Contents lists available at GrowingScience

Management Science Letters

homepage: www.GrowingScience.com/msl

A new method for decoding an encrypted text by genetic algorithms and its comparison with tabu search and simulated annealing

Mahdi Sadeghzadeh[a*] and MahsaTaherbaghal[b]

[a]*Department of Computer, Mahshahr Branch, Islamic Azad University, Mahshahr, Iran*
[b]*Department of Computer, Dezful Branch, University OF Applied Science and Technology, Dezful, Iran*

CHRONICLE

Article history:
Received June 28, 2013
Received in revised format
19 October 2013
Accepted 20 December 2013
Available online
December 31 2013

Keywords:
Decoding
Genetic algorithms
Search algorithms
Simulated annealing algorithms
Code analysis

ABSTRACT

Genetic Algorithm is an algorithm based on population and many optimization problems are solved with this method, successfully. With increasing demand for computer attacks, security, efficient and reliable Internet has increased. Cryptographic systems have studied the science of communication is hidden, and includes two case categories including encryption, password and analysis. In this paper, several code analyses based on genetic algorithms, tabu search and simulated annealing for a permutation of encrypted text are investigated. The study also attempts to provide and to compare the performance in terms of the amount of check and control algorithms and the results are compared.

1. Introduction

During the past few years, there has been growing demand for efficient and effective internet security and even some small changes may destroy the information and make it impossible to retrieve the source, properly. Therefore, we need to make necessary action to protect data against their attacker and guarantee to delivery of information through different procedures such as Cryptography (William & Stallings, 2006). Knowledge creation, analysis of cryptographic encryption and decryption methods includes two categories including encryption, password and analysis, which is associated with creation, efficient and effective encryption of data. The primary objective of cryptography is to allow the applicant authority to receive messages to prevent any possible message eavesdropping. In other words, the process is to search for gaps or inaccuracies in the encryption data (Albassall & Wahdan, 2004; Spillman et al., 1993). There are different types of classical ciphers but most fall into

* Corresponding author.
E-mail addresses: sadegh_1999@yahoo.com (M. Sadeghzadeh)

doi: 10.5267/j.msl.2013.12.037

one of two broad categories: substitution ciphers and transposition ciphers. In the former one, every plaintext character is substituted by a cipher character, using a substitution alphabet, and in the latter one, plaintext characters are permuted based on a predetermined permutation technique. Now new encryption systems have replaced the classic system, but searches for innovative analysis of classical codes are more common.

A permutation technique moves a password or breaks a block with fixed size and then works the characters in each block based on a particular permutation *P* such that permutations are accomplished. A simple transposition or permutation cipher works by breaking a message into fixed size blocks, and then permuting the characters within each block based on a fixed permutation, say *P*. The key to the transposition cipher is simply the permutation *P*. Therefore, the transposition cipher has the property that the encrypted message contains all the characters that were in the plaintext message. In other words, the unigram statistics for the message are unchanged by the encryption process. Let's consider an example of a transposition cipher with a period of ten 10, and a key *P={7, 10, 4, 2, 8, 1, 5, 9, 6, 3}*. In this case, the message is broken into blocks of ten characters, and after encryption, the seventh character in the block are moved to position 1, the tenth moved character in the block is moved to the second position, the forth is moved to position 3, the second to position 4, the eighth to position 5, the first to position 6, the fifth to the position 7, the ninth to the position 8, the sixth to the position 9 and the third is moved to position 10.

Table 1 shows the key and the encryption process of the previously described transposition cipher. It can be noticed that the random string "*X*" was appended to the end of the message to enforce a message length, which is a multiple of the block size. It is also clear that the decryption can be achieved by following the same process as encryption using the "inverse" of the encryption permutation. In this case the decryption key, P-1 is equal to {6, 4, 10, 3, 7, 9, 1, 5, 8, 2}.

Table 1
Example transposition cipher key and encryption

Key:
Simple text: 1,2,3,4,5,6,7,8,9,10
Encrypted text: 7,10,4,2,8,1,5,9,6,3
Cryptography:
Position : 12345678910 1234 5678 910 12345678910
Plaintext : TRANSPOSITION _ ALGORITHMXXXXXXX
Ciphertext : OTNRSTSIPAGI _ OOIARLNXXXHXTXXXM

2. Methodology

In this section, we present details of the implementation of genetic algorithm, tabu search and simulated annealing to attack the password handling. Fitness functions or evaluates function in this article is as follows,

$$C_{\mathrm{k}} = \alpha \sum_{(\mathrm{i}\epsilon\widetilde{\mathrm{A}})} \left|\mathrm{K}^{\mathrm{u}}_{(\mathrm{i})} - \mathrm{D}^{\mathrm{u}}_{(\mathrm{i})}\right| + \beta \sum_{(\mathrm{i,j}\epsilon\widetilde{\mathrm{A}})} \left|\mathrm{K}^{\mathrm{b}}_{(\mathrm{i,j})} - \mathrm{D}^{\mathrm{b}}_{(\mathrm{i,j})}\right| + \gamma \sum_{(\mathrm{i,j,k}\epsilon\widetilde{\mathrm{A}})} \left|\mathrm{K}^{\mathrm{t}}_{(\mathrm{i,j,k})} - \mathrm{D}^{\mathrm{t}}_{(\mathrm{i,j,k})}\right| \tag{1}$$

Here, Ã indicates where the alphabet is the language (e.g. English language {A... Z, _} _ indicated that the gap is the distance), *K* and *D*, respectively represent language characters and text characters are encrypted. In addition, b, u, t represent the single-syllable words, two-syllable, and are trigram, respectively. α and β, and γ represent the different weights assigned to each letter with α + β + γ = 1. When words are used trigram function complexity is O (N3) and the alphabet size is N.

Table 2 contains weights used for two-syllable and three-syllable words.

Table 2
Weights proposed by Clark (1998a, 1998b)

Bi/trigram	Score
TH	+2
HE	+1
IN	+1
ER	+1
AN	+1
ED	+1
THE	+5
ING	+5
AND	+5
EEE	-5

Table 3
Reformed weights by Clark (1998a, 1998b)

Bi/trigram	Score
E_	+2
S_	+1
__	-6
_T	+1
_TH	+5
HE	+1
THE	+5
TH	+1
HE_	+5
_A	+1
___	-10

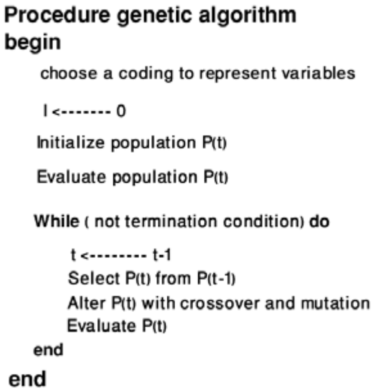
Table 3 demonstrates the weights modified by Clark (1998a, 1998b). Table 4 the weights used in this article shows.

Table 4
Reformed weights by Clark (1998a, 1998b)

Bi/trigram	Score
	-6
TH	+1
S_	+1
HE	+1
THE	+5
IN	+1
ING	+1
ER	+1
AND	+5
AN	+1
EEE	-5
ED	+1
HE_	+5
E_	+2
_TH	+5
_T	+1
___	-10
_A	+1

2.1. Genetic algorithm

Genetic Algorithms (GA) by John Holland and students at the University of Michigan, was introduced in early 1970 (Holland, 1975; Spears et al., 1993; Kolodziejczyk, 1997; Yaseen & Sahasrabuddhe, 1999; Grundlingh & Van Vuuren, 2003). This is a kind of method to solve complicated problems using the laws in nature. GA is a population of potential solutions, which must be maintained based on different rules such as selection and genetic composition of this multi-operator and mutation are developed (Matthews, 1993). Individuals in this population using a fitting function that mimics the environment are assessed. Fitness value of individuals with higher chances of survival structures to move to the next generations is formed. A quasi-public code in the form of a genetic algorithm is shown in Fig. 1.



```
Procedure genetic algorithm
begin
    choose a coding to represent variables
    I <------- 0
    Initialize population P(t)
    Evaluate population P(t)
    While ( not termination condition) do
        t <-------- t-1
        Select P(t) from P(t-1)
        Alter P(t) with crossover and mutation
        Evaluate P(t)
    end
end
```

Fig1. Simple genetic algorithm cycle

In this section, a GA method with assault with a password handling is expressed. Intercourse used to create two children is expressed as follows,

1 - Note: p1 and p2 are the parents, c1 and c2 are children, pi (j) j element defines the parent *i*, ci (j) specifies the elements of the child *i* is *j* {C_{ij}, k} specifies collection of child elements in *j* where *i* have to bet with *N*.

$$k = 0\ OR\ j = p + 1\ then\ \{C_i^{j,k}\} = \{\emptyset\} \quad (2)$$

```
1- Child 1 :
    a. Select a random numberr∈[1,P]
    b. ci(j)= pi(j) for j=1,……,r
    c. for i=1..........P-r and
          k=1... P
          If p2 (k) ∉{Cl, i+r−1} then
                $C_i$ (i+r) =p2 (k)
          Else
                k=k+1
2- child 2:
    a. Select a random numberr∈[1,P]
    b. C2(j)= pi(j) for j=P,……,r
    c. for i=1..........r and
          k=P… 1
          If p2 (k) ∉ {Cr−i+1, P} then
                c2(r-i) =p2 (k)
          Else
                k=k-1
```

Random mutations percent certain bits in the list and can change chromosomes. This can introduce features that should not related to original population and the genetic algorithm converges very fast before the full sampling procedure keeps costs. In this paper, the mutation operator randomly selects a position of genes and gene value in the range of gene changes. Note that the mutation operator may lead to an unauthorized person. Password displacement under attack by genetic algorithms is described as follows,

1 - Will receive the required algorithms: encrypted text (and its length), the words in English alphabet (one syllable, two syllables and polysyllabic.)
2 - Initialize the algorithm parameters. Population and maximum number of occurrences of M MAX.
3 - Create the initial population of random solutions and calculated P CURR cost solutions within each population based on Formula 1.
4 - For i = 1... MAX, do

a. Select the number of M / 2 pairs of keys P (CURR) for the generation of new parents have.
b. Combination of operator for each pair of selected parents to create a new population of solutions P (NEW).
c. For each mutation operator M use child thing,
d. For each child within P (NEW) using Formula 1, calculate the fitted rate,
e. Assign P (NEW) fitted to the highest (least cost) to the least fit (most expensive) and order them,
f. Arrange P (CURR) and P (NEW) to get a sorted list of solutions and merge with (duplicate is ignored), the list size is between *M* and *2M*. Select the number of *M* best solutions from the list merged to form *P* (CURR) new.
5- Output the best result from P (CURR).

2.2. Tabu search

The basic concept of Tabu Search is described by Glover (Glover, 1990; Glover & Laguna, 1993, 1997) for solving combinatorial optimization problem. It is a kind of iterative search and is characterized by the use of a flexible memory. It is able to eliminate local minima and to search beyond the local minimum. Therefore, it has the ability to find the global minimum multimodal search space. Tabu search by the memory structure, such as genetic algorithms are also used. The targeted use of funds has more memory. Tabu search by the memory uses more goals, including preventing solutions back to the search area. Keeping the list of possible solutions, which involves exposure, is achieved by the previous. Tabu search also is a mechanism to control the search. Tabu list makes us confident that some solutions are unacceptable. However, limitations created by tabu list can be limited in some cases the algorithm and a local optimal solution to restrict. Tabu search criteria can also expect a concept (aspiration) to overcome this issue provides. Alternative measures instead of waiting restrictions will create tabu possibility that the search for global optimization we generalize. Password displacement can also be attacked by the tabu search. General this algorithm is expressed in the following:

```
1. Input: Intercepted ciphertext, and the language statistics.
2. Initialize the algorithm parameters: the solution pool size M and the maximum number of iterations MAX.
3. Randomly generate an initial pool of solutions PCURR, and calculate the cost of each of the solutions in the pool using equation (1).
4. For I =1,… , MAX do:
   a. Find the best key in the tabu list which has the lowest cost associated with it. Call this key KBEST.
   b. For j=1,…, SPOSS do:
      i. apply the perturbation mechanism described in the simulated annealing attack to produce a new key KNEW,
      ii. Check if KNEW is already in the list of possibilities generated for this iteration or the tabu list. If so, return to step 4(b) I,
      iii. Add KNEW to the list of possibilities for this iteration.
   c. From the list of possibilities for this iteration, find the key with the lowest cost, PBEST,
   d. From the tabu list, find the key with the highest cost, TWORST,
   e. While the cost of PBEST is less than the cost of TWORST:
      i. Replace TWORST with PBEST,
      ii. Find the new PBEST,
      iii. Find the new TWORST.
5. Output the best solution from the tabu list, KBEST (the one with the least cost).
```

2.3. Simulated annealing

Kirkpatrick (1984) presented an algorithm based on the agreement between solids simulated annealing and problem solving hybrid optimization problems. Simulated annealing process cooling and heating of solids is slow to achieve the lowest energy state. Thinking of imitating the attitude of the fusion process was sufficient. Algorithm with random values of the solution is initialized to the start at the beginning of problem solving and temperature T (0). Temperature is slowly reduced and at any temperature in a number of solutions now tries to create confusion. Disturbed at any temperature changes in the cost function, ΔE determines the changes of energy. If ΔE <0 the chaos is accepted, otherwise the potential Metropolis by equation is defined as the following are accepted.

$$Probablity\ (E1 \rightarrow E2) = e^{(-\Delta E/T)} \tag{2}$$

Here E1 and E2 are the cost functions, ΔE changes in the cost function and T is the current temperature. If the proposed changes were accepted, then the current solution would subject to change. Password attacks by moving simulated annealing algorithm are expressed in the following:

1. Set the initial temperature, T (0),
2. Generate an initial solution - arbitrarily set to the identity transformation (could be randomly generated or otherwise),
3. Evaluate the cost function for the initial solution. Call this C(0),
4. For temperate T do many (e.g., 100 ×M) times:
   - Generate a new solution by modifying the current one in some manner,
   - Evaluate the cost function for the newly proposed solution,
   - Consult the Metropolis function to decide whether or not the newly proposed solution is accepted,
   - If accepted, update the current solution and its associated cost,
   - If the number of accepted transitions for temperature T exceeds some limit (e.g. 10×M) then jump to Step 5,
5. If the number of accepted transitions for temperature T was zero then stop (return the current solution as the best), otherwise reduce the temperature (e.g. $T^{(i+1)}=T^{i}\times 0.95$ and return to step 4).

3. The results

All three methods have been implemented in Matlab 7.1 software in system with Microsoft Windows XP Professional Version 2002 Service Pack 2, Intel(R) processor with Celeron processor with CPU 2.40GHz. Table 5 demonstrates the summary of our computations.

Table 5
Number of keys obtained against different password text size

Encrypted text size	Genetic Algorithm	Tabu search	Simulated annealing
100	5.25	4.7	5
200	8	7	7.6
400	12.75	10	11.25
600	13.7	11.75	12.25
11.75	15.2	12.6	13.75
1000	15.75	14	14.6

In Table 5, average production right on key elements for the expression levels are 15 permutations. Note that because of the encryption permutations used in a piece, the size of permutations can be done by a long text to decode correctly and the correct key is called the neighborhood when all the elements are identical (except for last place). The text is readable, especially when the size is large permutations. Fig. 2 demonstrates the results of three algorithms.

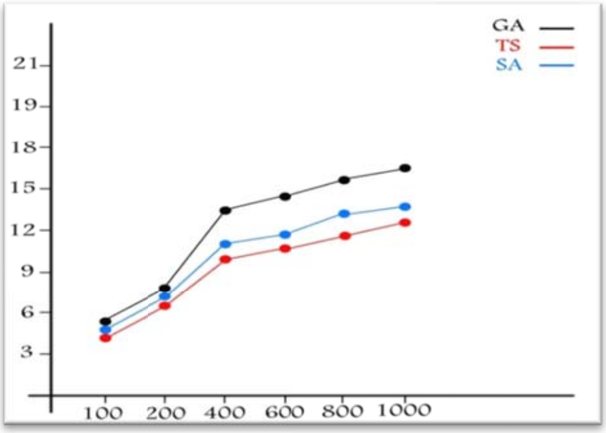


Fig. 2. Number of keys obtained against the encrypted text size

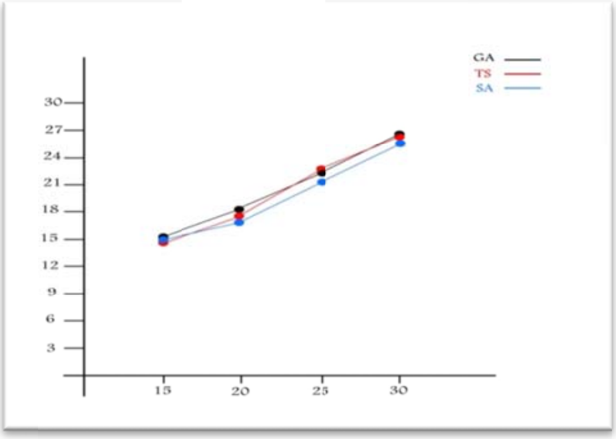


Fig. 3. Number of keys versus the amount found in different permutations

The second comparison is based on the amount of permutations as permutations in the text. Table 6 shows the number of correct keys obtained from a 1000 text encrypted word permutations. Fig. 3 shows details of the results of three algorithms.

Table 6
Number of keys versus the amount found in different permutations

Size of permutations	Genetic Algorithm	Tabu search	Simulated annealing
15	15.25	14.75	14.76
20	18	17.50	17
25	21.75	22.00	21.25
30	26.7	26.50	26

When the encrypted text size is too big, simply tabu search algorithm along with two genetic algorithms and simulated annealing provide promising results. In addition, the mean and standard deviation of the number of bits found in the correct keys for each algorithm is repeated 50 times. Table 7 shows details of average and standard deviation of three methods.

Table 7
The average and SD derived from 50 times to encrypt text simple assault

Encrypted text size	Genetic Algorithm		Tabu search		Simulated annealing	
	Average	Std. dev.	Average	Std. dev.	Average	Std. dev.
200	7.4	2.8	8.1	2.7	7.5	2.3
500	8.1	2.5	9.2	2.2	8.4	2.0
1000	9.1	2.2	10	1.8	9.2	1.9

4. Conclusion

In this paper, we have presented a genetic algorithm for data encryption and compared the performance of the proposed study with two alternative methods, namely simulated annealing and tabu search. The preliminary results indicate that the implementation of fixed permutation with an adaptation of GA method seems to perform better than other techniques. In addition, tabu search also seems to perform well under certain circumstances. The performance of the proposed method depends on the input parameters and when they are carefully tuned, we may get better results.

References

Albassall, A. M. B., & Wahdan, A. (2004, September). Genetic algorithm cryptanalysis of a feistel type block cipher. In *Electrical, Electronic and Computer Engineering, 2004. ICEEC'04. 2004 International Conference on* (pp. 217-221). IEEE.

Clark, A. J. (1998a). Optimisation heuristics for cryptology, PhD. Thesis.

Clark, A. J. (1988b). Physical protection of cryptographic devices. *Advances in Cryptology—EUROCRYPT'87* (pp. 83-93). Springer Berlin Heidelberg.

Garg, P. (2006). Genetic algorithm attack on simplified data encryption standard algorithm. *Special Issue: Advances in Computer Science and Engineering*, 23, 139-174.

Verma, A. K., Dave, M., & Joshi, R. C. (2007). Genetic Algorithm and Tabu Search Attack on the Mono-Alphabetic Subsitution Cipher in Adhoc Networks.*Journal of Computer science*, *3*(3), 134.

Glover, F. (1990). Tabu search—part II. *ORSA Journal on computing*, *2*(1), 4-32.

Glover, F., & Laguna, M. (1997). *Tabu search* (Vol. 22). Boston: Kluwer academic publishers.

Glover, F., & Taillard, E. (1993). A user's guide to tabu search. *Annals of operations research*, *41*(1), 1-28.

Grundlingh, W., & Van Vuuren, J. H. (2003). Using Genetic Algorithms to Break a Simple Cryptographic Cipher. *Retrieved March*, *31*.

Holland, J. H. (1975). *Adaption in Natural and Artificial Systems. Ann Arbor, Michigan: University of Michigan Pres.*

Yaseen, I. F., & Sahasrabuddhe, H. V. (1999). A genetic algorithm for the cryptanalysis of Chor-Rivest knapsack public key cryptosystem (PKC). *Computational Intelligence and Multimedia Applications, 1999. ICCIMA'99. Proceedings. Third International Conference on* (pp. 81-85). IEEE.

Kolodziejczyk, J. (1997). The Application of Genetic Algorithm in Crptoanalysis of Knapsack Cipher. In *European School on Genetic Algorithms, Eurogen97.*

Matthews, R. A. (1993). The use of genetic algorithms in cryptanalysis.*Cryptologia*, *17*(2), 187-201.

Kirkpatrick, S. (1984). Optimization by simulated annealing: Quantitative studies. *Journal of statistical physics*, *34*(5-6), 975-986.

Spears, W. M., De Jong, K. A., Bäck, T., Fogel, D. B., & De Garis, H. (1993, January). An overview of evolutionary computation. In *Machine Learning: ECML-93* (pp. 442-459). Springer Berlin Heidelberg.

Spillman, R., Janssen, M., Nelson, B., & Kepner, M. (1993). Use of a genetic algorithm in the cryptanalysis of simple substitution ciphers. *Cryptologia*, *17*(1), 31-44.

William, S., & Stallings, W. (2006). *Cryptography and Network Security, 4/E.* Pearson Education India.