Fig.10 where the highest fitness was attained after 50 generations, getting all correct letters in the used keys.

Using GA to attack a poly-alphabetic substitution cipher (Vigenère cipher) proved to be an efficient method of cryptanalysis based on the aspect of comparing the frequency of letter occurrence in the model text.

REFERENCES

[1] Lin, F.-T. and C.-Y. Kao, *A Genetic Algorithm for Ciphertext-Only Attack in Cryptanalysis.* IEEE International Conference, **1**: p. 650-654, 1995.

[2] Trappe, W. and L. Washington, *Introduction to Cryptography with Coding Theory*. Pearson International Edition ed. second, India: Pearson Practice Hall, 2006..

[3] GARG, P., *Genetic Algorithms, Tabu Search And Simulated Annealing: A comparison Between Three Approaches For The Cryptanalysis Of transposition Cipher.* Journal of Theoretical and Applied Information Technology, p. 387-392, 2009.

[4] Clark, A. *Modern Optimisation Algorithms for Cryptanalysis* in *1994 second Australian and New Zealand Conference, 1994.*

[5] Toemeh, R. and S. Arumugam, *Applying Genetic Algorithms For Searching Key Space Of Poly Alphabetic Substitution Ciphers.* The international Arab journal of information technology. **5**(1): p. 87-91, 2006.

[6] Clark, A. and E. Dawson, *A Parallel Genetic Algorithm For Cryptanalysis of The Poly alphabetic Substitution Cipher.* Cryptologia. **21**(2): p. 129-138, 1997.

[7] Bergamann, K.P., *Cryptanalysis Using Nature-Inspired Optimization Algorithms, Msc Thesis*, in *Department of computer science*. The University Of Calgary: Galgary, Alberta., August 2007.

[8] Stinson, D.R., *Cryptography Theory and practice*, 3rd Edition, Chapman and Hill/ CRC, 2006.

[9] Denning, D.E., *Cryptography and data security ,* Addisson-Wesley publishing company, Reading, 1982.

[10] Henk, G.A., *Fundamentals of Cryptology, A professional references and interactive tutorial*, Kluwer Academic publishers, New Your, London and Moscow, 2000.

[11] Schneier,B., *Applied Cryptography Protocols , Algorithms and Source Code in C.* Second edition, Wiley India edition 2007.

[12] Stallings, W., *Cryptography and Network Security, Principles and Practices*. Pearson Education, 3rd Edition , India: Pearson Education, 2003.

[13] Jones, C.F. and M. Christman, *Genetic Algorithm Solution of Vigenere Alphabetic Codes.* Soft Computing in Industrial Applications, 2001. SMCia/01. Proceeding of the 2001 IEEE Mountain Workshop on 25-27: p. 59-63. June, 2001.

[14] Friedman, W.F., *Military Cryptanalysis , Part II. SIMPLER VARIETIES OF POLYALPHABETIC SUBSTITUTION SYSTEMS*. United State . WASHINGTON: United State . WASHINGTON, 1938.

[15] Bauer, F.L., *Decrypted Secrets , Methods and Maxims of Cryptology*. 4th ed, Verlag Berlin Heidelberg: Springer, 2007.

[16] Sikora, T.F**.**, *Basic Cryptanalysis*, in *A cryptography series*, U.S. GOVERNMENT PRINTING OFFICE:USA, 1990.

[17] Verma, A.K., M. Dave, and R.C. Joshi, *Genetic Algorithm and Tabu Search Attack on the Mono-Alphabetic Substitution Cipher Adhoc Networks.* Journal of Computer Science, **3**(3): p. 134-137, 2007.

[18] Hill, P.C.J. and *L.S. Member*, *Vigenère through Shannon to Planck – a Short History of Electronic Cryptographic Systems.* IEEE 2008. **HISTELCON-08**: p. 41-46, 2008.

[19] Holland, J., *"Adaptation in natural and artificial systems",* University of Michigan Press, Ann, Arbor, 1975.

[20] Goldberg, D.E, *Genetic algorithms in search, Optimization and Machine learning,* Addison-Wesley publishing company, Reading, 1989.

[21] Delman, B., *Genetic Algorithms in Cryptaography*, in *Department of Computer Engineering*. Msc. Thesis. Kate Gleason College of Engineering , Rochester Institute of Technology: Rochester, New Yourk , July 2004..

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

[23] Melanie, M., *An Introduction to Genetic Algorithms*. 5th ed, Cambridge, Massachusetts • London, England: The MIT Press, 1999.

[24] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*. Revised and Extended Edition, ed. Third.: Springer, 1996.

[25] *Genetic Algorithm and Direct Search Toolbox 2 , User's Guide*. 2009.