International Journal of Modern Engineering Research (IJMER)

www.ijmer.com Vol.2, Issue.3, May-June 2012 pp-949-954 ISSN: 2249-6645

Heuristic Search Procedures for Cryptanalysis and Development of Enhanced Cryptographic Techniques

Rajashekarappa[1], Dr. K M S Soyjaudah[2]

*(Dept. of Computer Science and Engineering, JSS Academy of Technical Education Mauritius, Avenue Droopnath Ramphul, Bonne Terre, Vacoas, Mauritius)

** (Dept. of Electrical and Electronic Engineering, University of Mauritius, Reduit, Mauritius)

ABSTRACT

Cryptology is a thriving research area of great practical importance. It is a fundamental building block of communications security. Cryptographic techniques currently being utilized in the field of communication. This paper presents an approach for the heuristic search procedures for cryptanalysis and development of enhanced cryptographic techniques. To implement the proposed Tabu search, Genetic, and Simulated Annealing algorithms firstly by utilising cipher text as well as some plain text and secondly by using only the cipher text to retrieve the original data. The cryptanalysis of simplified data encryption standard can be formulated as NP-Hard combinatorial problem. The goal of this paper is to comparison between Tabu Search, Genetic Algorithm and simulated annealing were made in order to investigate the performance for the cryptanalysis on SDES. The methods were tested and extensive computational results show that Tabu Search algorithm performs better than Genetic algorithm and simulated annealing for such type of NP-Hard combinatorial problem.

Keywords: **Genetic Algorithm, Key search space, Simplified data encryption standard, Simulated annealing, Tabu Search algorithm.**

I. INTRODUCTION

Cryptography is the science of hiding information. It is now a part of the computer science formally, though first cryptographers appeared thousands years before the computer. The art of recovery of the hidden information, or cryptanalysis, appeared in the very beginning, and is still one of the most intriguing part of cryptography.

Cryptanalysis starts with a search for a weakness in a cryptosystem, for a flaw that was missed by its designer. An encrypted message must not reveal any information about its origin, so the cryptosystem must make it look as random as possible. Any mistake, any missed property may become a target for a cryptanalyst and a starting point for a compromise of the cryptosystem's security a break.

This survey is devoted to the cryptanalysis of symmetric primitives. Historically, by a symmetric encryption we understand that all the parties have the same information needed for encryption and decryption, with block and stream ciphers as the most famous examples. A block cipher transforms a large block of data with an algorithm parameterized by a secret key. A stream cipher expands a secret key into arbitrarily long sequence, which is mixed with a data stream. Cryptanalysis is the art of analyzing ciphertext to extract the plaintext or the key. In other words, cryptanalysis is the opposite of cryptography. It is the breaking of ciphers. Understanding the process of code breaking is very important when designing any encryption system. The science of cryptography has kept up with the technological explosion of the last half of the 20th century. Current systems require very powerful computer systems to break the code or cryptanalyse most ciphers. While cryptanalysis has improved as well, some systems may exist that are unbreakable by today's standards.

The substitution ciphers are easy to break. Before computers were available, expert cryptanalysts would look at ciphertext and make guesses as to which letters were substituted for which other letters. Early cryptanalysis techniques included computing the frequency with which letters occur in the language that is being intercepted. For example, in the English language, the letters e, s, t, a, m, and n occur much more frequently than do q, z, x, y, and w. So, cryptanalysts look at the ciphertext for the most frequently occurring letters and assign them as candidates to be e, s, t, a, m, and n. Cryptanalysts also know that certain combinations of letters are more common in the English language than others are. For example, q and u occur together, and so do t and h. The frequency and combinations of letters help cryptanalysts build a table of possible solution letters. The more cipher text that is available, the better the chances of breaking the code.

Of the different categories of attacks such as ciphertext only attack, known plaintext, chosen plaintext, chosen ciphertext; ciphertext only attack is a harder one and thus we consider one such attack in this paper.

The objective of the study is to determine the efficiency and accuracy of Tabu Search algorithm for the cryptanalysis of SDES[6]. To compare the relative performance of Genetic algorithm, Simulated Annealing with tabu search.

The rest of the paper is organized as follows: Section 2 presents the literature review. Section 3 gives a brief overview of S-DES, Section 4 gives the overview of Tabu Search and Section 5 gives the algorithm of Simulated Annealing and Genetic Algorithm. Experimental results are discussed in Section 6. Section 7 concludes the paper and Future works.

II. RELATED WORK

The proposed work will require an in depth understanding of the area of cryptography and enable the development of general as well as specific algorithms for cryptanalysis[1]. Moreover, the enciphering algorithms developed in this work will find many real time applications in military, banking and other sectors where secure transmission is essential. A cipher takes a message text and some secret keying data (known as the key) as its input and produces an encrypted version of the original message, (known as the cipher text). An attack on a cipher can make use of the cipher text alone or it can make use of some plaintext and its corresponding cipher text (referred to as a known plaintext attack) (Andrew John Clark, 1998).
Cryptanalysis is the process of recovering the plaintext and/or key from a cipher. Many cryptographic systems have a finite key space and, hence, are vulnerable to an exhaustive key search attack. Yet, these systems remain secure from such an attack because the size of the key space is such that the time and resources required for a search are prohibitive.

A random search through a finite but large key space is not usually an acceptable cryptanalysts tool. Massoudi et al. (2008) explored the possibility of using a random type search to break a cipher. The focus of their work was on the use of a genetic algorithm to conduct a directed random search of a key space[2]. The ability to add direction to what seems to be a random search is a feature of genetic algorithms which suggests that it may be possible to conduct an efficient search of a large key space (Ayman M. B. Albassal, Abdel-Moneim A. Wahdan, 2004). In fact, carefully tailored genetic algorithms can find efficient distinguishers for ciphers much fasgter than previously reported techniques (Aaron, G et.al (2007), J. C. Hernandez, J.C and Isasi, P (2004))[3][4].
An interesting survey of the application of evolutionary computation algorithms such as Genetic algorithm, simulated annealing and tabu search to provide a robust and efficient methodology for cryptanalysis, was conducted by (Garg, P. (2010))[5].

III. THE S-DES ALGORITHM

This section briefly gives the overview of S-DES Algorithm. The SDES encryption algorithm takes an 8-bit block of plaintext and a 10-bit key as input and produces an 8-bit block of ciphertext as output. The decryption algorithm takes an 8-bit block of ciphertext and the same 10-bit key used for encryption as input and produces the original 8-bit block of plaintext as output. The encryption algorithm uses five basic functions: 1. An initial permutation (IP). 2. A complex function called fK which involves both permutation and substitution operations and depends on a key input 3. A simple permutation function (SW) that switches the two halves of the data. 4. The function fK a TS in and 5. A permutation function that is the inverse of the initial permutation (IP^{-1}).The function fK takes as input the data passing through the encryption algorithm and an 8-bit key[9].

A. Key Generation

For key generation, a 10-bit key is considered from which two 8-bit sub keys are generated. In this case, the Key is first subjected to a permutation P10= [3 5 2 7 4 10 1 9 8 6], then a shift operation is performed. The numbers in the array represent the value of that bit in the original10-bit key. The output of the shift operation then passes through a permutation function that produces an 8-bit output P8 =[6 3 7 4 8 5 10 9] for the first sub key (K1). The output of the shift operation also feeds into another shift operation and another instance of P8 to produce the second sub key K2. In all bit strings, the leftmost position corresponds to the first bit. The block schematic of the S-DES Key generation algorithm is shown in Fig. 1.

B. Encryption Algorithm

The block schematic of the SDES encryption algorithm is shown in Fig. 2. The Encryption process involves the sequential application of five functions:
1. Initial and final permutation (IP): The input to the algorithm is an 8-bit block of plaintext, which is first permuted using the IP function IP = [2 6 3 1 4 8 5 7]. This retains all 8-bits of the plaintext but mixes them up. At the end of the algorithm, the inverse permutation is applied; the inverse permutation is done by applying, IP^{-1} = [4 1 3 5 7 2 8 6] Where, $IP^{-1}(IP(X)) = X$.

2. Function fK:
The function fk, which is the complex component of S-DES, consists of a combination of permutation and substitution functions. The functions are given as follows:
fK (L, R) = (L XOR f(R, key), R)where, L, R be the left 4-bits and right 4-bits of the input, XOR is the exclusive-OR operation and key is a sub -key. Computation of f(R, key) is done as follows.

i. Apply expansion/permutation
E/P= [4 1 2 3 2 3 4 1] to input 4-bits.

ii. Add the 8-bit key (XOR).

iii. Pass the left 4-bits through S-Box S0 and the right4-bits through S-Box S1.

iv. Apply permutation P4 = [2 4 3 1].
The two S-boxes are defined as follows:

S0	S1
1032	0123
3210	2013
0213	3010
3132	2103

The S-boxes operate as follows: The first and fourth input bits are treated as 2-bit numbers that specify a row of the S-box and the second and third input bits specify a column of the S box. The entry in that row and column in base 2 is the 2-bit output.

3. The Switch Function (SW):
Since the function fK allows only the leftmost 4-bits of the input, the switch function (SW) interchanges the left and right 4-bits so that the second instance of fK operates on different 4-bits. In this second instance, the E/P, S0,S1 and P4 functions are the same as above but the key input is K2[11].

IV. TABU SEARCH

Tabu search is a widely used meta heuristic that uses some common-sense ideas to enable the search process to escape from a local optimum[8].
Any application of tabu search includes as a subroutine a local search procedure that seems appropriate for the problem being addressed. A local search procedure operates just like a local improvement procedure except that it may not require that each new trial solution must be better than the preceding trial solution. The process begins by using this procedure as a local improvement procedure in the usual way (i.e, only accepting an improved solution at each iteration) to find a local optimum. A key strategy of tabu search is that it then continues the search by allowing non-improving moves to the best solutions in the neighborhood of the local optimum[8]. Once a point is reached where better solutions can be found in the neighborhood of the current trial solution, the local improvement procedure is reapplied to find a new local optimum.
This use of memory to guide the search by using tabu lists to record some of the recent history of the search is a distinctive feature of tabu search. This feature has roots in the field of artificial intelligence[12].
Long term memory is used to help implement both concepts. However, we will focus on the basic form of tabu search summarized below without delving into these additional concepts [9].

Outline of a Basic Tabu Search Algorithm
Initialization: Start with a feasible initial trial solution.
Iteration: Use an appropriate local search procedure to define the feasible moves into the local neighborhood of the current trial solution. Eliminate from consideration any move on the current tabu list unless that move would result in a better solution than the best trial solution found so far. Determine which of the remaining moves provides the best solution. Adopt this solution as the next trial solution, regardless of whether it is better or worse than the current trial solution. Update the tabu list to forbid cycling back to what had been the current trial solution. If the tabu list already had been full, delete the oldest member of the tabu list to provide more flexibility for future moves[10].

Cost function: The ability of directing the random search process of the tabu search by selecting the fittest chromosomes among the population is the main characteristic of the algorithm. So the fitness function is the main factor of the algorithm. The choice of fitness measure depends entirely on the language characteristics must be known. The tabu search technique used to find candidate key is to compare n-gram statistics of the decrypted message with those of the language(which are assumed known). Equation 1 is a general formula used to determine the suitability of a proposed key(k), here, K is known as language Statistics i.e., for English, [A,…….,Z],D is the decrypted message statistics, and u/b/t are the unigram, bigram and trigram statistics. The values of α, β and γ allow assigning of different weights to each of the three n-gram types where α+ β + γ =1.

$$C_k \approx \alpha . \sum_{i \epsilon A} \left| K^u{}_{(i)} - D^u{}_{(i)} \right| + \beta . \sum_{i,j \epsilon A} \left| K^b{}_{(i,j)} - D^b{}_{(i,j)} \right| + \gamma . \sum_{i,j,k \epsilon A} \left| K^t{}_{(i,j,k)} - D^t{}_{(i,j,k)} \right| - - - (1)$$

When trigram statistics are used, the complexity of equation(1) is O (P3) where P is the alphabet size. So it is an expensive task to calculate the trigram statistics. Hence we will use assessment function based on bigram statistics only. Equation 1 is used as fitness function for tabu search attack. The known language statistics are available in the literature [7].

The tabu search [7] prevents the search from returning to a previously explored region of the solution space too quickly. This is achieved by retaining a list of possible solutions that have been previously encountered. These solutions are called ‘tabu’; hence the name of the technique. The size of the tabu list influences the performance of the algorithm. In each iteration, the best new key formed replaces the worst existing one in the tabu list.

The algorithm is presented as below.
1. Input: Intercepted ciphertext, the key size P, and the language statistics.

2. Initialise parameters: The size of the tabu list STABU, the size of the list of possibilities considered in each iteration SPOSS, and the maximum number of iterations MAX.
3. Initialise: The tabu list with random and distinct keys and calculate the cost for each key in the tabu list.
4. For I =1,… , MAX do:
a. Find the best key with the lowest cost in the current tabulist, KBEST.

b. For j=1,…, SPOSS do:
i. apply the perturbation mechanism to produce a new keyKNEW.
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 thisiteration.
c. From the list of possibilities for this iteration, find the keywith the lowest cost, PBEST.
d. From the tabu list, find the key with the highestcost, 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).

An important distinction in TS arises by differentiating between short term memory and longer term memory. Each type of memory is accompanied by its own special strategies. However, the effect of both types of memory may be viewed as modifying the neighborhood **N**(*x*) of the current solution *x*. The modified neighborhood, which we denote by **N***(*x*), is the result of maintaining a selective history of the states encountered during the search[13].

In the TS strategies based on short term considerations, **N***(*x*) characteristically is a subset of **N**(*x*), and the tabu classification serves to identify elements of **N**(*x*) excluded from **N***(*x*). In TS strategies that include longer term considerations, **N***(*x*) may also be expanded to include solutions not ordinarily found in **N**(*x*). Characterized in this way, TS may be viewed as a dynamic neighborhood method. Characteristically, a TS process based strictly on short term strategies may allow a solution *x* to be visited more than once, but it is likely that the corresponding reduced neighborhood **N***(*x*) will be different each time.

Simulated Annealing

Annealing is the process of slowly cooling a heated metal in order to attain a minimum energy state. The idea of mimicking the annealing process has been efficiently exploited by Kirkpatrick et al.[14] to solve combinatorial optimization problems. The algorithm is initialized with a random solution to the problem being solved and a starting temperature T0. The temperature is slowly decreased and at each temperature, a number of attempts are made to perturb the current solution. At each perturbed temperature, a change in the cost function ΔE is determined. If ΔE<0, then the proposed perturbation is accepted; otherwise it is accepted with a probability indicated by the Metropolis equation given by Probability (E1→E2) = $e^{(-\Delta E/T)}$, where E1 and E2 are the cost functions, ΔE is the change in cost function and T is the current temperature. If the proposed change is accepted, then the current solution is updated. The temperature is reduced when a predefined number of attempts have been made to update the current solution. Possibilities of termination are when a certain minimum temperature is reached or a certain number of temperature reductions have occurred; or the current solution has not changed for a number of iterations. The Simulated Annealing algorithm is presented as below:

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 (eg., 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 will be accepted. If accepted, update the current solution and its associated cost. If the number of accepted transitions for temperature T exceeds some limit (eg. 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 (eg. $T^{(i+1)=} T^{i} \times 0.95$ and return to step 4.

Genetic Algorithm for Cryptanalysis:

A genetic algorithm [4] is a search heuristic inspired by biological evolution. The basic GA algorithm involves the generation of a population of possible solutions, evaluation of the solutions according to a fitness function, selection of a set of fit "parent" solutions, and finally reproduction of those parents to generate a new population of possible solutions.

1. Input: Intercepted ciphertext, and the language statistics.
2. Initialise the algorithm parameters: the solution pool size M and the maximum number
of iterations MAX.
3. Randomly generate an initial pool of solutions P_{CURR}, and calculate the cost of each of the solutions in the pool.
4. For I =1… MAX do:
   a. Select M/2 pairs of keys from P_{CURR} to be the parents of the new generation.
   b. Perform the mating operation on each of the pairs of parents to produce a new pool of solutions $P_{NEW.}$
   c. For each of the M children, perform a mutation operation.
   d. Calculate the cost associated with each of the keys in the new solution pool P_{NEW}.
   e. Sort P_{NEW} from the most suitable (the least cost) to the least suitable (the most cost).
   f. Merge P_{CURR} with P_{NEW} to give a list of sorted solutions (discard duplicates). Choose the best M keys to become the new current pool $P_{CURR.}$
5. Output the best solution from $P_{CURR.}$

V. Experimental Setup and Results

Number of experiments is carried out to outline the effectiveness of Tabu Search. The Tabu Search algorithm is coded in MATLAB 7, and tested on more than 100 benchmark data sets adapted. Among the unigrams, bigrams and trigrams, Unigram is more useful and the benefit of trigram over digram is small.

Table. 1 shows the Comparative Results of Genetic Algorithm, Simulated Annealing and Tabu Search. In this section a number of experiments are carried out which outlines the effectiveness of all three algorithm described above. The purpose of these experiments is to compare the performance of Genetic algorithm, Simulated Annealing algorithm approach with tabu search approach for the cryptanalysis of simplified SDES algorithm. The experiments were implemented in MATLAB 7 on a Pentium IV(1.83 Ghtz). Experimental results obtained from these algorithms were generated with 200 runs per data point e.g. twenty different messages were created for all the algorithms and each algorithm was run 80 times per message. The best result for each message was averaged to produce data point.

This table 1 shows the average number of key elements (out of 10) correctly recovered versus the amount of cipher text and the computation time to recover the keys from the search space. The table shows results for amounts of cipher text ranging from 100 to 1000 character.

VI. Conclusion and Future Work

Heuristic-based optimisation techniques have been finding recent application in attack of ciphers. This paper presents for the first time such a study for the attack of block ciphers. Tabu Search for the cryptanalysis of Simplified Data Encryption Standard is presented. The time complexity of the proposed approach has been reduced drastically when compared to the Genetic Algorithm, and Simulated Annealing Algorithm. Though SDES is a simple encryption algorithm, this is a promising method and can be adopted to handle other complex block ciphers like DES and AES. The cost function values used here can be applied for other block ciphers also. A simple heuristic for the attack of known ciphertext with a pair of them has been developed to retrieve the plaintext to almost 90% accuracy for a SDES 10-bit key. It is envisaged that the results reported in this paper will be useful for the attack of other stream and block ciphers; and also in solving a class of heuristic-based optimisation problems.

The second comparison was made upon the period of transposition cipher. It was found that the tabu search is most powerful to find the correct solution. Result indicates that tabu search is extremely powerful technique for attacking SDES. The future works are extending this approach for attacking DES and AES ciphers

References

[1] Andrew John Clark, *Optimisation Heuristics for Cryptology* , PhD thesis, Information Security Research Centre Faculty of Information Technology Queensland University of Technology,1998.

[2] Floreano D., Durr P., and Mattiussi C., *Neuroevolution: From architectures to learning, Evolutionary Intelligence*, 1:47–62, 2008.

[3] Ayman M. B. Albassal, Abdel-Moneim A. Wahdan, *Genetic Algorithm Cryptanalysis of a Fiestel Type Block Cipher*, International Conference on Electrical, Electronic and Computer Engineering, Egypt, PP. 217–221, 2004.

[4] Aaron, G., Hamilton, J and Dozier, G, *A Comparison of Genetic Algorithm Techniques for the Cryptanalysis of TEA*, International Journal of Intelligent Control and Systems, *12(4)* pp. *325-330*, 2007.

[5] Garg, P, *Evolutionary Computation Algorithms for Cryptanalysis: A Study, (IJCSIS) International Journal of Computer Science and Information Security,Vol. 7,* No. *1,* 2010.

[6] Poonam Garg, Crypatanalysis of SDES via Evolutionary Computation Techniques, *Inter National Journal of Computer Science and Information Security* Vol.*1*, No.*1*, May 2009J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. *2*. Oxford: Clarendon, 1892, pp.68–73.

[7] Glover F, Tabu Search — Part I, *ORSA Journal on Computing 2*: *1*, 4-32, 1990.

[8] Frederick S. Hillier, Gerald J. Lieberman, *Introduction to Operations Research Concepts and Cases* (Eightedition,McGraw- Hill, 2009).

[9] William Stallings, *Cryptography and Network Security Principles and Practices* (Fourth edition, McGraw-Hill, 2003.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989).

[10] Behrouz A. Forouzan, *Cryptography and Network Security* (Firstedition, McGraw- Hill, 2006).

[11] James Kennedy and Russell Eberhart, Particle Swarm Optimisation, *Proceedings of the IEEE International Conference on Neural Networks*, pp.1942-1948, 1995.

[12] Laguna M. J. W. Barnes and F. Glover, Intelligent Scheduling with Tabu Search: An Application to Jobs with Linear Delay Penalties and Sequence Dependent Setup Costs and Times, *Journal of Applied Intelligence* Vol.*3*, pp.159-172,1993.

[13] Chanas S. and P. Kobylanski, A New Heuristic Algorithm Solving the Linear Ordering Problem, *Computational Optimization and Applications*, Vol. *6*, pp. 191-205, 1996.

[14] Garg Poonam, Shastri Aditya, Agarwal D. C, Genetic Algorithm, Tabu Search & Simulated annealing Attack on Transposition Cipher, *proceeding of Third AIMS International conference on management at IIMA* – 2006, pg 983-989.

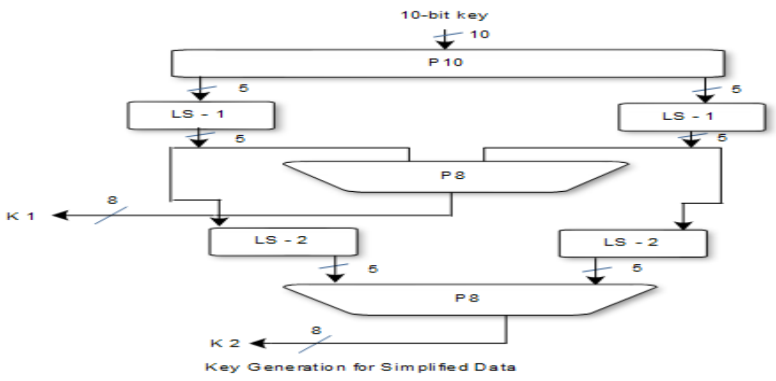

Fig. 1Key Generation

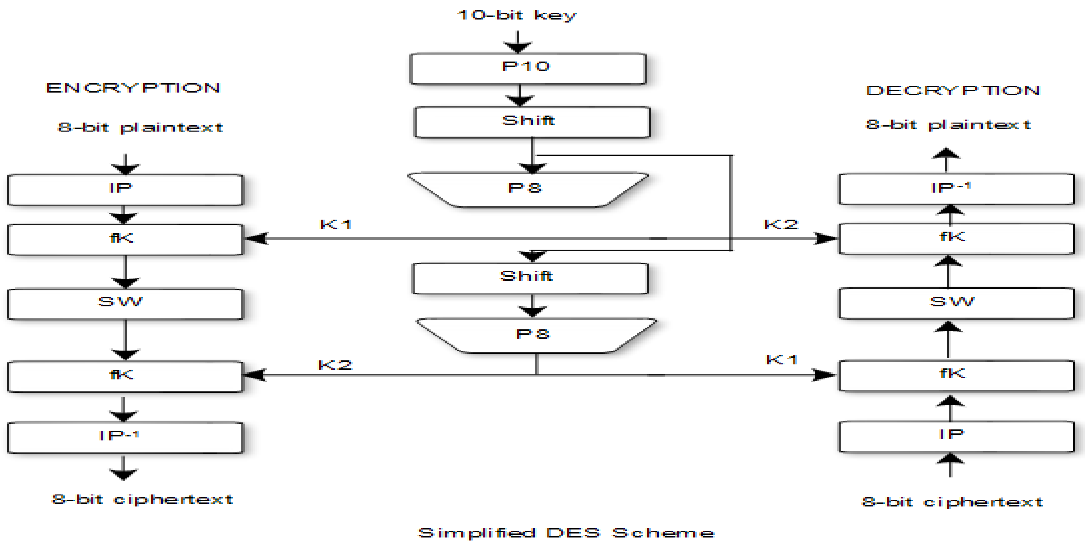

Fig. 2 S-DES Encryption.

Amount of Cipertext	Genetic Algorithm		Simulated Annealing		Tabu Search	
	Time(Minute)	Number of bits matched in the Key	Time(Minute)	Number of bits matched in the Key	Time(Minute)	Number of bits matched in the Key
100	30	3	20.9	4	7.8	6
200	44.3	1	19.7	2	6.6	5
300	22.9	4	11.1	5	8.4	7
400	28.1	3	18.6	6	8.6	7
500	34.5	4	20.3	5	11.4	8
600	65.9	5	13.7	7	10.1	8
700	12.7	2	14.5	4	7.9	7
800	11.3	1	18.9	2	4.4	5
900	70.2	7	10.4	5	8.5	7
1000	38.4	5	15.8	6	11.9	9

Table. 1 The number of bits recovered from the key as compared with Genetic Algorithm, Simulated Annealing and Tabu Search.