American Journal of Computer Science and Engineering Survey
a Pubicon Open Journal

RESEARCH OPEN ACCESS

The Firefly Algorithm and Application in Cryptanalysis of Monoalphabetic Substitution Ciphers

Amrit Pal Singh*[1], Dr. S K. Pal[2] and Dr. M P S Bhatia[3]

[1]GTBIT, GGSIPU, New Delhi, INDIA
[2]Scientific Analysis Group, DRDO, Metcalfe House Complex, Delhi – 110 054 INDIA
[3]NSIT, University of Delhi, New Delhi, INDIA
Email Id:
amritpal.ipu@gmail.com

Abstract

The firefly algorithm (FA) is Nature Inspired Algorithm; this algorithm is based on flashing behavior of fireflies. The Monoalphabetic substitution cipher creates cipher text by replacing each alphabet with another alphabet. The cryptanalysis of substitution cipher involves statistical data of language. In this paper we proposed new Firefly Algorithm for cryptanalysis of the Monoalphabetic substitution cipher and then compared result with random algorithm on the basis of convergence time and corrected percentage of decrypted text.

Keywords- Cryptanalysis, firefly, Monoalphabetic substitution cipher.

Pubicon

Introduction

In the context of combinatorial optimization (CO), algorithms can be classified as either complete or approximate algorithms. Complete algorithms are guaranteed to find for every finite size instance of a CO problem an optimal solution in bounded time. Yet, for CO problems that are *NP*-hard, no polynomial time algorithm exists, assuming that *P! = NP*[2].

Cryptology is the branch of science that deals with the study of methods for encryption and decryption of data, and is commonly employed in the transmission of secure data. Cryptology can be divided into two branches: cryptography and cryptanalysis. Cryptography deals with the creation of new and powerful schemes for the encryption of data. Cryptanalysis deals with the study of breaking these schemes of encryption by identifying possible flaws or vulnerabilities[5].

In this paper we use Firefly Metaheuristic algorithm. Firefly Algorithm (FA)[1,6,8] developed by Xin-She Yang at Cambridge University in 2007. FA is an optimization algorithm inspired by behavior and motion of fireflies.

In this paper we present new algorithm for cryptanalysis of the mono alphabetic substitution cipher by using firefly algorithm. English language statistical data has been used to assign fitness values to potential solutions. Such algorithm is expected to work best for large input cipher text lengths.

FIREFLY ALGORITHM

Behavior of Fireflies

The flashing light of fireflies is very attractive and beautiful view in the regions lying in the tropical and temperature ones. There are nearly around two thousand species of firefly, and most of the fireflies produce tiny rhythm making flashes. The pattern of flashes made by these fireflies are likely different for each particular species. The procedure of producing flash light is produced by a process commonly called as bioluminescence, while the debate on true functions of such signaling systems is still going on. However, two fundamental functions of such flashes are to attract various communicating partners, and to attract potential victim. In addition to this, the flashing technique might also be used to serve as a protective warning mechanism to remind the potential predators of the weird taste of the fireflies.

The rhythmic flash, the rate of flashing and the amount of time form part of the signal system that brings both the sex together. The female fireflies respond to a male fireflies unique pattern of attractive flashing in the same species, whereas in some other species such as Photuris, female fireflies can eavesdrop on the bioluminescent courtship signals and even mimic the mating flashing pattern of other species so as to lure and eat the male fireflies who may mistake the flashes as a potential suitable mate. Some tropic fireflies can also even synchronize their flashing techniques, thus they form emerging biological self-organized behavior.

We know that the light intensity at a particular distance r from the light source obeys the inverse square law. That is to say, the light intensity I decrease as the distance r increases in terms of **I α 1/r²**. Furthermore, the air absorbs light which becomes weaker and weaker as the distance between them goes on increasing. These two factors combine and make most of the fireflies visual till a limited distance, usually it can be several hundred meters at night, which is proved to be good for fireflies to communicate to each other.

Concept

Now we can idealize some of the flashing characteristics of fireflies so as to develop firefly-inspired algorithms. Flashing characteristics of fireflies uses to develop firefly-inspired algorithm. Firefly Algorithm (FA) developed by Xin-She Yang at Cambridge University in 2007, which use the following three idealized rules:

- All fireflies are of the same sex so that one firefly gets attracted to other one regardless of sex determining technique.
- Attractiveness is proportional to its desired brightness, hence for any of the two flashing fireflies, the less brighter firefly will move towards the more brighter firefly. The attractiveness of a firefly is directly proportional to its brightness and they both goes on decreasing as the distance between them goes on increasing. If there is no other brighter firefly than a particular one, it will move randomly in the space.
- The brightness of a firefly is often most affected or they can be determined by the landscape of the objective function.

For a maximization problem, the brightness can simply be proportional to the value of the objective function. Other type of brightness can also be stated in a similar way to the fitness function in genetic algorithms.

Light Intensity and Attractiveness

In the firefly algorithm, there are two important issues: the variation of light intensity and formulation of the attractiveness. Making it simple, we can consider that the attractiveness of a firefly is determined by its brightness which in turn is associated with the encoded objective function.

In the simplest case for maximum optimization problems, the brightness of a firefly, I at a particular location x can be chosen as $\mathbf{I(x) \propto f(x)}$. However, the attractiveness is relative, β, it should be judged by the other unisex fireflies. Thus, it will also differ with the distance rij between firefly i and j. In addition to this, the intensity of light goes on decreasing with the distance from its source point, and the media absorbs light, so the attractiveness should be allowed to vary with the degree of absorption. In the simple way, the intensity of light I(r) varies according to the inverse square law.

$$\mathbf{I(r) = \frac{I_s}{r^2}} \qquad \ldots\ldots\ldots\ldots\ldots (1)$$

where Is is the intensity at the source. For a particular medium, fixed light absorption coefficient γ and the intensity of light 'I' varies along the distance 'r', i.e.

$$\mathbf{I=Io}\ e^{-\gamma r} \qquad \text{................ (2)}$$

where Io is the original light intensity. In order to avoid the singularity at r = 0 in the expression Is/r^2, the combined effect of both the inverse square law and absorption can be approximated as the following Gaussian form

$$\mathbf{I=Io}\ e^{-\gamma r^2} \qquad \text{................ (3)}$$

As a firefly's attractiveness is proportional to the light intensity seen by other following fireflies, we can also define the constant of attraction, 'β'of a firefly by

$$\beta=\beta o\ e^{-\gamma r^2} \qquad \text{................ (4)}$$

Where βo is the attractiveness at r = 0. As it is often faster to calculate $1/(1 + r^2)$ than other exponential functions, the above function, can be approximated as

$$\beta=\frac{\beta o}{(1+\gamma r^2)} \qquad \text{.................. (5)}$$

Both (4) and (5) define a characteristic distance Γ=1/γ over which the attractiveness changes significantly from β_o to $\beta_o e^{-1}$ for equation (4) or $\beta_o/2$ for equation (5).

In the actual implementation, the attractiveness function β(r) can be any monotonically decreasing functions such as the following generalized form

(m > = 1).

$$\beta(r)=\beta o\ e^{-\gamma r^m} \qquad \text{................ (6)}$$

For a fixed, the characteristic length becomes

$$\mathbf{\Gamma=\gamma^{-1/m}\rightarrow 1,\ m\rightarrow\infty.} \qquad \text{................ (7)}$$

Conversely, for a given length scale Γ in an optimization problem, we can use the parameter γ as a typical initial value, i.e.

$$\gamma=\frac{1}{\Gamma^m} \qquad \text{................ (8)}$$

The distance between any two fireflies i and j at x_i and x_j, respectively, is known as the Cartesian distance

$$rij = ||xi - xj|| = \sqrt{\Sigma_{k=1}^{d}(xi,k - xj,k)^2} \quad \text{................ (9)}$$

Where xi,k is the kth component of the spatial coordinate xi of the ith firefly. In 2-D case, consider

$$rij = \sqrt{(xi - xj)^2 - (yi - yj)^2} \quad \text{..................... (10)}$$

The movement of a brighter firefly j is attracted by another least attractive (brighter) firefly is, and is determined by

$$xi = xi + \beta o\ e^{-\gamma rij^2}(xj - xi) + \alpha€i \quad \text{........................ (11)}$$

Where the second term is due to the attraction. The third term can be randomized along α being the randomization parameter, and €i is a vector of random numbers drawn from a Gaussian distribution or uniform distribution. For example, the simplest form is €i can be replaced by rand − ½ where rand is a random number generator uniformly distributed in [0, 1]. For most our implementation, we can take βo= 1 and α Є [0, 1].

It is worth pointing out that (11) is a random walk biased towards the brighter fireflies. If $\beta_o = 0$, it becomes a simple random walk. Furthermore, the randomized term can easily be extended to other distributions such as Levy flights.

The parameter γ now characterizes the variation in attraction, and its value is mostly important in determining the speed of the convergence and how the FA algorithm behaves. In theory, γ Є [0, ∞), but in practical, γ= O (1) can be determined by the characteristic length Γ of the system to be optimized. Thus, for most of the applications, most of the times Γ's lower limit is 0.1 and upper limit is 10.

```
FFA Meta-heuristic ()
Begin;
Initialize algorithm parameters:
MG: the maximum number of generations
γ: the light absorption coefficient
r: the finite distance from the light source
d: the domain space
Define the objective function of f(x), where x=(x1,........,xd)T
Generate the initial population of fireflies or xi (i=1, 2 ,..., n)
To calculate the light intensity of Ii at xi via f(xi)
while (t<MG)
for i = 1 to n (all n fireflies);
for j=1 to n (n fireflies)
 if (Ij > Ii),
move firefly i towards j by using 11 equation;
end if
Distance r affects the Attractiveness with factor Exp [$-\gamma r2$];
The updation of the light intensity depends upon the evaluation of new solutions;
end for j;
end for i;
 Sorting is done based on new solutions of fireflies, find the best in each iteration(current best);
End while;
Post process results and visualization;
End procedure
```

Variants of firefly Algorithm

Gaussian Firefly

Firefly algorithm has some disadvantage such as trapping into several local optimums. Firefly algorithm do local search as well and sometimes can't get rid of them. New firefly algorithm called GD-FF (Gaussian Distribution Firefly)algorithm[6] In standard Firefly algorithm, the agents make a move by just considering a predefined movement that guides them to better position in their neighborhood. In order to shift all fireflies in the same manner, it is used random walk concepts to move all of the agents based on a Gaussian distribution. In proposed algorithm, at the end of each iteration, a normal Gaussian distribution is introduced that is shown in Eq. (12).

$$p = f\left(\frac{x}{\mu}, \delta\right) = \frac{1}{\delta\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\delta^2}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (12)$$

Where *x* points to the error between best solution and fitness value of firefly *i*.

$$x = f(g_{best}) - f(x_i) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(13)$$

μ is mean and δ is known as a standard deviation. Because we use the standard normal distributions, it is set to μ=0 and δ=1. Then a random number will be drawn from this Gaussian distribution that is related to each firefly probability (p). Social behavior of fireflies is introduced by:

```
1. Objective function f(x), x=(x1,x2,…,xd)$^T$
2. Initialize a population of fireflies $x_i$ ( i= 1,2, . . ,n)
3. Define light absorption coefficient $\gamma$
4. While (t<MaxGeneration)
5. For i=1:n (all n fireflies)
6. For j=1:i
7. Light intensity is determined by f(xi)
8. If (Ii >Ij)
```

```
9.  Move firefly i towards j in all d dimensions
10. Else
11. Move firefly i towards best solution in thatiteration
12. End if
13. Attractiveness varies with distance via exp (-$\gamma r^2$)
14. End for j
15. End for i
16. Rank the fireflies and find the current best
17. Define normal distribution
18. For k = 1: n all n fireflies
19. Draw a random number from defined distribution
20. Evaluate new solution (new_cost(k))
21. If ((new_cost(k)<cost(i))&&(new_cost(k)<last_cost_iteration(k)))
22. Move firefly i towards current best
23. End if
24. End for k
25. End while
26. Postprocess results and visualization
```

Chaotic Firefly

There are three parameters in the firefly algorithm. Those are α, β and γ, which control the randomness, attractiveness and the modal scales, respectively. For most implementations, we can take $\alpha=O(1)$, $\beta=O$ (1) and $\gamma=O(1)$ [8]. However, randomness reduction technique is often used as iterations continue, and this is often achieved by using an annealing-like exponential function[8, 21].

$$\alpha = \alpha_o \eta^t \qquad \text{…………… (14)}$$

$$\alpha \rightarrow \alpha_o \eta \qquad \text{…………… (15)}$$

Where $0 < \eta < 1$ is a cooling parameter. Typically, we can use $\alpha_o = 1$ and $\eta = 0.9 \approx 0.99$. This equivalently introduces a cooling schedule to the firefly algorithm, as used in the traditional simulated annealing. Recently studies showed this works well (Yang, 2008). There may be better ways to tune this parameter and reduce randomness to be discussed later in this section.

It is worth pointing out that (11) is essentially a random walk which is biased towards the brighter fireflies. If $\beta_o = 0$, it becomes a simple random walk. As it is true for all the metaheuristic algorithms, algorithm-dependent parameters can affect the performance of the algorithm greatly for an algorithm of interest, a natural question is whether we can automatically tune these parameters? If so, what is the best way to fine-tune these parameters?

For randomness reduction, it should be linked with the diversity of the current solutions. One simple way to automatically tune α is to set α as proportional to the standard deviation of the current solutions. However, in case of multimodal problems, this standard deviation should be calculated for each local mode amongst the local subgroups of fireflies. For example, for two modes *A* and *B* with their current best solutions x_a* and x_b* , respectively, the population will gradually subdivide into two main subgroups with population sizes of n_1 and n_2 , respectively, one around *A* and one around *B* . There are two standard deviations σ_A and σ_B which should be calculated among the solutions which are relative to x_a* and x_b*, respectively. Then the overall α should be a function of σ_A and σ_B. The simplest way is to combine them by weighted average

$$\sigma = \frac{(\sigma_A n_1 + \sigma_B n_2)}{n_1 + n_2}, \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (16)$$

$$n_1 + n_2 = n,$$

As iterations continue, σ decreases in general. If we set

$$\alpha = \zeta\sigma, \quad 0 < \zeta < 1 \qquad \ldots\ldots\ldots\ldots\ldots\ldots (17)$$

Then α is automatically associated with the scale of the problem of interest. In practice, η may be affected by the dimensions *d*, so $\zeta = \sqrt{d/(2d+1)}$. The parameter γ should be linked with the scale *L* of the modes. A quite simple rule is that the change of the attractiveness term should be O (1) through the search landscape, which provides a simple relationship $\gamma = 1/\sqrt{L}$. Parameter β controls the behavior of fireflies; however, its tuning is more subtle. From the above discussion of (17), when β is quite large, fireflies may experience a chaotic behavior, and this can be used to effectively enhance the search capability of the algorithm. The algorithm can be expected to converge more quickly. So for the same firefly (5.9), if we reduce β gradually, as the iteration proceeds further, this path will gradually settle down and converge to a global optimal point.

Now we have three distinct versions of FA: The standard version of FA with α as a cooling schedule, a chaos-enhanced FA with β reduced gradually, and the chaotic FA in combination with automatic parameter tuning (AutoFA).

HGA-FF (Hybrid Genetic algorithm and Firefly algorithm)

Another discussed issue of Firefly algorithm is the apparent weakness in exploring search space. In order to deal with this problem, we used the nature of the Genetic algorithm for more desirable global search. Genetic algorithm is suitable in exploring search space and find new better solutions. In this new model we use a co-evolutionary algorithm that each one of them has their own populations. At the end of each iteration they swap their populations. In this newly designed model, GA searches globally and thereafter find better solutions than the last iteration and creates new population. After finishing, the GA firefly algorithm will search locally in the created population of GA. We can list out the steps shown in the figure 1 of new HGA-LL as under[2,5,20]

In model, in GA steps, the new generations which are produced by new algorithm cost better than their parents one; then, this helps next iterations to have better solutions. As mentioned above, we can use different selection methods in this model[24].

```
1. Initialize two populations for genetic and firefly algorithm
2. Run firefly and genetic algorithm concurrently
3. Evaluate new solution
4. Check termination criteria
5. If termination criteria = false
6. Change populations
7. else
8. Go to step 2
9. End if
```

Pseudo code 4- HGA-FF Algorithm

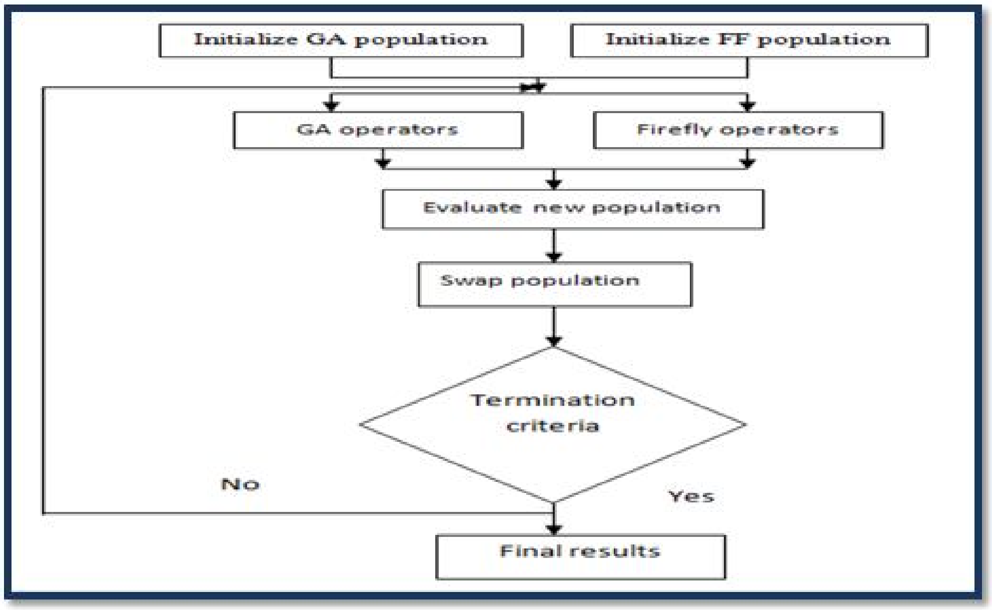


Figure 1. Flow chart for HGA-FFA

CRYPTANALYSIS OF SUBSTITUTION CIPHER

Cryptography is a complex and mathematically challenging study. It has the involvement of taking some data or message and obfuscating it so that it is unreadable by the parties for which it was not intended. The message is referred to as plain text before it is encrypted. After the message has been converted into the encrypted form, it is referred to as cipher text[17].

The study of cipher text while attempting to restore the message to plaintext is known as cryptanalysis. Cryptanalysis is an equally mathematically challenging and complex as cryptography [14, 15]. Each permutation of the 26 letters of the English gives a unique key for encrypting a message. If a particular permutation is useful to encrypt a message, then the inverse of that permutation could be used to decrypt it[15]. Because of the very complexity involved with cryptanalysis work the key space is of size $26! \approx 4.03291461*10^{26} \approx 2^{88}$with an exhaustive search is having a work factor of 2^{87}on average[12,15].

The most direct attack on a cryptosystem is said to be an exhaustive key search attack. The key size therefore provides for a lower bound on the security of a cryptosystem[13, 14]. Substitution ciphers can be then solved by exhaustively searching through the (astronomically large) key space for the very key that produces the decrypted text most closely resembling meaningful English[16].

Considering a simple substitution cipher on the letter A….Z indexed by 1…..26 as above. The key space for this type of system is a set of bijective functions f:1….26→1….26. *Given* cipher text C decryption can be thought of as a function $f_c(K)$ from the key space to the space of plain text messages[14]. Decrypting cipher text using keys that are very much 'nearly the same' giving rise to plain texts that are nearly the same. Similarly, keys that are 'nearly correct' give rise to plain texts that are basically nearly correct. With respect to correctness the decryption operation is reasonably continuous over the key space[14,16].

Frequency Analysis

Given a sample of an English language newspaper text (stripped of spaces, punctuation and other extraneous characters) the following gives the approximate percentage of occurrences of each character. The relative frequencies can change according to subject matter and style of writing but it is still possible to pick out those characters with a high frequency of occurrence and those which are rare[14,15].

Language statistics remain unchanged by the encryption process and hence frequency analysis presents a basic tool for breaking of substitution ciphers. In most languages, some letters of the alphabet appear more frequently than others. The *n*-gram statistics indicates the frequency distribution of all possible instances of *n* adjacent characters[12].

A frequency count could be conducted on a cipher to learn what the most and least common characters are in the cipher. The most common letters that are there in the English language are E, T, N, R, O, A, and I and S. These very eight characters make up around 67% of the words in the English language. Vowels, A,E,I,O, and U which make up around 40% of English text. A pair of letters together is referred to as a *Diagram*. The common digraphs in the English language are, TH, HE, EN, RE and ER. There are also *Trigrams which* consist of frequency of three letters next to each other THE, ING, CON,ENT and ERE[15,17]. If the frequency of the cipher text is actually the same as in the plaintext then the encoded method is actually a transposition cipher instead of a substitution cipher.

Fitness Function

The technique used to compare the candidate keys is to compare the statistics of the decrypted message with those of the language. The letter frequency is then used for attacks against cryptographic cipher. The frequencies of letters that typically occur in natural languages are those which are known and are well documented[13,14].

$$f(key) = \left(1 - \frac{\sum_{i=1}^{26}\left\{|sf[i]-df[i]|+\sum_{j=1}^{26}|sdf[i,j]-ddf[i,j]|\right\}}{4}\right)^{8} \quad \text{........ (18)}$$

The letter A…Z are referenced to by the indices 1….26. Here $sf[i]$ is the standardfrequency of character $df[i]$ in English it is the measured frequency of the character i in the decoded cipher text. Similarly$sdf[i,j]$is the standard frequency of bigram character i followed by character j in English.$ddf[i,j]$ is the corresponding frequency of that bigram of the decoded ciphertext. The really surprising element is the exponent of 8, included to amplify small differences[14].

The fitness function compares unigram and bigram frequencies of characters in the known language with the corresponding frequencies contained by the cipher text. Keys with higher fitness value have more chance of being selected[19].

Algorithm's Used for Cryptanalysis

Random Algorithm

In first search method, given in Algorithm 1, Create n number of Random keys initially and calculate corresponding fitness value for them. For various iterations discard n/4 number of keys of low fitness value and create new n/4 random keys.

```
Cryptanalysis_of_substitution_cipher_1 ()
{
N= no of random keys.
For i=1 to n
   {
   Key[i,:]=generate Random key.
   P=decrypt cipher text using key(i).
   [f[i]]=fitness_value(p);
   }
Sort all fireflies on the basis of f.
For i=1 to No_of_Iteration
   {
   For j=1 to n/4
```

```
                {
                 Key[j,:]=generate Random key.
                 P=decrypt cipher text using key(i).
                 [f[j]]=fitness_value(p);
                  }
        Sort all fireflies on the basis of f.
        }
       Decrypt cipher text using key[n,:].
       }
```

Proposed Firefly Algorithm

In second search method, given in Algorithm 2, is a Metaheuristic search Algorithm over the key space. In this thesis I present new firefly algorithm for cryptanalysis of substitution. Algorithm first generate n random keys as fireflies calculate four attributes as

f:- 1D array n×1 define fitness value,
a:- 2D array n×26 size define location where frequency of sf[i]=df[i],
nv:- 1D array n×1 define number of corrected vowels,
nc:- 1D array n×1 define number of corrected characters in key.
Now keys are sort by nv. Then high nv value key attract low nv value key by using below algorithm.

```
Cryptanalysis_of_substitution_cipher_2()
{
N= no of fireflies.
For i=1 to n
   {
   Key[i,:]=generate Random key.
   P=decrypt cipher text using key(i).
   (f[i],a[i,:],nv[i],nc[i])=fitness_value(p);
   }
   Sort all fireflies on the basis of nv.
   For i=1 to No_of_Iteration
{
For j=1 to n-1
  {
  If(f[n]<f[j])
    {
    For k=1 to 26
      {
      If(a[n,k]==1)
        {
        Loc=find location of char key[n,k] in key[j,:]
        Swap key[j,k]with key[j,loc]
        }
      }
```

```
          }
        }
    Sort all fireflies on the basis of nv.
    For i=1 to n/4
        {
        Key[i,:]=generate Random key.
        P=decrypt cipher text using key(i).
        [f[i],a[i,:],nv[i],nc[i]]=fitness_value(p);
        }
    }
    Decrypt cipher text using key[n,:].
    }
```

IMPLEMENTATION AND RESULTS

In this work, for the computational procedures described above a computer simulation program was implemented in a Matlab 7.9.0(R2009b) computer program. A Laptop computer Intel core 2 Duo was used for computational experiments throughout. The cryptanalysis of substitution cipher following steps are

Step 1:- Let us take plain text described below used to obtained results for cryptanalysis of substitution cipher.

Plain text =

"REVENUE CANNIBALIZATION RESULTING FROM CLIENT ADOPTION OF INDUSTRIALIZED AND OFTEN CLOUD BASED SERVICES RISKS MUTING THE GROWTH OPPORTUNITIES FOR THE ITO PROVIDERS THAT ARE HEAVILY WEIGHTED IN INFRASTRUCTURE OUTSOURCING SAID BRYAN BRITZ RESEARCH DIRECTOR AT GARTNER FORTY THREE PROVIDERS RECORDED REVENUE OF BILLION OR MORE THIS GROUP OF PROVIDERS COLLECTIVELY GREW BY IN AFTER EXCLUDING INDIA BASED IT SERVICES PROVIDERS CLOUD CENTRIC PROVIDERS AND PROVIDERS THAT MADE SIZABLE ACQUISITIONS DURING THE YEAR THE REMAINING GROUP OF LARGE ITO PROVIDERS GREW BY ONLY DURING SAID GARTNER IN THE RELEASE"

Step 2:- Generate key by using random_key() function

Key=

Table 1. Key used to create cipher text

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

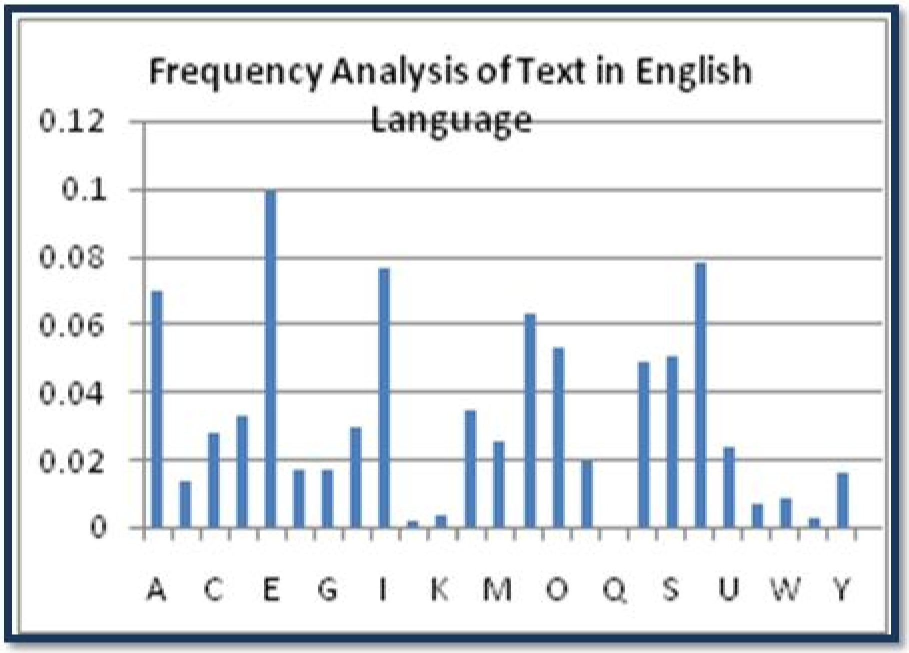
Step 3:- Cipher text is created by applying above key on the plain text

Cipher text =

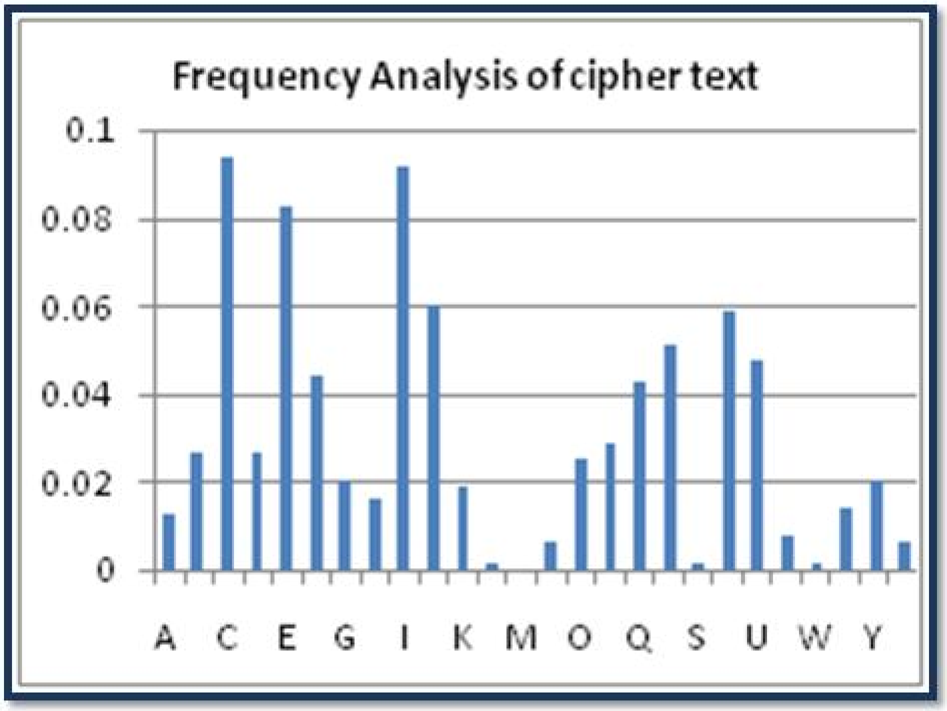
ICGCRPC DURREXUBEZUJETR ICFPBJERO HITV DBECRJ UQTKJETR TH ERQPFJIEUBEZCQ URQ THJCR DBTPQ XUFCQ FCIGEDCF IEFLF VPJERO JYC OITNJY TKKTIJPREJECF HTI JYC EJT KITGEQCIF JYUJ UIC YCUGEBA NCEOYJCQ ER ERHIUFJIPDJPIC TPJFTPIDERO FUEQ XIAUR XIEJZ ICFCUIDY QEICDJTI UJ OUIJRCI HTIJA JYICC KITGEQCIF ICDTIQCQ ICGCRPC TH XEBBETR TI VTIC JYEF OITPK TH KITGEQCIF DTBBCDJEGCBA OICN XA ER UHJCI CWDBPQERO ERQEU XUFCQ EJ FCIGEDCF KITGEQCIF DBTPQ DCRJIED KITGEQCIF URQ KITGEQCIF JYUJ VUQC FEZUXBC UDSPEFEJETRF QPIERO JYC ACUI JYC ICVUERERO OITPK TH BUIOC EJT KITGEQCIF OICN XA TRBA QPIERO FUEQ OUIJRCI ER JYC ICBCUFC

Step 4:- Frequency analysis of plain text and cipher text

A frequency count can be conducted on sample of an English language newspaper text to learn what the most and least common characters are in the cipher. E, T, I, A, N, R, O and S are the most common letters used in the English language as shown in figure 2.



(i)



(ii)

Figure 2. frequency analyses of (i) Text in English language (ii) cipher text

The figure 2(i) shows the frequency of each character in the plain text of English language i.e. unigrams which is stored in c one dimensional array. The figure 2(ii) shows the frequency of cipher text decrypted by random key defined in step 2. Similarly frequency of bigram calculated and store in two dimensional array.

Step 5:- Final Result by Random Algorithm

Final result given by Matlab code presented below result contains key used to decrypt cipher text, decrypted text by the random algorithm, corrected percentage of decrypted text and a graph shown in figure 3 mean of fitness value for 100 keys against number of iteration. Firstly best key recover by random algorithm used to decrypt text shown below.

Table 2. Key Recovered by Random Algorithm to decrypt cipher text

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

Decrypted text shown below:-

"REBEUNE JAUUIQAYIZAWIVU REHNYWIUT XRVK JYIEUW ASVPWIVU VX IUSNHWRIAYIZES AUS VXWEU JYVNS QAHES

HERBIJEH RIHDH KNWIUT WCE TRVFWC VPPVRWNUIWIEH XVR WCE IWV PRVBISERH WCAW ARE CEABIYL FEITCWES IU IUXRAHWRNJWNRE VNWHVNRJIUT HAIS QRLAU QRIWZ REHEARJC SIREJWVR AW TARWUER XVRWL WCREE PRVBISERH REJVRSES REBEUNE VX QIYYIVU VR KVRE WCIH TRVNP VX PRVBISERH JVYYEJWIBEYL TREF QL IU AXWER EMJYNSIUT IUSIA QAHES IW HERBIJEH PRVBISERH JYVNS JEUWRIJ PRVBISERH AUS PRVBISERH WCAW KASE HIZAQYE AJGNIHIWIVUH SNRIUT WCE LEAR WCE REKAIUIUT TRVNP VX YARTE IWV PRVBISERH TREF QL VUYL SNRIUT HAIS TARWUER IU WCE REYEAHE"

The above decrypted text is compared by original plain text and then calculates corrected characters percentage of decrypted text which is 32.3296 %.

Take mean of fitness value for 100 keys in every iteration and then take plot shown below in figure 3 which describe the performance of Random algorithm based on number of iteration. Graph shown in the figure 3 shows after 3000 iteration no improvement in the mean of fitness value, it means Algorithm converges after 3000 iteration.

Step 6:- Final Result by Firefly Algorithm

Final result given by Matlab code described in section 4.3.7.2 of M.Tech Thesis is presented below result contains key used to decrypt cipher text, decrypted text by the random algorithm, corrected percentage of decrypted text and a graph shown in figure 4 mean of fitness value for 100 keys against number of iteration. Firstly best key recover by Firefly algorithm used to decrypt text shown below.

Table 3. Key Recovered by Firefly Algorithm to decrypt cipher text

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

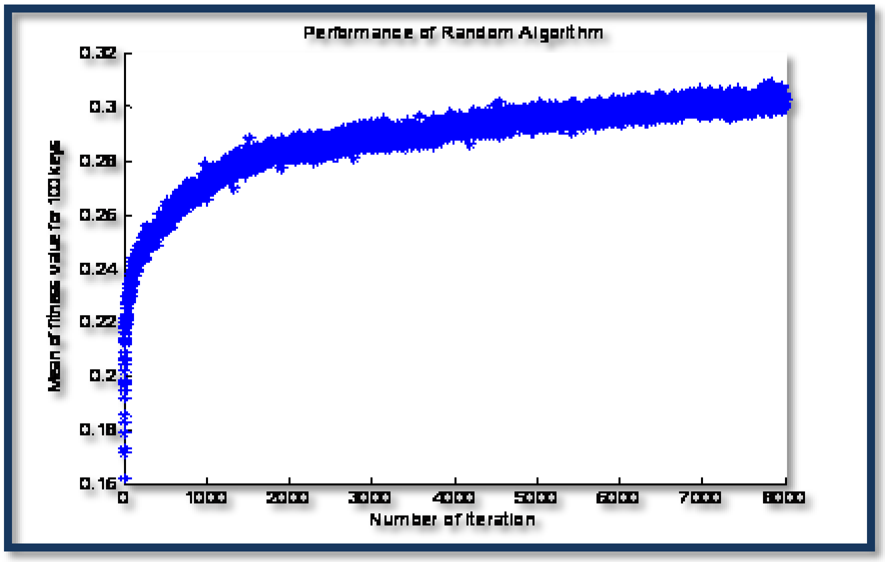


Figure 3. Shows growth of mean of fitness value against Number of iteration by Random Algorithm

Decrypted text shown below:-

"REKEGUE LAGGFVAXFWATFMG RENUXTFGP IRMC LXFEGT ADMJTFMG MI FGDUNTRFAXFWED AGD MITEG LXMUD VANED NERKFLEN RFNON CUTFGP THE PRMBTH MJJMRTUGFTFEN IMR THE FTM JRMKFDERN THAT ARE HEAKFXZ BEFPHTED FG FGIRANTRULTURE MUTNMURLFGP NAFD VRZAG VRFTW RENEARLH DFRELTMR AT PARTGER IMRTZ THREE JRMKFDERN RELMRDED REKEGUE MI VFXXFMG MR CMRE THFN PRMUJ MI JRMKFDERN LMXXELTFKEXZ PREB VZ FG AITER ESLXUDFGP FGDFA VANED FT NERKFLEN JRMKFDERN LXMUD LEGTRFL JRMKFDERN AGD JRMKFDERN THAT CADE NFWAVXE ALQUFNFTFMGN DURFGP THE ZEAR THE RECAFGFGP PRMUJ MI XARPE FTM JRMKFDERN PREB VZ MGXZ DURFGP NAFD PARTGER FG THE REXEANE "

The above decrypted text is compared by original plain text and then calculates corrected characters percentage of decrypted text which is 38.6688 %.

Take mean of fitness value for 100 keys in every iteration and then take plot shown below in figure 4 which describe the performance of Firefly algorithm based on number of iteration. Graph shown in the figure 4 shows after 450 iterations no improvement in the mean of fitness value, it means Algorithm converges after 450 iterations.

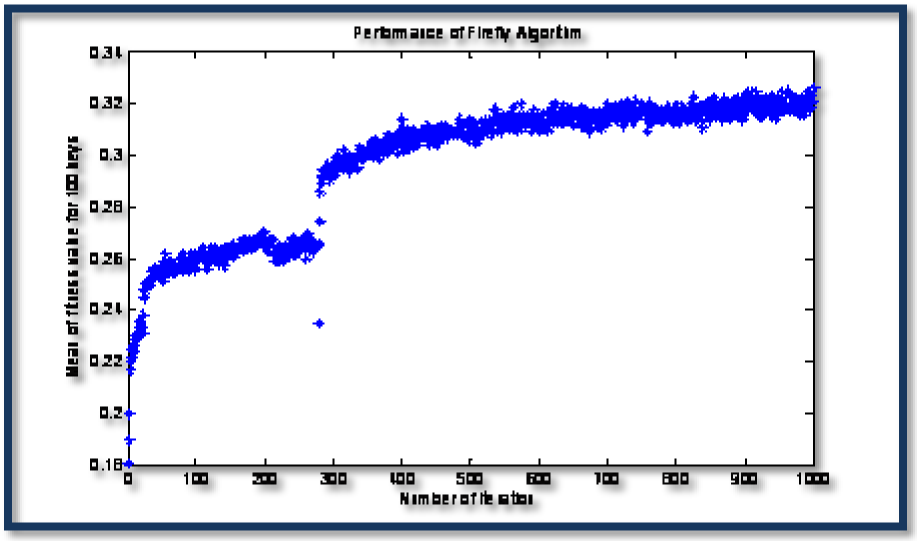


Figure 4. Shows growth of mean of fitness value against Number of iteration by Firefly Algorithm

CONCLUSION AND FUTURE WORK

As shown in the above section two algorithms used for cryptanalysis of substitution cipher, Random Algorithm converge after 3000 iterations and Firefly Algorithm converges after 450 iterations shown in figure 3 and 4 respectively. It shows improvement in convergence time. Percentage for corrected decrypted text by Random Algorithm is 32.3296 % and by Firefly Algorithm is 38.6688 %; it shows Firefly Algorithm is better in the terms of percentage of corrected decrypted text.

Further we can apply variants of firefly such as Gaussian Firefly, Chaotic Firefly, and Hybridization with Genetic Algorithm (described in next chapter) for the improvement on the performance and increase percentage of corrected decrypted text.

References

1. X. S. Yang, “Nature-Inspired Metaheuristic Algorithms”, Luniver Press, 2008.
2. Christian Blum, Maria Jos´e Blesa Aguilera, Andrea Roli, Michael Sampels, Hybrid Metaheuristics, An Emerging Approach to Optimization, Springer, 2008 .
3. WengKee Wong, Nature-Inspired Metaheuristic Algorithms for Generating Optimal Experimental Designs.
4. Saibal K. Pal, C.S Rai, Amrit Pal Singh “Comparative Study of Firefly Algorithm and Particle Swarm Optimization for Noisy Non-Linear Optimization Problems” I.J. Intelligent Systems and Applications, 2012, 10, 50-57 MECS.
5. Jitin Luthra, Saibal K. Pal “A Hybrid Firefly Algorithm using Genetic Operators for the Cryptanalysis of a Monoalphabetic Substitution Cipher” *World Congress on Information and Communication Technologies* 2011 IEEE.
6. Sh. M. Farahani, A. A. Abshouri, B. Nasiri, and M. R. Meybodi, “A Gaussian Firefly Algorithm”, *International Journal of Machine Learning and Computing*, Vol. 1, No. December 2011.
7. Hajo Broersma “Application of the Firefly Algorithm for Solving the Economic Emissions Load Dispatch Problem,” *International Journal of Combinatorics*, Volume 2011.
8. *Xin-She Yang*, Chaos-Enhanced Firefly Algorithm with Automatic Parameter Tuning, *International Journal of Swarm Intelligence Research*, December 2011.

9. S. S. Omran, A. S. Al-Khalid, D. M. Al-Saady “Using Genetic Algorithm To Break A Mono - Alphabetic Substitution Cipher” 2010 *IEEE Conference* on Open Systems (ICOS 2010), December 5-7, 2010
10. Xiang-yin Meng, Yu-long Hu, Yuan-hang Hou, Wen-quan Wang, The Analysis of Chaotic Particle Swarm Optimization and the Application in Preliminary Design of Ship”, *International Conference on Mechatronics and Automation*, August, 2010.
11. Michael James Banks “A Search-Based Tool for the Automated Cryptanalysis of Classical Ciphers” The University of York Department of Computer Science May, 2008
12. Mohammad Faisal Uddin and Amr M. Youssef” Cryptanalysis of Simple Substitution Ciphers Using Particle Swarm Optimization” 2006 *IEEE* Congress on Evolutionary Computation July 16-21, 2006
13. Mohammad Faisal Uddint, Amr M. Youssefτ “An Artificial Life Technique for the Cryptanalysis of Simple Substitution Ciphers” *IEEE CCECE/CCGEI*, Ottawa, May 2006.
14. John Andrew Clark “Metaheuristic Search as a Cryptological Tool” University of York Department of Computer Science December, 2001
15. Sam Hasinoff “Solving Substitution Ciphers” Department of Computer Science, University of Toronto
16. Thomas Jakobsen “A Fast Method for cryptanalysis of substitution cipher” January, 1995
17. Craig Smith “Basic Cryptanalysis Techniques” November 17th, 2001
18. Amrapali Dhavare, Richard M. Low, Mark Stamp “E_cient Cryptanalysis of Homophonic Substitution Ciphers” Department of Computer Science, San Jose State University
19. Badrisham bin Ahmad and Mohd Aizaini bin Maarof “Cryptanalysis using Biological Inspired Computing Approaches” Annual Research Seminar 2006
20. SH. MASHHADI FARAHANI, A. AMIN ABSHOURI, B. NASIRI, M. R. MEYBODI “Some hybrid models to improve Firefly algorithm performance”
21. Ying Song, Zengqiang Chen, and Zhuzhi Yuan “New Chaotic PSO-Based Neural Network Predictive Control for Nonlinear Process” *IEEE TRANSACTIONS ON NEURAL NETWORKS*, VOL. 18, NO 2, MARCH 2007.