IARS' International Research Journal
p-ISSN 2202-2821 e-ISSN 1839-6518 (Australian ISSN Agency)
Vol. 08 No. 01 2018
82800801201801

Cryptanalysis of Mono-Alphabetic Substitution Ciphers using Genetic Algorithms and Simulated Annealing

Shalini Jain
Marathwada Institute of Technology, Dr. Babasaheb Ambedkar Marathwada University, India

Nalin Chhibber
University of Waterloo, Ontario, Canada

Sweta Kandi
Marathwada Institute of Technology, Dr. Babasaheb Ambedkar Marathwada University, India

ABSTRACT – *In this paper, we intend to apply the principles of genetic algorithms along with simulated annealing to cryptanalyze a mono-alphabetic substitution cipher. The type of attack used for cryptanalysis is a ciphertext-only attack in which we don't know any plaintext. In genetic algorithms and simulated annealing, for ciphertext-only attack, we need to have the solution space or any method to match the decrypted text to the language text. However, the challenge is to implement the project while maintaining computational efficiency and a high degree of security. We carry out three attacks, the first of which uses genetic algorithms alone, the second which uses simulated annealing alone and the third which uses a combination of genetic algorithms and simulated annealing.*

Key words: Cryptanalysis, mono-alphabetic substitution cipher, genetic algorithms, simulated annealing, fitness

I. Introduction

Cryptanalysis involves the dissection of computer systems by unauthorized persons in order to gain knowledge about the unknown aspects of the system. It can be used to gain control of the encrypted data, even if the key is not known to the cryptanalyst. It is one of the two main components of cryptology, the study of codes; the other one being cryptography. Cryptographic methods are useful in developing encryption patterns so as to secure the confidentiality of a message.

In this work, we deal with two parts: The first part involves developing a cryptosystem using the cryptographic principles of mono-alphabetic substitution ciphers. The second part deals with breaking (gaining access to the ciphertext) the developed cryptosystem using genetic algorithm technique in combination with simulated annealing.

The rest of this paper is organized as follows. In Section II, mono-alphabetic substitution cipher is described in detail. Section III deals with the cryptanalysis of mono-alphabetic ciphers. Genetic algorithms and simulated annealing are discussed in Section IV. The results are depicted in Section V. Finally, our conclusions are drawn in section VI.

II. Mono-alphabetic substitution cipher

A mono-alphabetic substitution cipher is a class of substitution ciphers in which the same letters of the plaintext are replaced by the same letters of the ciphertext. Mono, which means one, signifies that each letter of the plaintext has a single substitute of the ciphertext. In a poly-alphabetic substitution cipher, multiple substitutions are made in the ciphertext corresponding to the letters in the plaintext [1-2].

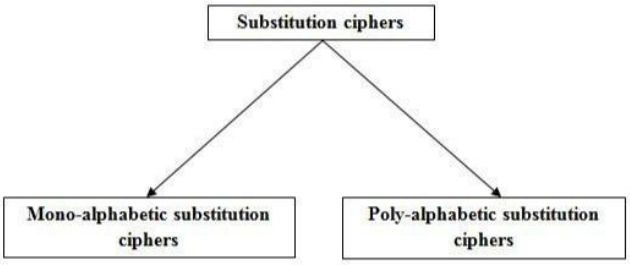

Substitution ciphers
Mono-alphabetic substitution ciphers
Poly-alphabetic substitution ciphers

Figure 1: Types of substitution ciphers

A. Affine cipher – a mono-alphabetic substitution cipher

The affine cipher is a type of mono-alphabetic substitution cipher, in which each letter is mapped to its numeric equivalent, encrypted using a simple mathematical function, and converted back to a letter. Since affine cipher is an example of mono-alphabetic substitution cipher, it has the disadvantages associated with all mono-alphabetic ciphers. Each letter in the alphabet is encrypted with the function (ax+b) mod (26), where a is the key and b is the magnitude of the shift. Here, a should be co-prime to 26 [3].

B. Encryption and decryption in affine cipher

Encryption function is as follows:

E(x) = (ax + b) mod 26

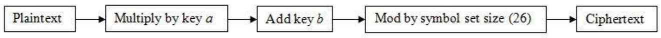


Figure 2: Encryption Process

Decryption function is as follows:

D(x) = a-1 (x – b) mod 26

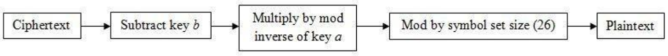


Figure 3: Decryption Process

III. Cryptanalysis of mono-alphabetic ciphers

A. Types of cryptanalysis attacks

- Ciphertext only: the cryptanalyst has access only to a collection of ciphertexts.
- Known plaintext: the attacker has a set of ciphertexts to which he knows the corresponding plaintext.
- Chosen plaintext: the attacker can obtain the ciphertexts corresponding to an arbitrary set of plaintexts of his own choosing.
- Chosen ciphertext: the attacker can obtain the plaintexts corresponding to an arbitrary set of ciphertexts of his own choosing.

B. Frequency analysis of substitution ciphers

Frequency analysis is the study of frequency of letters in the ciphertext. The basic use of frequency analysis is to first count the frequency of ciphertext letters and then associate guessed plaintext letters with them [4]. More X's in the ciphertext than anything else suggests that X corresponds to e in the plaintext, but this is not certain; t and a are also very common in English, so X might be either of them also. It is unlikely to be a plaintext z or q which are less common. Thus the cryptanalyst may need to try several combinations of mappings between cip hertext and plaintext letters [5-6].

More complex use of statistics can be conceived, such as considering counts of pairs of letters (digrams), triplets (trigrams), and so on. This is done to provide more information to the cryptanalyst.

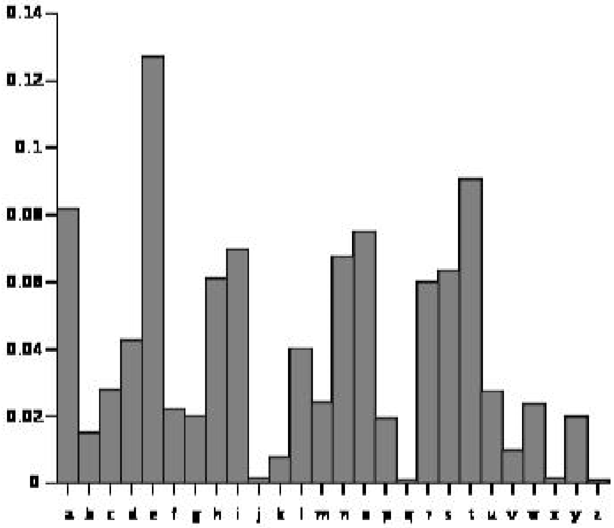


Figure 4: Relative frequency of letters in the English alphabet

IV. Genetic algorithms and simulated annealing

A. Genetic algorithms

A genetic algorithm (or GA) is a search technique used in computing to find true or approximate solutions to optimization and search problems. Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination). The evolution usually starts from a population of randomly generated individuals and happens in generations [7-8].

In each generation, the fitness of every individual in the population is evaluated, multiple individuals are selected from the current population (based on their fitness), and modified (recombined and possibly mutated) to form a new population [9].

The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfa ctory fitness level has been reached for the population [10]. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

B. Measure of fitness

The fitness function for this work has been evaluated by making use of frequency analysis of the known text and the decrypted text. The statistics have been applied for co unting the n- grams, namely, unigram, bigram and trigram [9].

Fitness function formula:

F = 0.1 * [K(u) – D(u)] + 0.3 * [K(b) – D(b)] + 0.6 * [K(t) – D(t)]

where

K = known text statistics (using affine cipher)

D = decrypted text statistics (using GA chromosomes)

u = unigram count

b = bigram count

t = trigram count

Here, we have chosen 0.1, 0.3 and 0.6 as the weights for unigram, bigram and trigram respectively.

C. Simulated annealing

There are certain optimization problems that become unmanageable using combinatorial methods as the number of objects becomes large. A typical example is the traveling salesman problem, which belongs to the NP-complete class of problems. For these problems, there is a very effective practical algorithm called simulated annealing (thus named because it mimics the process undergone by misplaced atoms in a metal when its heated and then slowly cooled). While this technique is unlikely to find the optimum solution, it can often find a very good solution, even in the presence of noisy data [11-13].

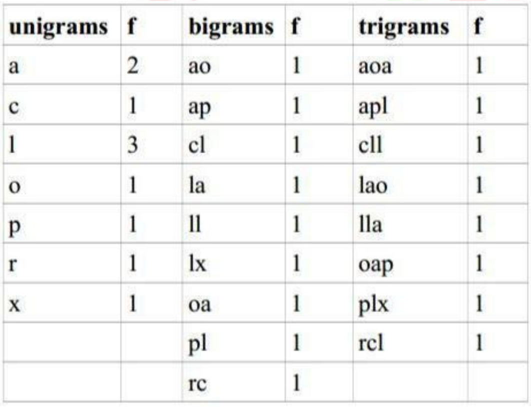
V. Results

A. Generation of training data from affine cryptosystem

Plaintext: helloworld

Ciphertext: rcllaoaplx

B. Primary frequency analysis of the ciphertext

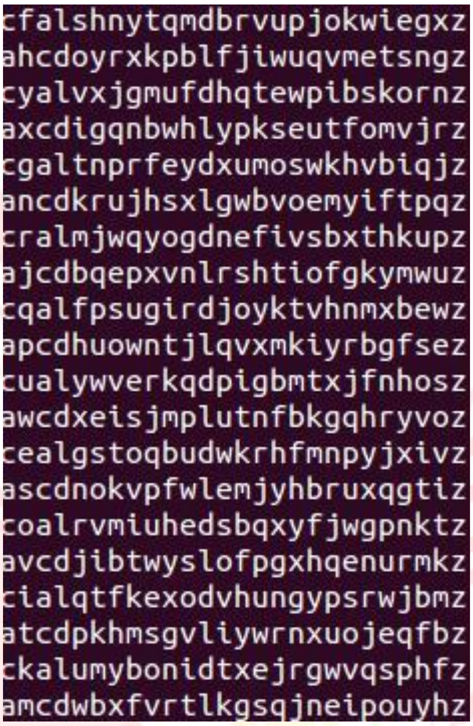


unigrams	f	bigrams	f	trigrams	f
a	2	ao	1	aoa	1
c	1	ap	1	apl	1
l	3	cl	1	cll	1
o	1	la	1	lao	1
p	1	ll	1	lla	1
r	1	lx	1	oap	1
x	1	oa	1	plx	1
		pl	1	rcl	1
		rc	1		

Figure 5: Frequency analysis of the ciphertext

VI. 5.3 Initialization of genetic algorithm

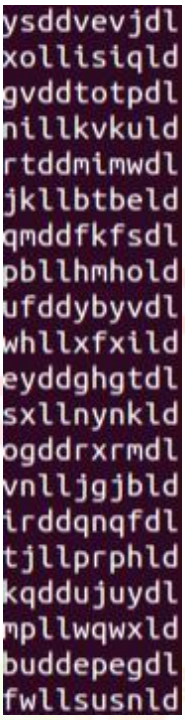
- Number of chromosomes = 20
- Length of each chromosome = 26
- Total number of generations = 100



```
cfalshnytqmdbrvupjokwiegxz
ahcdoyrxkpblfjiwuqvmetsngz
cyalvxjgmufdhqtewpibskornz
axcdigqnbwhlypkseutfomvjrz
cgaltnprfeydxumoswkhvbiqjz
ancdkrujhsxlgwbvoemyiftpqz
cralmjwqyogdnefivsbxthkupz
ajcdbqepxvnlrshtiofgkymwuz
cqalfpsugirdjoyktvhnmxbewz
apcdhuowntjlqvxmkiyrbgfsez
cualywverkqdpigbmtxjfnhosz
awcdxeisjmplutnfbkgqhryvoz
cealgstoqbudwkrhfmnpyjxivz
ascdnokvpfwlemjyhbruxqgtiz
coalrvmiuhedsbqxyfjwgpnktz
avcdjibtwyslofpgxhqenurmkz
cialqtfkexodvhungypsrwjbmz
atcdpkhmsgvliywrnxuojeqfbz
ckalumybonidtxejrgwvqsphfz
amcdwbxfvrtlkgsqjneipouyhz
```

Figure 6: Initial chromosome population

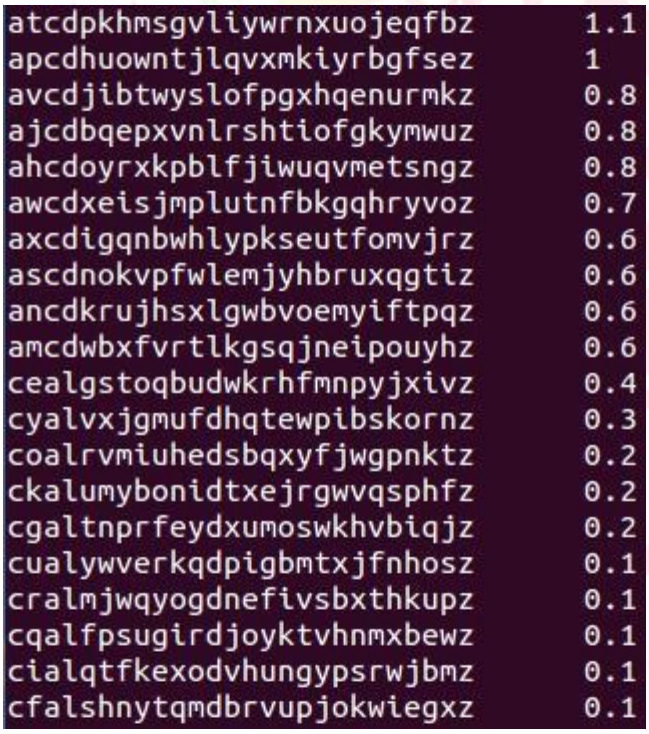
A. Mapping of chromosomes



```
ysddvevjdl
xollisiqld
gvddtotpdl
nillkvkuld
rtddmimwdl
jkllbtbeld
qmddfkfsdl
pbllhmhold
ufddybyvdl
whllxfxild
eyddghgtdl
sxllnynkld
ogddrxrmdl
vnlljgjbld
irddqnqfdl
tjllprphld
kqddujuydl
mpllwqwxld
buddepegdl
fwllsusnld
```

Figure 7: Chromosome mapping to generate the decrypted ciphertexts

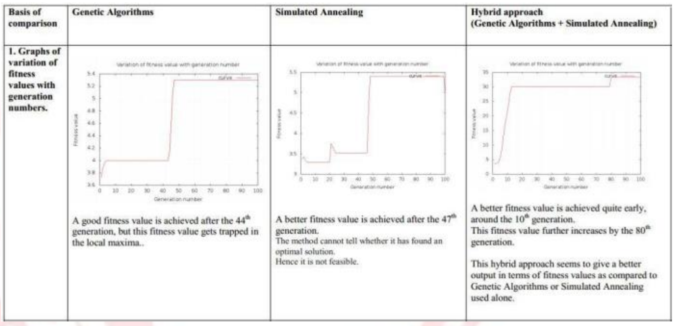
B. Fitness function calculation



```
atcdpkhmsgvliywrnxuojeqfbz      1.1
apcdhuowntjlqvxmkiyrbgfsez      1
avcdjibtwyslofpgxhqenurmkz      0.8
ajcdbqepxvnlrshtiofgkymwuz      0.8
ahcdoyrxkpblfjiwuqvmetsngz      0.8
awcdxeisjmplutnfbkgqhryvoz      0.7
axcdigqnbwhlypkseutfomvjrz      0.6
ascdnokvpfwlemjyhbruxqgtiz      0.6
ancdkrujhsxlgwbvoemyiftpqz      0.6
amcdwbxfvrtlkgsqjneipouyhz      0.6
cealgstoqbudwkrhfmnpyjxivz      0.4
cyalvxjgmufdhqtewpibskornz      0.3
coalrvmiuhedsbqxyfjwgpnktz      0.2
ckalumybonidtxejrgwvqsphfz      0.2
cgaltnprfeydxumoswkhvbiqjz      0.2
cualywverkqdpigbmtxjfnhosz      0.1
cralmjwqyogdnefivsbxthkupz      0.1
cqalfpsugirdjoyktvhnmxbewz      0.1
cialqtfkexodvhungypsrwjbmz      0.1
cfalshnytqmdbrvupjokwiegxz      0.1
```

Figure 8: Calculation of Fitness Values

C. Comparison of the fitness values obtained by using Genetic Algorithms, Simulated Annealing and using a hybrid approach of Genetic Algorithms along with Simulated Annealing



Basis of comparison	Genetic Algorithms	Simulated Annealing	Hybrid approach (Genetic Algorithms + Simulated Annealing)
1. Graphs of variation of fitness values with generation numbers.	A good fitness value is achieved after the 44th generation, but this fitness value gets trapped in the local maxima..	A better fitness value is achieved after the 47th generation. The method cannot tell whether it has found an optimal solution. Hence it is not feasible.	A better fitness value is achieved quite early, around the 10th generation. This fitness value further increases by the 80th generation. This hybrid approach seems to give a better output in terms of fitness values as compared to Genetic Algorithms or Simulated Annealing used alone.

Figure 9: Comparison between the three techniques

VII. Conclusion

In this paper, we discussed and implemented three kinds of attacks on a mono- alphabetic substitution cipher, called affine cipher. The first one uses genetic algorithms alone, the second one uses simulated annealing alone and the third one uses a combination of genetic algorithms along with simulated annealing. As using alone genetic algorithm or simulated annealing results in over- fitting, poor optimization time, often traps in local maxima. Whereas simulated annealing alone may not answer whether it has found an optimal solution, in certain cases complimentary method may be required. Also, it needs more iterations than other search heuristics to converge. Last but not least, it cannot be used in all the applications specially where fitness is smooth.

VIII. References

[1]. M. Ralph and W. Ralph, "A Word-Based Genetic Algorithm for Cryptanalysis of Short Cryptograms", American Association for Artificial Intelligence pp.229- 233, 2009.

[2]. D. Bethany, "Genetic Algorithm in Cryptography", Rochester Institute of Technology, Rochester, New York, July 2010.

[3]. W. R. Grundlingh and Jan Van Vuuren, "Using Genetic Algorithms to break a simple Cryptographic Cipher", March 31, 2005.

[4]. O. David, "Evolutionary Algorithm for Decryption of Mono-alphabetic Homophonic Substitution Ciphers Encoded as Constraint Satisfaction Problems", Atlanta, Georgia, USA, 2008.

[5]. R. Spillman, M. Janssen, B. Nelson, and M. Kepner, "Use of a genetic algorithm in the of simple substitution ciphers", Cryptologia 17(1), pp.31 - 44, January 2003.

[6]. A. J. Clark, Optimization Heuristics for Cryptology, PhD, Thesis, Queensland University of Technology, February 2012.

[7]. L. C. Washington, Introduction to cryptography with coding theory, Pearson Education, Inc., 2nd edition, 2006.

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

[9]. Using Genetic Algorithm "To Break A Mono - Alphabetic Substitution Cipher", S. S. Omran, A. S. A l-Khalid, D. M. Al-Saady, 2010 IEEE Conference on Open Systems (ICOS 2010), pp. 63 -67, 2010.

[10]. G. J. Simmons, Contemporary Cryptology, "The Science of Information Integrity", The Institute of Electrical and Electronics Engineers, Inc., New York, 2001.

[11]. D. Kahn, The Code breakers, The New American Library, Inc., USA, 2003.

[12]. W. Stallings, cryptography and network security, Pearson Education, Inc., 4th edition, 2005.

[13]. T. Ragheb and A. Subbanagounder, Applying Genetic Algorithms for Searching Key- Space of Poly-alphabetic Substitution Ciphers, The International Arab Journal of Information Technology, Vol. 5, No. 1, pp.87 - 91, January.