San Jose State University
SJSU ScholarWorks

Master's Projects

Master's Theses and Graduate Research

Fall 2011

EFFICIENT ATTACKS ON HOMOPHONIC SUBSTITUTION CIPHERS

Amrapali Dhavare
San Jose State University

EFFICIENT ATTACKS ON HOMOPHONIC SUBSTITUTION CIPHERS

A Project Report

Presented to

The faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Masters of Computer Science

by

Amrapali Dhavare

(SJSU ID: 007486180)

December 2011

SAN JOSE STATE UNIVERSITY

The Undersigned Project Committee Approves The Project Titled

EFFICIENT ATTACKS ON HOMOPHONIC SUBSTITUTION CIPHERS

By

Amrapali Dhavare

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

Dr. Mark Stamp, Department of Computer Science Date

Dr. Sami Khuri, Department of Computer Science Date

Dr. Richard Low, Department of Mathematics Date

APPROVED FOR THE UNIVERSITY

Associate Dean, Office of Graduate Studies and Research Date

ABSTRACT

Efficient Attacks On Homophonic Substitution Ciphers

by Amrapali Dhavare

Substitution ciphers are one of the earliest types of ciphers. Examples of classic substitution ciphers include the well-known simple substitution and the less well-known homophonic substitution. Although simple substitution ciphers are indeed simple - both in terms of their use and attacks; the homophonic substitution ciphers are far more challenging to break. Even with modern computing technology, homophonic substitution ciphers remain a significant challenge.

This project focuses on designing, implementing, and testing an efficient attack on homophonic substitution ciphers. We use an iterative approach that generalizes the fastest known attack on simple substitution ciphers and also employs a heuristic search technique for improved efficiency. We test our algorithm on a wide variety of homophonic substitution ciphers. Finally, we apply our technique to the “Zodiac 340” cipher, which is an unsolved ciphertext created in the 1970s by the infamous Zodiac killer.

ACKNOLEDGMENTS

I would like to thank my project advisor Dr. Mark Stamp for his exceptional guidance, valuable insights, unending support, and immense patience. Without him my project could never have been completed. I would also like to thank my committee members Dr. Sami Khuri and Dr. Richard Low for their support and patience.

Finally, I would like to thank my family and my husband Sachin Gaikwad for their encouragement, motivation, and emotional support.

Table of Contents

List of Figures

List of Tables

1 Introduction

The substitution ciphers among the classic cryptographic systems are one of the oldest ciphers [17]. They have been popularly well known and widely studied. Many variants of the substitution cipher have been invented. Few of the examples are simple and homophonic substitution ciphers. The simple substitution cipher is indeed simple in terms of its use and it has been successfully broken using frequency based attacks. On the other hand, a slight variant of simple substitution cipher called the homophonic substitution cipher is much more complex and robust to the frequency based attacks.

The infamous Zodiac 340 cipher has a good chance of being a homophonic substitution cipher, since its predecessor the Zodiac 408 was a homophonic cipher [12]. The Zodiac ciphers were created by a serial killer named Zodiac in 1960-70 [2]. Out of the four Zodiac ciphers, only one cipher named as Zodiac 408 was broken successfully. The remaining three Zodiac ciphers still remain unsolved; even after forty years. The Zodiac 340 cipher is the most famous of all Zodiac ciphers. Even in today's world, where extremely powerful supercomputers are used to solve exceptionally complex problems; the Zodiac 340 cipher still remains a mystery.

The goal of this project is to design, implement, and test an efficient attack on homophonic substitution ciphers. This attack will also be used as an attempt to break the

Zodiac 340 cipher. The attack proposed in this paper makes heavy use of the fast algorithm presented in the paper [7]. In the mentioned paper, the fast algorithm was proposed for breaking simple substitution ciphers. The Section 3.2 presents a way to extend the fast algorithm to apply for homophonic substitution ciphers. The extension of the fast algorithm to homophonic substitution gives only a partial solution. The problems imposed by the complex nature of homophonic substitution are addressed in Section 4.

Our solution is based on the hill-climbing heuristic technique, where an arbitrary solution is refined through a series of iterations [10]. The time spent on the iterations is optimized by using digram frequency comparisons as opposed to parsing the ciphertext in each iteration, as done in frequency based attacks. The algorithm presents a multi-layered architecture with three nested loops in order to address the problems imposed by the homophonic substitution ciphers and the hill-climbing technique.

This project report is organized as follows. Section 2 briefly describes various concepts used in our solution. Section 3 describes the fast algorithm proposed in the paper [7] and presents a way for extending it to the homophonic substitution ciphers. Section 4 describes the complete solution for attacking the homophonic substitution ciphers. Sections 6, 7, and 8 describe the tests, results, and the analysis of the results. Section 9 describes the Zodiac ciphers. Finally, Section 10 concludes the project report.

2 Background

In the process of deriving our solution, we studied and researched various concepts mainly including the substitution ciphers, digram frequencies, and hill-climbing technique. In this section we review these concepts briefly.

2.1 Substitution Ciphers

The substitution ciphers can be defined as the ciphers in which every letter in a plaintext is substituted with a ciphertext symbol and the original position of the plaintext letter is retained in the resultant ciphertext [5]. There are various ways in which the substitution can be done. For example, one plaintext letter can be substituted with only one single ciphertext symbol corresponding to one to one mapping, one plaintext letter can be substituted with multiple ciphertext symbols corresponding to one to many mapping, and multiple plaintext letters can be substituted with multiple ciphertext symbols corresponding to many to many mapping. The substitution ciphers have many variants based on the type of mapping used for substitution; two such variants known as the simple and homophonic substitution ciphers are described in detail in the following sections.

2.1.1 Simple Substitution

The simple substitution is the simplest form of substitution ciphers, where each plaintext letter is mapped to a single ciphertext symbol, that is, the mapping from plaintext to ciphertext is one to one [17]. The one to one mapping of simple substitution cipher makes it susceptible to attacks based on statistical frequency analysis. An example of a simple substitution cipher is given in Figure 1.

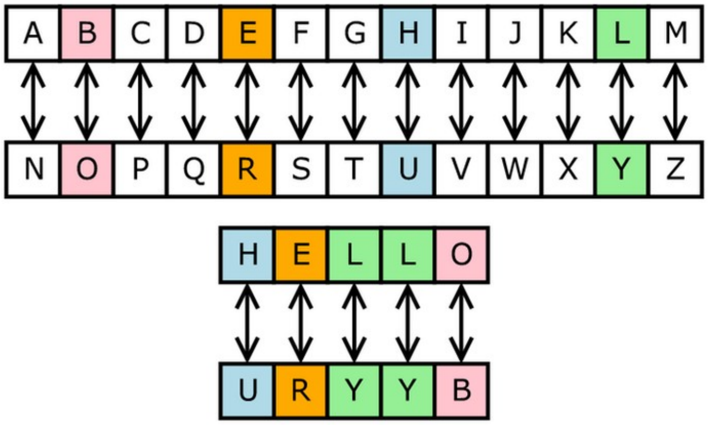


Figure 1: Simple Substitution Cipher [15]

As displayed in Figure 1, the plaintext “HELLO” is encrypted as “URYYB”. An important point to be noticed here is that, the ciphertext symbol 'Y' maintains the frequency of the letter 'L' from the plaintext. Similarly, when larger plaintext is encrypted using simple substitution cipher, its corresponding ciphertext maintains the letter frequency distribution of the plaintext.

Considering English as the expected language of plaintext, the total number of distinct plaintext letters is 26. The theoretical key-space of simple substitution can be calculated as the total number of permutations of the possible keys which is equal to 26! [17]. Therefore, the work factor for exhaustive search is 26! which is approximately equal to 2^{88} . Taking an example, an exhaustive key search on a personal computer which can test 10^6 keys per second, will take $2^{88}/10^6=4.03*10^{20}$ seconds which is equivalent to $1.28*10^{13}$ years. Thus, the exhaustive search for simple substitution is infeasible.

One of the most popular attacks on simple substitution cipher is using letter frequency statistics. In any language, each letter has a certain frequency associated with it; for example, in English language [6], the letter 'e' has the highest frequency (13%) of occurrence followed by the letters 't' (9%) or 'a' (8%). Thus, in simple substitution, since each plaintext letter is mapped to a single cipher symbol, the symbol frequency distribution in encrypted ciphertext reflects the original frequency distribution of the plaintext. This information is extremely useful in designing an attack on simple substitution ciphers. The attack parses the ciphertext in order to collect the cipher symbol frequencies. The cipher symbol frequency statistics are then used for mapping the ciphertext symbols to the plaintext letters. Therefore, it is quite easy to break the simple substitution ciphers using letter frequencies. The attack based on the statistical letter frequency analysis is described with following algorithm.

1. Construct the initial key by using the letter frequency statistics
2. Parse the ciphertext with the putative key to obtain the putative plaintext
3. Compute a score to measure how close the putative plaintext is to the expected language of plaintext (This can be done by counting the number of meaningful words using a dictionary [13])
4. Loop for a number of iterations
    1. Modify the putative key
    2. Parse the ciphertext with the modified putative key to obtain the putative plaintext
    3. Compute score for new putative plaintext
5. Repeat

The major drawback of this algorithm is that, the ciphertext is parsed in every iteration. Therefore, as the size of the ciphertext increases, this algorithm becomes more and more expensive. The fast algorithm described in Section 3, substantially reduces the time spent on parsing the ciphertext in every iteration.

The one to one mapping of simple substitution cipher makes it susceptible to statistical frequency based attacks. If the frequency distribution of simple substitution cipher is manipulated in such a way that the ciphertext produces a random frequency distribution, then the frequency based attack will not work on such ciphers. The homophonic substitution cipher is one such variant of the substitution cipher where the frequency distribution is flattened in the resultant ciphertext.

2.1.2 Homophonic Substitution

Homophonic substitution cipher is a much more complicated variant of substitution cipher where, instead of using one to one mapping of simple substitution, one to many mapping is used [8]. In one to many mapping, each plaintext letter can be substituted with multiple ciphertext symbols. However, each ciphertext symbol can represent one and only one plaintext letter. Such mapping tends to flatten the frequency statistics in the resulting ciphertext and consequently makes the attacks based on statistical frequency based analysis more and more difficult. An example of homophonic cipher is given in Figure 2.

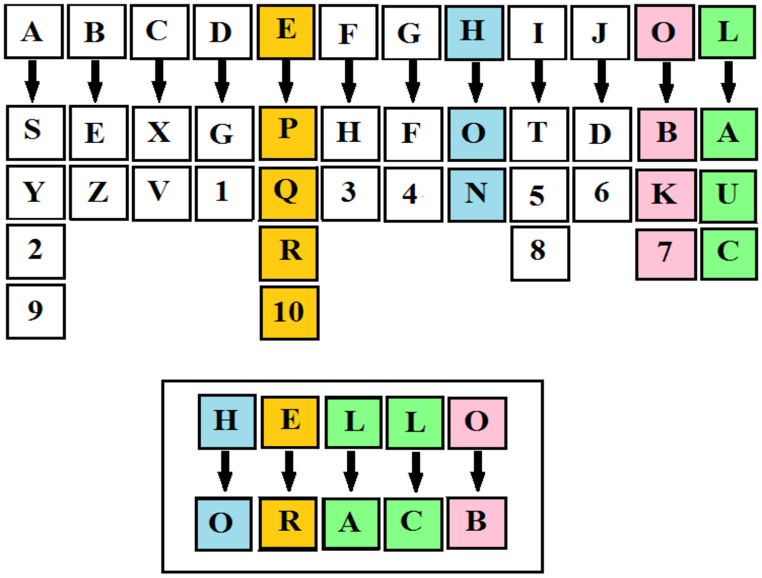


Figure 2: Homophonic Substitution Cipher

As seen in Figure 2, each letter can be substituted with multiple cipher symbols. For instance, letter 'L' can be substituted with 'A', 'U', or 'C'. In the ciphertext of word "HELLO", it is seen that the two occurrences of 'L' are substituted with two different ciphertext symbols. Thus, the resultant ciphertext does not give any idea that the cipher symbols 'A' and 'C' actually represent the same plaintext letter 'L'.

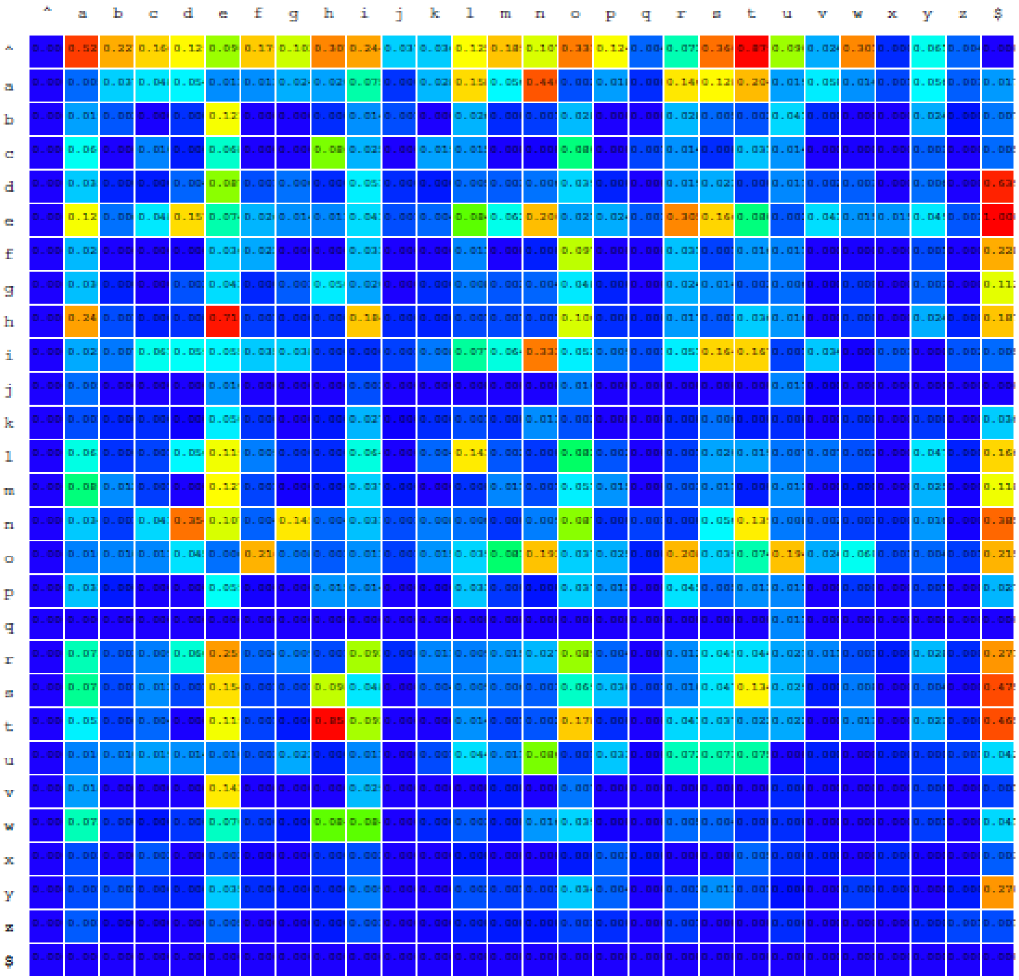
If the ciphertext has 'N' distinct ciphertext symbols and the expected language of plaintext is English, then the homophonic substitution cipher has the theoretical key space of

$26^N \approx 2^{5N}$ as opposed to 26! of simple substitution cipher. An exhaustive key search on a personal computer which can test 10^6 keys per second for a ciphertext with N = 100, will take $26^{100}/10^6 = 3.14 * 10^{135}$ seconds which is equivalent to $9.96 * 10^{127}$ years. An exhaustive search for simple substitution cipher takes $1.28 * 10^{13}$ years. Thus, the difference between key-spaces for simple and homophonic substitution ciphers increases exponentially as the number of distinct ciphertext symbol increases.

2.2 Digram Frequencies

The digram frequency can be defined as the frequency of occurrence of a certain symbol followed by another symbol. It is studied that, knowledge of digram frequency distribution of the expected language of plaintext and the digram frequency distribution of the ciphertext is sufficient to break the simple substitution cipher [9]. The use of digram frequencies in designing an attack on substitution ciphers substantially reduces the efforts spent on parsing the ciphertext in every iteration. The digram distribution matrix for English language is displayed in Table 1.

In the digram distribution matrix displayed in Table 1, the space character is also considered along with the 26 letters of the English language. The character '^' represents the space occurring at the beginning of a word and character '$” represents the space occurring at the end of a word. The digram frequencies in the matrix are color coded. The red color represents the higher values of the digram frequencies and the blue color represents the lower values of the digram frequencies.



	^	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	$
^	0.00	0.52	0.22	0.16	0.12	0.09	0.17	0.10	0.30	0.24	0.03	0.03	0.12	0.18	0.10	0.33	0.12	0.00	0.07	0.36	0.87	0.09	0.02	0.30	0.00	0.06	0.00	0.00
a	0.00	0.00	0.03	0.04	0.05	0.01	0.01	0.02	0.02	0.07	0.00	0.02	0.15	0.05	0.44	0.00	0.01	0.00	0.14	0.12	0.20	0.01	0.05	0.01	0.00	0.05	0.00	0.01
b	0.00	0.01	0.00	0.00	0.00	0.12	0.00	0.00	0.00	0.01	0.00	0.00	0.02	0.00	0.00	0.02	0.00	0.00	0.02	0.00	0.00	0.04	0.00	0.00	0.00	0.02	0.00	0.00
c	0.00	0.06	0.00	0.01	0.00	0.06	0.00	0.00	0.08	0.02	0.00	0.01	0.01	0.00	0.00	0.08	0.00	0.00	0.01	0.00	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00
d	0.00	0.01	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.01	0.02	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.63
e	0.00	0.12	0.00	0.04	0.15	0.07	0.02	0.01	0.01	0.04	0.00	0.00	0.08	0.06	0.20	0.02	0.02	0.00	0.30	0.16	0.08	0.00	0.04	0.01	0.01	0.04	0.00	1.00
f	0.00	0.02	0.00	0.00	0.00	0.03	0.02	0.00	0.00	0.03	0.00	0.00	0.01	0.00	0.00	0.09	0.00	0.00	0.03	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.22
g	0.00	0.03	0.00	0.00	0.00	0.04	0.00	0.00	0.05	0.02	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.00	0.02	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.11
h	0.00	0.24	0.00	0.00	0.00	0.71	0.00	0.00	0.00	0.18	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.01	0.00	0.03	0.01	0.00	0.00	0.00	0.02	0.00	0.18
i	0.00	0.02	0.00	0.06	0.05	0.05	0.03	0.03	0.00	0.00	0.00	0.00	0.07	0.06	0.33	0.05	0.00	0.00	0.05	0.16	0.16	0.00	0.03	0.00	0.00	0.00	0.00	0.00
j	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
k	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03
l	0.00	0.06	0.00	0.00	0.05	0.11	0.00	0.00	0.00	0.06	0.00	0.00	0.14	0.00	0.00	0.08	0.00	0.00	0.00	0.02	0.01	0.00	0.00	0.00	0.00	0.04	0.00	0.16
m	0.00	0.08	0.01	0.00	0.00	0.12	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.01	0.00	0.05	0.01	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.00	0.02	0.00	0.11
n	0.00	0.03	0.00	0.04	0.35	0.10	0.00	0.14	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.05	0.13	0.00	0.00	0.00	0.00	0.01	0.00	0.38
o	0.00	0.01	0.01	0.01	0.04	0.00	0.21	0.00	0.00	0.01	0.00	0.01	0.03	0.08	0.19	0.03	0.02	0.00	0.20	0.03	0.07	0.19	0.02	0.06	0.00	0.00	0.00	0.21
p	0.00	0.01	0.00	0.00	0.00	0.05	0.00	0.00	0.01	0.01	0.00	0.00	0.03	0.00	0.00	0.03	0.01	0.00	0.04	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.02
q	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
r	0.00	0.07	0.00	0.00	0.06	0.25	0.00	0.00	0.00	0.09	0.00	0.01	0.00	0.01	0.02	0.08	0.00	0.00	0.01	0.04	0.04	0.02	0.01	0.00	0.00	0.02	0.00	0.27
s	0.00	0.07	0.00	0.01	0.00	0.15	0.00	0.00	0.09	0.04	0.00	0.00	0.00	0.00	0.00	0.06	0.03	0.00	0.01	0.04	0.13	0.02	0.00	0.00	0.00	0.00	0.00	0.47
t	0.00	0.05	0.00	0.00	0.00	0.11	0.00	0.00	0.85	0.09	0.00	0.00	0.01	0.00	0.00	0.17	0.00	0.00	0.04	0.03	0.02	0.02	0.00	0.01	0.00	0.02	0.00	0.46
u	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.02	0.00	0.01	0.00	0.00	0.04	0.01	0.08	0.00	0.03	0.00	0.07	0.07	0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.04
v	0.00	0.01	0.00	0.00	0.00	0.14	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
w	0.00	0.07	0.00	0.00	0.00	0.07	0.00	0.00	0.08	0.08	0.00	0.00	0.00	0.00	0.01	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.04
x	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.27
z	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 1: Digram Frequencies For English Language [3]

2.3 Heuristic Methods

As stated in Section 2.2, the homophonic substitution cipher has an extremely huge key space, for which no algorithm is available which can solve the cipher in polynomial time. Therefore, we decided to consider a heuristic approach to design our solution.

Heuristic algorithm is defined below as given in the book [10],

> *"Heuristic algorithm is used to describe an algorithm that tries to find a certain combinatorial structure or solve an optimization problem by the use of heuristics. A heuristic is a method of performing a minor modification, or a sequence of modifications, of a given solution or partial solution in order to obtain a different solution"*

The heuristic algorithms are used to determine good or close to optimal solutions in fast and easy manner. However, the heuristic algorithms do not guarantee that they will find the exact or even approximate solution, unlike the exact and approximate algorithms respectively [19]. Our solution is based on the hill-climbing technique which is a kind of a heuristic algorithm.

2.3.1 Hill-climbing Technique

Hill-climbing is an iterative technique which starts with an arbitrary solution and refines the solution through a series of iterations [10]. During each iteration, a minor modification is done to the solution to obtain a different solution. The modified solution is evaluated using a function to decide if the modified solution is better or worse than the previous solution. If the modified solution is better, then the change is retained; else, the change is discarded and the previous solution is modified with a different change. Thus, with every modification, the algorithm proceeds only to a better solution.

To design a hill-climbing algorithm, two key points need to be clearly defined. The first key point is a way to incrementally modify the solution during iterations and the second key point is a way of measuring the "goodness" of the solution. The goodness of the solution can be measured in terms of a numeric score. The solution which improves the score is retained and the solution which degrades the score is discarded. Thus, the algorithm always climbs up towards a better solution, as indicated by the name of the technique – Hill-climbing.

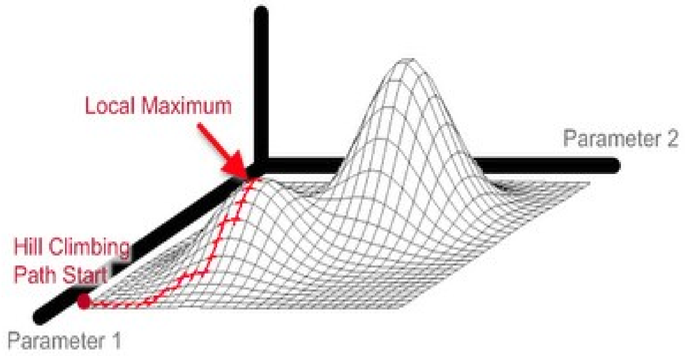


Figure 3: Hill-climbing Technique [11]

The major drawback of the hill-climbing technique is that, it is crucial where the initial solution starts. Depending on the initial starting point, the algorithm can obtain only the local optimum solution and occasionally the global optimum solution. Also, it is quite possible during the iterations that a solution with a bad score might end up in a much better solution, if retained for the future iterations. However, as this technique does not consider any modification that does not give a better solution, it ignores all such instances of the solution. To overcome the drawback, multiple initial starting points should be considered instead of considering only one single starting point. The advantage of using multiple initial solutions is that, each solution will reach its own local optimum solution and these multiple local optimum solutions can be compared with each other to select the best solution among them.

The hill-climbing technique works on substitution ciphers, but it does not work on the modern ciphers. In substitution ciphers, if the number of correctly solved ciphers symbols is more, then the putative plaintext will look more similar to the actual plaintext. In other words, the distance between a putative key and the actual key is reflected in the distance between the putative plaintext and the actual plaintext. The closer a putative key is to the actual key, the resultant putative plaintext too will be closer to the actual plaintext, as compared to the putative plaintext resulted from a putative key which is not as close to the actual key. On the other hand, for a modern cipher, the distance between the putative key and the actual key does not matter at all. For any incorrect putative key, irrespective of how close it is to the actual key, the putative plaintext will still look random and nowhere close to the actual plaintext. This behavior can be clearly seen in the Figure 4, which shows the results of an experiment conducted on the modern block cipher AES(Advanced Encryption Standard) [1].

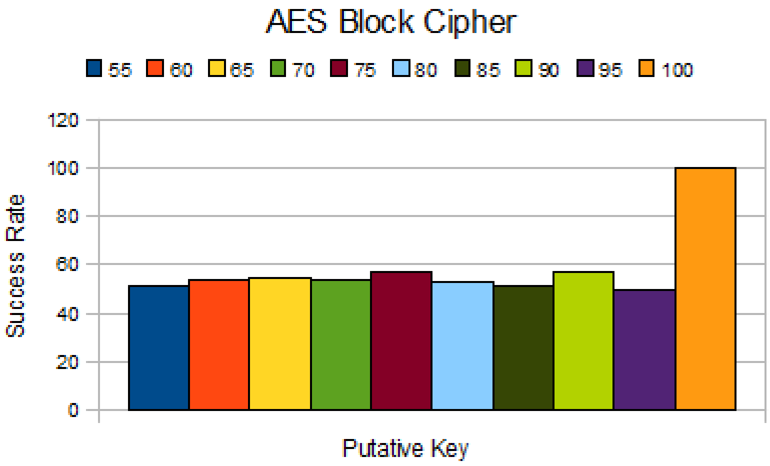


Figure 4: Graph Of AES Block Cipher Success Rate

In the displayed graph, the X axis represents the percentage of closeness of the putative key to the actual key and it increases progressively from 55% to 100%. The Y axis represents the percentage of similarity between the putative plaintext and the actual plaintext. The graph clearly shows that even if the putative key gets closer and closer to the actual key, the percentage of similarity between the putative plaintext and the actual plaintext remains random. It is only when the putative key is 100% same as the actual key, the putative plaintext completely matches with the actual plaintext. The details of the experiment are given in Section 13.1.

3 Fast Algorithm For Substitution Ciphers

An extremely smart and fast method to break simple substitution cipher was proposed in the paper [7]. This fast algorithm uses digram frequency distribution to find the solution faster. With this algorithm, the ciphertext is parsed only once in the beginning to construct the digram distribution matrix. The subsequent evaluations of plaintext are done by manipulating the digram distribution matrix only. Therefore, parsing of the ciphertext in every iteration is no more required in this algorithm.

For this algorithm, two digram frequency distribution matrices are required – one for the expected language of plaintext and another for the ciphertext. The distribution matrix with the digram frequencies of the expected language of plaintext is taken as a reference. The two matrices are compared with each other in order to evaluate an intermediate solution during the iterations. The numeric difference between the matrices is used to compute a score which reflects the "goodness" of the intermediate solution. The more similar the matrices are to each other, lesser will be the score. The distribution matrix for ciphertext is constructed only once at the beginning. In later iterations, when a solution is modified, only required changes are done to the corresponding rows and columns of the matrix, without constructing the whole matrix again. Thus, a valuable amount of time spent in parsing the ciphertext with the intermediate solution is saved in each iteration. The sketch of a generic algorithm is given below.

1. Construct or obtain the distribution matrix for the expected language of the plaintext
2. Construct an initial key and the distribution matrix for the ciphertext
3. Compute score for the initial key using the distribution matrices
4. Iterations
    1. Alter the key little bit by swapping two elements
    2. Update the distribution matrix for the ciphertext with the modified key
    3. Compute score for the modified key using the modified distribution matrix

For the stated algorithm, three key points need to be elaborated. The first key point is that, a method for constructing the initial solution needs to be defined. There are various ways for constructing the initial solution such as using a simple frequency analysis of the ciphertext, using a partial knowledge of the solution, or the initial solution can be purely random. Any of these methods can be selected for constructing the initial solution.

Next, the second point is that, a method needs to be defined for modifying the solution during iterations. A minor modification is made to the solution in each iteration to obtain a different solution. This modification can be done by swapping two elements of the solution. The swapping can be performed in the following manner. Let S be a vector of N ciphertext symbols ranked in the order of their descending frequencies such that S_1 will have the highest frequency, S_2 will have the second highest frequency, followed by S_3 , S_4 , and so on. The elements for swapping can be selected through a series of progressive rounds. In the first round, all the adjacent elements will be selected for swapping. That is, S_1 will be swapped with S_2 , S_2 with S_3 and so on. In the second round, the adjacent elements with a distance of two will be selected for swapping. That is S_1 will be swapped with S_3 , S_2 with S_4 , and so on. In the last round, S_1 will be swapped with S_N . These rounds can be summarized as follows. First, we try $S_1|S_{2,}S_2|S_{3,}S_3|S_{4.}..,S_{N-1}|S_N$, then $S_1|S_{3,}S_2|S_{4,}S_3|S_{5.}..,S_{N-2}|S_N$, then $S_1|S_{4,}S_2|S_{5,}S_3|S_{6.}..,S_{N-3}|S_N$, …, and finally $S_1|S_N$.

The last key point is that, a method for evaluating the goodness of the intermediate solution needs to be defined. The evaluation of an intermediate solution can be done by constructing a function to compute a score to measure how close the distribution matrix of ciphertext is to the distribution matrix of the expected language of plaintext. This function will compare the two distribution matrices to measure the goodness of the solution. The function can be simply defined as the sum all numerical differences of all corresponding elements of the two digram frequency distribution matrices. Let, *F(t)* be the function that evaluates the “goodness” of text 't' and returns a numeric score. Let *K* be the putative key and *V* be the score returned by *F(t)* and finally, let *D* be the distribution matrix of ciphertext and *E* be the distribution matrix of the expected language of plaintext. The evaluation function can be represented by the following math formula [7]

$$v=\mathrm{f}\left(d\left(c,\grave{\mathrm{k}}\right)\right)=\sum_{i,j}\left(D_{ij}\left(d\left(c,\grave{\mathrm{k}}\right)\right)-E_{ij}\right)=\sum_{i,j}\left(\grave{D}_{ij}\left(d\left(c,k\right)\right)-E_{ij}\right)$$

According to the evaluation formula, for every modification to the key *K*, only those rows and columns of the distribution matrix are altered, which are affected by the modification of the key. All the other rows and columns are kept as they are.

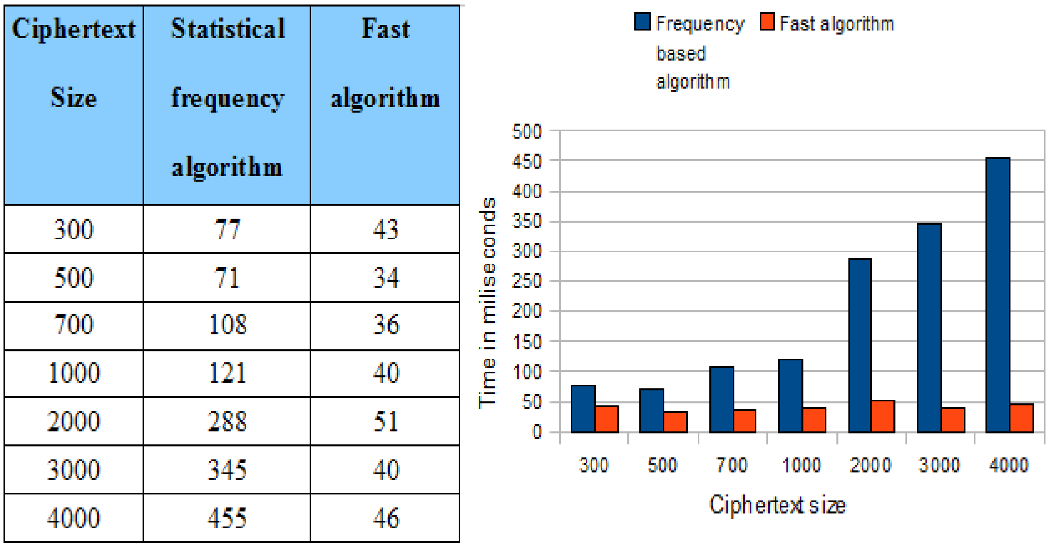
The matrix modification is explained in detail with the following diagram. For example, if we want to modify the key by swapping letters *D* and *G,* then this change can be applied to the distribution matrix by modifying only those rows and columns belonging to the letters *D* and *G*. Rest of the rows and columns belonging to other letters which are not altered need not be modified.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
A	0.3	0.2	0.1	0.2	0.1	0.3	0.2	0.1	0.3	0.2
B	0.1	0.2	0.1	0.2	0.3	0.2	0.4	0.2	0.4	0.1
C	0.3	0.2	0.1	0.1	0.4	0.5	0.6	0.2	0.6	0
D	0.2	0.2	0.1	0.1	0	0.1	0.2	0.3	0.2	0.4
E	0.3	0.4	0.4	0.2	0.1	0.9	0	0.3	0	0.3
F	0.2	0.2	0.3	0.4	0.2	0.1	0.9	0	0	0.3
G	0.5	0.4	0.2	0.2	0.1	0.2	0.6	0.3	0.5	0.2
H	0.4	0.3	0.1	0.1	0.1	0.2	0	0.4	0.2	0.2
I	0	0.2	0	0.3	0.2	0	0.5	0.2	0.3	0.1
J	0.1	0.3	0.2	0	0	0.3	0.3	0.2	0.5	0.2

Table 2: Modification Of Distribution Matrix

3.1 Simple Substitution

As described in the Section 2.1.1, the statistical frequency based attack on simple substitution is expensive, as the ciphertext is parsed in every iteration [6]. Where as, in the fast algorithm approach, the ciphertext is parsed only once in the beginning to construct the digram distribution matrix and all subsequent evaluations of the solution are done using matrix manipulations only. Thus, the fast algorithm proves to be much more time efficient as compared to the statistical frequency based attack. The Figure 5 presents the comparison between the two algorithms in terms of execution time on simple substitution cipher. The table displays the results of an experiment conducted on multiple simple substitution ciphers of various sizes using both algorithms. The substantial difference in the execution times between the two algorithms can be clearly seen in Figure 5. The time is given in milliseconds. As the ciphertext size increases, the time taken by statistical frequency based algorithm increases rapidly. On the other hand, for fast algorithm, the execution time does not increase that noticeably. In the graph displayed in Figure 5, blue and orange bars represent the execution times for frequency based algorithm and fast algorithm respectively.



Ciphertext Size	Statistical frequency algorithm	Fast algorithm
300	77	43
500	71	34
700	108	36
1000	121	40
2000	288	51
3000	345	40
4000	455	46

Figure 5: Comparison Between Frequency Based And Fast Algorithm

3.2 Homophonic Substitution

This section describes how the fast algorithm can be extened for the homophonic substitution ciphers. For homophonic substitution ciphers, the ciphertext can have more than 26 distinct symbols and multiple cipher symbols can represent the same plaintext letter. This essentially affects total three key points similar to those described for the generic fast algorithm. The affected key methods are - construction of the initial key, modification to the key, and modification of the distribution matrix. We will next describe methods for extending the affected functions for homophonic substitution ciphers.

Let the number of distinct cipher symbols in a given ciphertext be represented by N. Then, the putative key for homophonic cipher can be represented as 1xN array. The putative key can be constructed by using a pure random guess. An educated guess of the putative key can be made by adding up the frequencies of selective ciphertext symbols in such a way that the addition matches the frequency of a particular plaintext letter. This is explained with the example given in Table 3. As seen in the Table 3, the ciphertext symbols '3' and '4' are chosen to map to the plaintext letter 'e', because the frequencies of symbols '3' and '4' add up to the expected frequency of letter 'e'.

Cipher symbol	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**…**	**27**	**28**
Frequency	8.3	7.5	6.7	6.6	6.3	5.5	4.4	…	0	0
Plaintexts letter	A	O	E	E	I	T	T	…	Q	Z

Table 3: Construction Of Initial Key For Homophonic Cipher

The modification of the key can be done in the same way as that of the simple substitution, by progressively selecting two elements from the putative key for swapping. The only difference for homophonic substitution will be that, the ciphertext symbols representing the same plaintext letter will not be swapped. This is described with the following example.

Cipher symbol	0	1	2	3	4	5	6	…	27	28
Frequency	8.3	7.5	6.7	6.6	6.3	5.5	4.4	…	0	0
Plaintexts letter	O	A	E	E	I	T	T	…	Q	Z
Cipher symbol	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**…**	**27**	**28**
Frequency	8.3	7.5	6.7	6.6	6.3	5.5	4.4	…	0	0
Plaintexts letter	A	E	O	E	I	T	T	…	Q	Z
Cipher symbol	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**…**	**27**	**28**
Frequency	8.3	7.5	6.7	6.6	6.3	5.5	4.4	…	0	0
Plaintexts letter	A	O	T	E	E	T	T	…	Q	Z

Table 4: Modification Of Key For Homophonic Cipher

As seen in the Table 4, in the third iteration, symbols '2' and '3' are not swapped with each other, as the operation would not have produced any useful result. Instead, symbol '2' is swapped with the next symbol '4'.

The function of modifying the distribution matrix can be extended in the following way for homophonic substitution cipher. As the homophonic cipher can have more than 26 distinct cipher symbols, we need to maintain two digram distribution matrices for representing ciphertext digram frequencies – D1 and D2. The first distribution matrix D1 will have the size NxN and can be represented as D1 [N] [N]. This matrix will represent the numeric counts of the digram frequencies of ciphertext symbols. This matrix will be constructed only once at the beginning and it will only be referred and not modified during the iterations. The second distribution matrix D2 will have the size of 26 x 26 and can be represented as D2 [26] [26]. This matrix will represent the probabilities of the digram frequencies after the putative key has been applied to D1. This matrix will be modified when the key is modified during iterations. The information from matrix D1 will be used for modifying the matrix D2. The matrix D2 will be used for actual comparison with the matrix E for computing the score in order to evaluate a solution.

4 Design

The Section 3.2 described how the fast algorithm can be extended for homophonic substitution ciphers. However, it gives only a partial solution. In order to get the complete solution, we need to solve the problems imposed by hill-climbing technique and homophonic substitution. To solve the drawback of the hill-climbing technique of obtaining only the local optimum solution, multiple initial starting points need to be constructed [10]. This can be done by using the fast algorithm for homophonic cipher mentioned in the Section 3.2 as the *Inner Hill-climbing* layer and having an outer layer *Random Solution Generator* to simply generate a number of instances of arbitrary solutions as the starting points. The *Inner Hill-climbing* will be called on each of the arbitrary solution constructed in the *Random Solution Generator* layer and it will return the corresponding local optimum solution back to the *Random Solution Generator* layer. After all of the arbitrary solutions are processed by the *Inner hill-climbing* layer, the *Random Solution Generator* layer will compare all the local optimum solutions to find the best available solution.

Next, coming to the problem imposed by the homophonic substitution, since, the homophonic cipher has the nature of one to many mapping, multiple cipher symbols can represent the same plaintext letter. While attacking the homophonic cipher, we do not have the knowledge of exact frequency distribution mapping used in the ciphertext. All we know is the number of the distinct cipher symbols. In order to construct the initial arbitrary solution, it is necessary to know the frequency distribution mapping from cipher symbols to the plaintext letters. For example, if the expected language of plaintext is English, then the number of distinct plaintext letters is 26. Now, if the ciphertext has 27 distinct cipher symbols, it can be safely assumed that the letter 'e' (which has the highest frequency for English language) could be represented by 2 cipher symbols and all remaining English letters could be represented by single cipher symbols.

Assuming that the ciphertext has the frequency statistics as flattened as possible, a rough guess about the “Frequency Distribution Mapping”can be made using the number of distinct cipher symbols in the ciphertext. The Table 5 displays the frequency distribution mapping for selective cipher symbol sizes. The first column represents the number of total distinct cipher symbols in the ciphertext. The first row represents the plaintext letters in descending order of their frequency. The rest of the cells represent the number of cipher symbols representing the corresponding plaintext letter. For example, in a ciphertext with 35 cipher symbols, the letter 'E' can be represented with 4 cipher symbols, the letter 'T' can be represented with 2 cipher symbols, 'A' with 2, and so on.

CS	E	T	A	O	I	N	S	R	H	D	L	C	U	M	F	W	G	Y	P	B	V	K	X	J	Q	Z
26	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
27	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
35	4	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
45	5	4	3	3	3	3	3	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
55	7	5	4	4	4	3	3	3	3	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
65	8	6	5	5	5	4	4	4	4	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
75	9	7	6	6	5	5	5	4	4	3	3	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1
85	11	8	7	7	6	6	5	5	5	3	3	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1
95	12	9	8	7	7	7	6	6	5	4	4	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1

Table 5: Frequency Distribution Mapping

The Table 5 presents the frequency distribution mapping under the assumption that the frequency statistics of the plaintext letters are as flattened as possible in the ciphertext. However, it is quite possible that the ciphertext may not have the frequency statistics of the plaintext letters as much flattened. Therefore, on top of the *Random Solution Generator* layer, an *Outer hill-climbing* layer needs to be added to provide various permutations of the frequency distribution mappings for the given ciphertext. The Figure 6 displays a high level design diagram presenting the relation between all three layers.

The following sections describe each of the layers briefly.

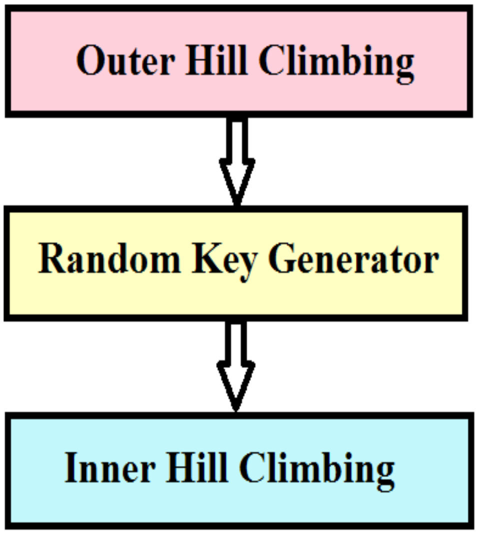


Figure 6: High Level Design Diagram

4.1 Inner Hill-climbing

The *Inner Hill-climbing* layer takes an arbitrary solution as an input from the *Random Solution Generator* layer. It then performs the fast algorithm for homophonic substitution cipher on the input solution and derives the corresponding local optimum solution. During the iterations, the solution is incrementally modified by swapping two elements at a time. This layer returns the local optimum solution and the corresponding score to the *Random Solution Generator* layer.

4.2 Random Solution Generator

The *Random Solution Generator* layer takes the frequency distribution mapping as an input from the *Outer Hill-climbing* layer. It generates a number of arbitrary solutions based on the input frequency distribution mapping. It then calls the *Inner Hill-climbing* layer for each of the solution generated in this layer. After the *Inner Hill-climbing* layer returns the local optimum solutions for all of the arbitrary solutions generated in this layer, it selects the best solution based on the score. It then returns the best solution and the best score to the *Outer Hill-climbing* layer.

4.3 Outer Hill-climbing

The *Outer Hill-climbing* layer computes the frequency distribution mapping based on the number of distinct cipher symbols in the given ciphertext. The frequency distribution mapping is computed with the assumption that the frequency statistics are as flattened as possible in the ciphertext. It calls the *Random Solution Generator* layer with the computed frequency distribution mapping. It then performs the hill-climbing technique on the frequency distribution mapping by modifying the frequency distribution, keeping the total cipher symbols size constant. For each of the frequency distribution mappings, it calls the *Random Solution Generator* layer and selects the best solution among all the solutions returned by *Random Solution Generator* layer as the final solution.

4.4 Block Diagram

The data flow through all the layers is displayed in the Figure 7. The ciphertext is provided to the *Outer Hill-climbing* layer as an input. The *Outer Hill-climbing* layer will generate various frequency distribution mappings depending on the number of distinct cipher symbols. Each of the frequency distribution mappings is then provided to the *Random Solution Generator* which in turn generates a number of random initial starting points. Each of the initial starting points is passed to the *Inner Hill-climbing* layer which then refines the starting point through a series of iterations and computes the local optimum solution. The *Inner Hill-climbing* returns the local optimum solution back to the *Random Solution Generator* layer. The *Random Solution Generator* selects the best solution from the list of all local optimum solutions and returns it back to the *Outer Hill-climbing* layer. The *Outer Hill-climbing* layer selects the best solution from the list of solutions returned by *Random Solution Generator* layer. Finally, the *Outer Hill-climbing* layer outputs the best solution as the final solution in the form of the corresponding putative plaintext and putative key.

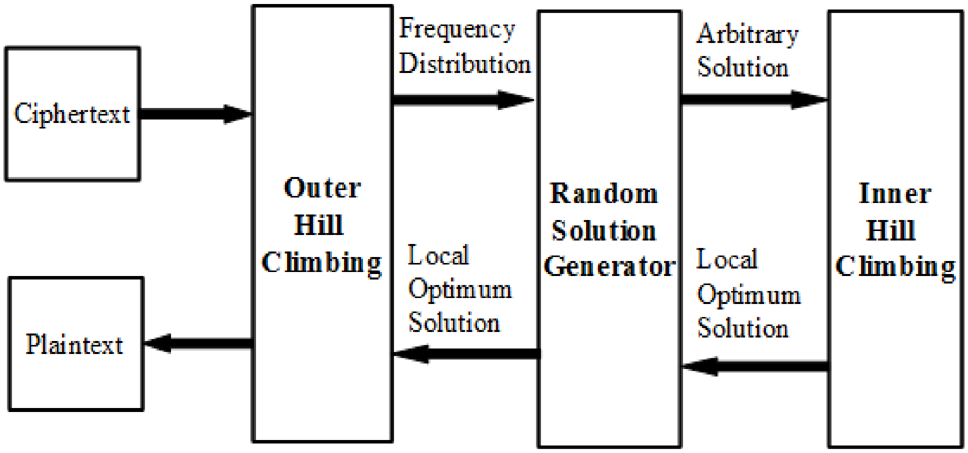


Figure 7: Block Diagram

4.5 Work Factor

As mentioned in the Section 2.1.3, the exhaustive key-space for homophonic substitution cipher is $26^N \approx 2^{5N}$, therefore the exhaustive key search attack is simply infeasible. The work factor for our solution can be calculated from four important factors - number of frequency distribution mappings(generated in *Outer Hill-climbing*), number of initial starting points(generated in *Random Solution Generator*), number of times elements swapped in *Inner Hill-climbing* layer, and number of comparisons done for computing the score. The resultant work factor is given below.

*Work factor = Number of frequency distribution mappings * Number of initial starting points * Number of times elements swapped * Number of comparisons done for computing the score*

Substituting N for the number of distinct cipher symbols and R for the number of initial starting points. We get the work factor as $26^2 * R * N^4 \approx 2^9 * R * N^4$ The work factor for our solution is much less than the work factor for exhaustive search. The term N^4 is the most influential term in the work factor and it can also be confirmed from the Figure 8 which displays a graph of execution times taken for breaking homophonic ciphers of various sizes and various cipher symbol sizes. It can be seen from the graph that, as the value of N increases, the execution time also increases in proportion to N^4 .

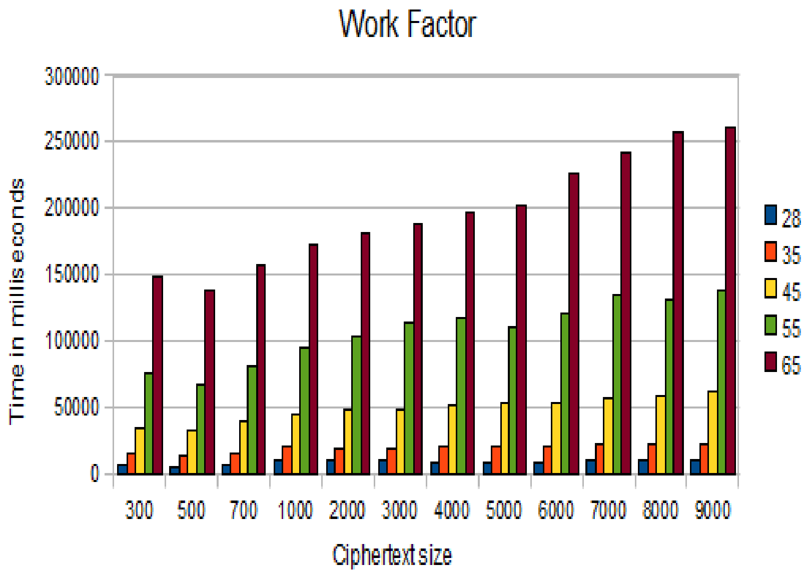


Figure 8: Work Factor

5 Implementation Details

We completed the implementation in three phases. In the first phase, we completed the implementation of fast algorithm for simple substitution cipher. In the second phase, we completed the implementation of extension of the fast algorithm for homophonic substitution cipher. In the third and last phase, we completed the implementation of the *Inner Hill-climbing, Random Solution Generator,* and *Outer Hill-climbing* layers. The implementation is done in C++ language on a UNIX platform.

6 Tests

We tested our solution on a variety of simple and homophonic substitution ciphers. To test the accuracy of the implementation, we self-generated various ciphertexts and saved the actual solutions. We also saved the putative solutions obtained after executing the attack to break the corresponding ciphers. Using the actual solution and the putative solution obtained from the execution, we calculated the percentage of correctly solved cipher symbols and plotted graphs to display the success rates as results. Apart from the self-generated simple and homophonic ciphers, test cases also include the famous Zodiac 340 cipher.

6.1 Test Plan

The test plan was executed in three phases. In phase 1, the self-generated simple substitution ciphers of various sizes were tested. In phase 2, self-generated homophonic substitution ciphers of various ciphertext sizes and cipher symbol sizes were tested. In the third and last phase, the famous Zodiac ciphers were tested.

6.2 Test Cases

The test cases cover different scenarios involving various ciphertext sizes, cipher symbol sizes, and number of initial starting points. All the test cases are written by assuming English as the expected language of plaintext. Also, every test case is constructed assuming two sets of the plaintext letters – the first set includes 26 English characters along with the space character and the second set includes just the 26 English characters.

6.2.1 Self-Generated Simple Substitution Ciphers

The test cases for simple substitution ciphers include total 24 ciphers of various ciphertext sizes ranging from 300 characters to 10,000 characters and plaintext letter sets of 26 English characters plus the space character and just the 26 English characters. The number of initial starting points provided to the *Inner Hill-climbing* layer was 40 in all the test cases. Table 6 describes the test cases for simple substitution ciphers.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
1	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27
2	26 English characters	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26

Table 6: Simple Substitution Cipher Test Cases

6.2.2 Self-Generated Homophonic Substitution Cipher

The test cases for homophonic substitution ciphers include total 2,008 ciphers of various ciphertext sizes ranging from 300 characters to 10,000 characters, various cipher symbol sizes ranging from 27 to 100, and various number of initial starting points ranging from 1 to 100. The test cases also include plaintext letter sets of 26 English characters plus the space character and just the 26 English characters. Table 7 describes the test cases for homophonic substitution ciphers.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
1	26 English characters and Space ' '	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27 to 63
2	26 English characters and Space ' '	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	28, 35, 45, 55, 63
3	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27 to 63
4	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	28, 35, 45, 55, 63, 65, 75, 85, 95, 100
5	26 English characters and Space ' '	Inner hill-climbing with 100 initial starting points	300, 500, 700, 1000	28, 35, 45, 55, 63

6	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points and Outer hill-climbing layer	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	28, 35, 45, 55, 63
7	26 English characters	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26 to 63
8	26 English characters	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27, 35, 45, 55, 63
9	26 English characters	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26 to 63
10	26 English characters	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26, 35, 45, 55, 63, 65, 75, 85, 95, 100
11	26 English characters	Inner hill-climbing with 100 initial starting points	300, 500, 700, 1000	27, 35, 45, 55, 63
12	26 English characters	Inner hill-climbing with 40 initial starting points and Outer hill-climbing layer	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27, 35, 45, 55, 63

Table 7: Homophonic Substitution Cipher Test Cases

7 Results & Observations

This section presents the results of the test cases described in Section 6.2. For each of the test cases, the results are presented in the form of graphs representing the final scores and the success rates of correctly solved symbols.

7.1 Self-Generated Ciphers

7.1.1 Simple Substitution Cipher

This section presents the results of all test cases of simple substitution ciphers.

1. **Test Case 1 :** The description of the test case is given in Table 8

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
1	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27

Table 8: Test Case 1 For Simple Substitution Cipher

Results: The graphs in Figure 9 present the results for test case 1.

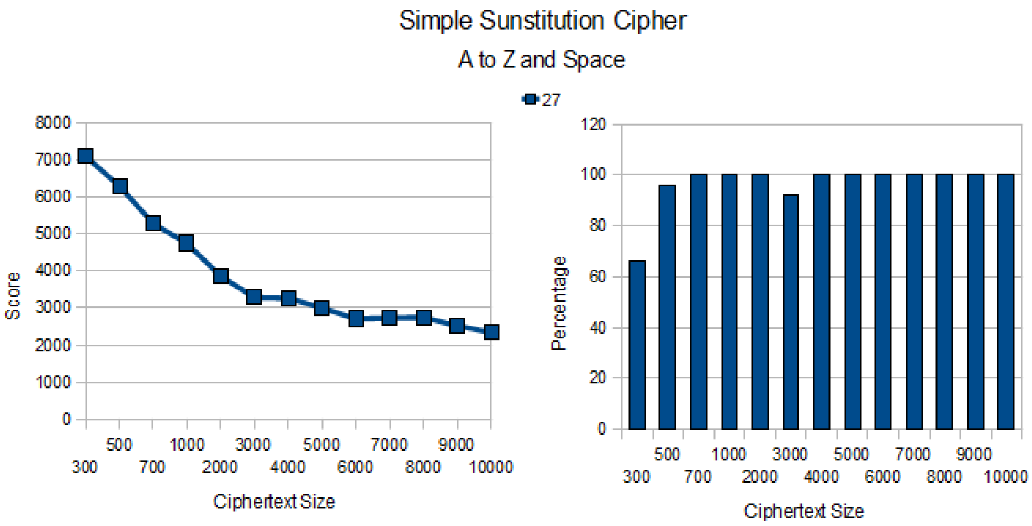


Figure 9: Simple Substitution Cipher Test Case 1 Results

Observations: It can be clearly seen from the graphs that, as the size of the ciphertext increases, the score decreases and the percentage of correctly solved symbols increases. This is mainly because, larger size of ciphertext provides better statistics of the cipher symbol frequencies.

2. **Test Case 2:** The description of the test case is given in Table 9.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
2	26 English characters	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26

Table 9: Test Case 2 For Simple Substitution Cipher

Results: The graphs in Figure 10 present the results for test case 2.

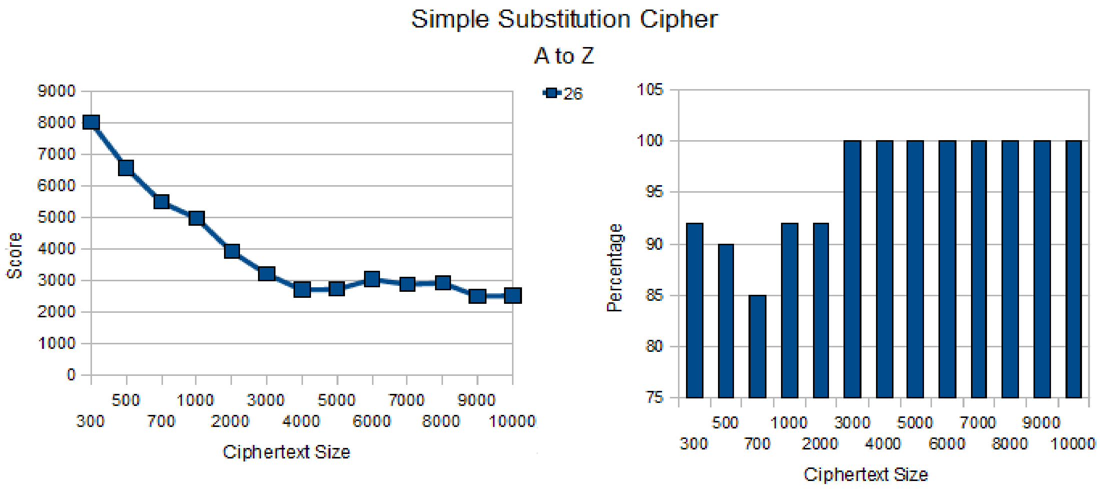


Figure 10: Simple Substitution Cipher Test Case 2 Results

Observations: It can be clearly seen from the graphs that, as the size of the ciphertext increases, the score decreases and the percentage of correctly solved symbols increases.

7.1.2 Homophonic Substitution Cipher

This section presents the results of selective test cases for homophonic substitution ciphers. The results of the remaining test cases are presented in the Section 13.2.

1. **Test Case 1:** The description of the test case is given in Table 10.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
1	26 English characters and Space ' '	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27 to 63

Table 10: Test Case 1 For Homophonic Substitution Cipher

Results: For test case 1, **t**he graph in Figure 11 presents the results in terms of final score and the graph in Figure 12 presents the results in terms of success rate.

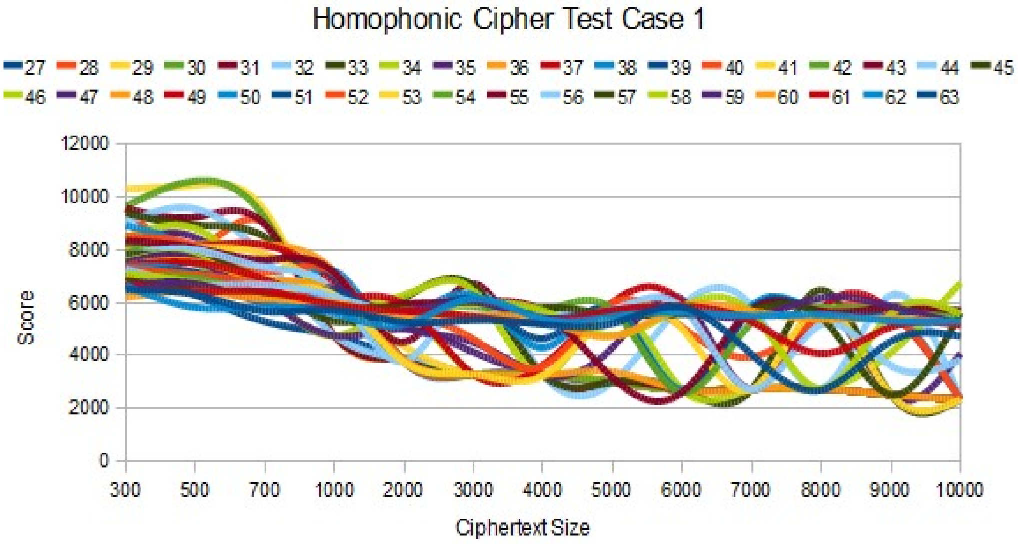


Figure 11: Homophonic Substitution Cipher Test Case 1 Results - Score

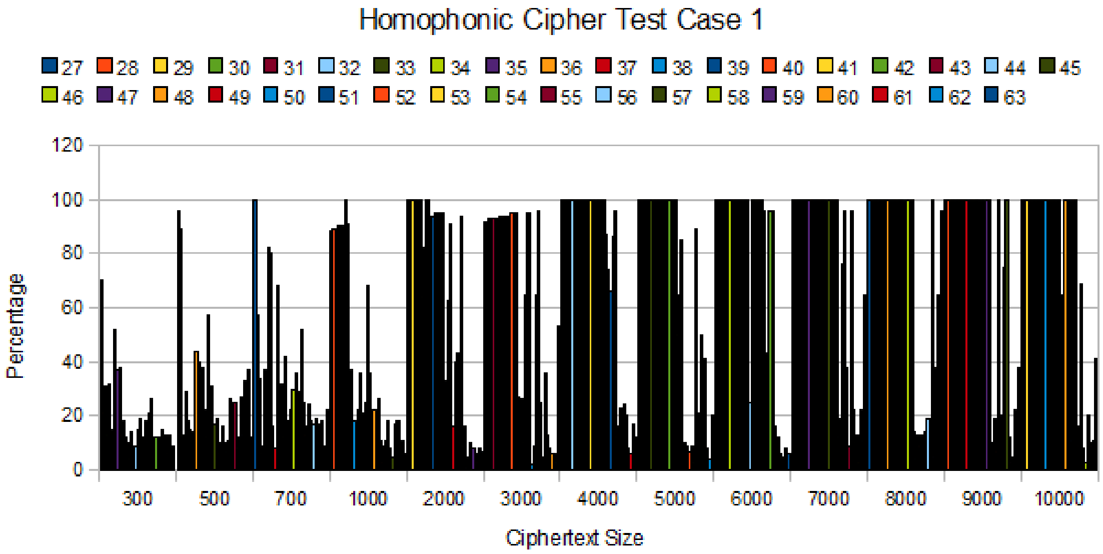


Figure 12: Homophonic Substitution Cipher: Test Case 1 Results – Success Rate

Observations: Overall, a pattern can be seen in the scores of the various ciphertext instances. For all of the cipher symbol sizes, the scores tend to decrease as the ciphertext size increases. However, there are two major issues seen in these graphs. First, the ciphertexts with lesser sizes display poorer scores and the poorer percentages of the correctly solved symbols and second, for some instances of the cipher symbol sizes, a "sine wave" like behavior is seen in the Figure 11, that is, even with increasing size of ciphertext, scores change periodically from better to worse and from worse to better.

2. **Test Case 2:** This test case is a sub set of the Test Case 1. In this test case, only selected instances of cipher symbol sizes are considered. The detailed description of the test case is given in Table 11.

Testcase ID	**Plaintext Letter Set**	**Specific Case**	**Ciphertext Size (in Bytes)**	**Cipher Symbol Size**
2	26 English characters and Space ' '	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	28, 35, 45, 55, 63

Table 11: Test Case 2 For Homophonic Substitution Cipher

Results: For test case 2, **t**he graph in Figure 13 presents the results in terms of final score and the graph in Figure 14 presents the results in terms of success rate.

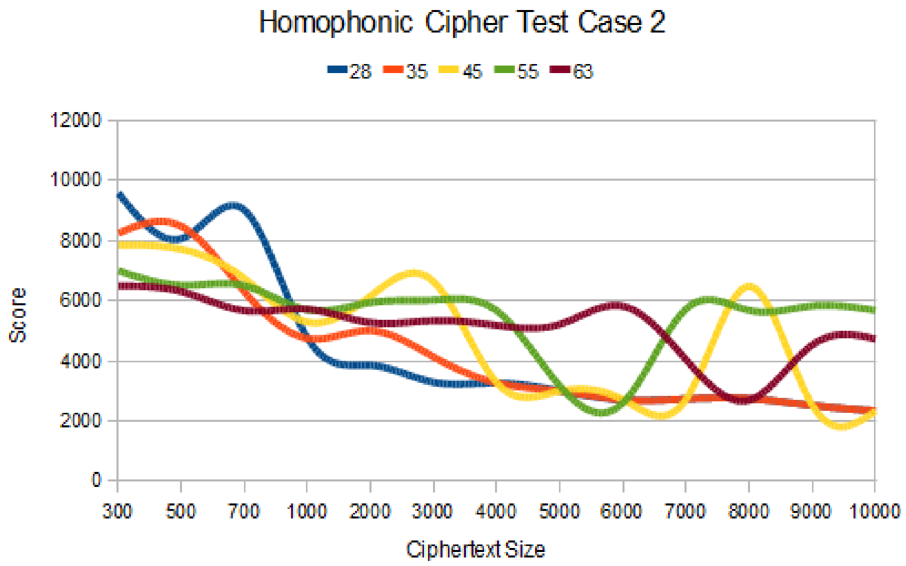


Figure 13: Homophonic Substitution Cipher: Test Case 2 Results - Score

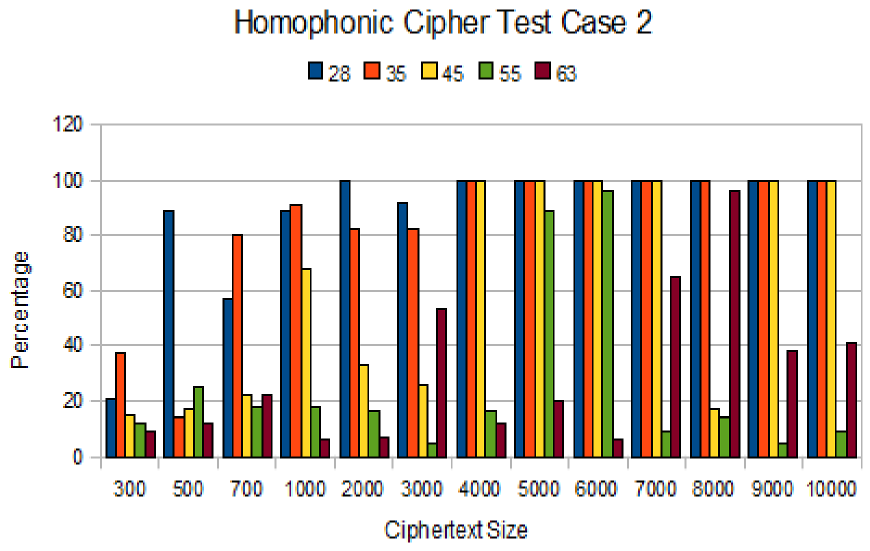


Figure 14: Homophonic Substitution Cipher: Test Case 2 Results - Success Rate

Observations: As this test case is a subset of the Test Case 1, the observations are the same. Only difference is that, limited instances of cipher symbol sizes make it easier to observe and analyze.

3. **Test Case 3:** The description of the test case is given in Table 12.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
3	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27 to 63

Table 12: Test Case 3 For Homophonic Substitution Cipher

Results: For test case 3, **t**he graph in Figure 15 presents the results in terms of final score and the graph in Figure 16 presents the results in terms of success rate.

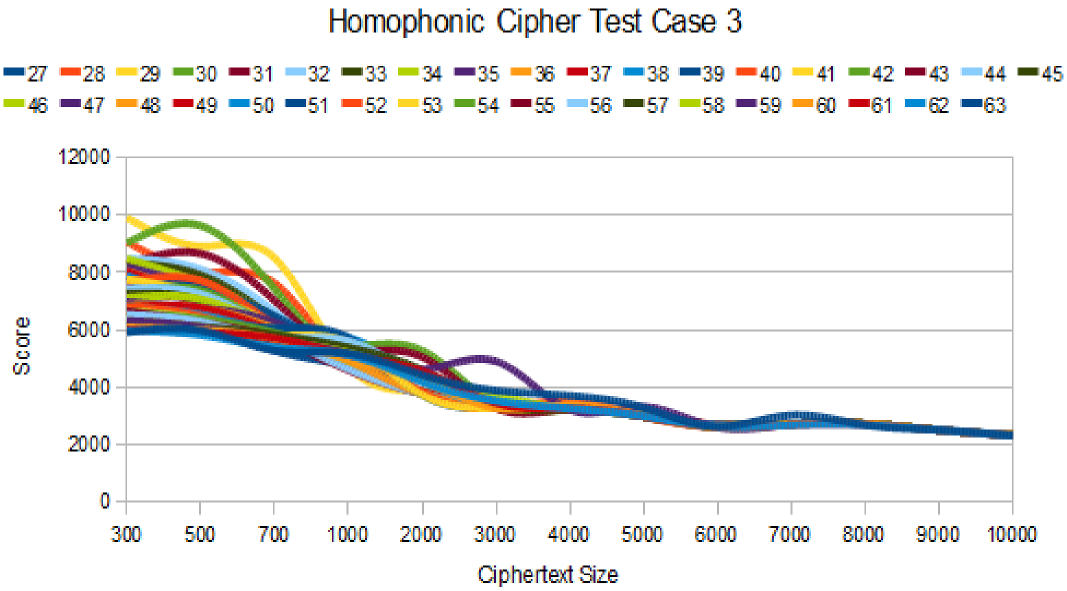


Figure 15: Homophonic Substitution Cipher: Test Case 3 Results - Score

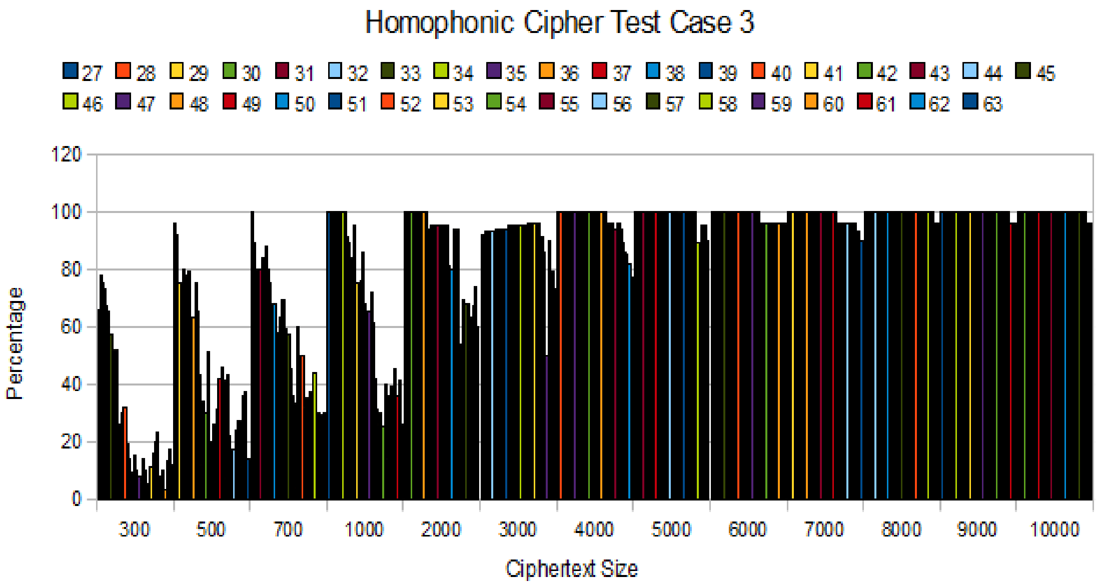


Figure 16: Homophonic Substitution Cipher: Test Case 3 Results - Success Rate

Observations: After increasing the number of input solutions to inner hill-climbing layer from 1 to 40, a noticeable change is seen in the graphs. A neat pattern is seen in the scores of the ciphertext instances. For all of the cipher symbol sizes, the scores converge into one point as the ciphertext size increases. The only issue in this test case is that, the ciphertexts with lesser sizes display poorer scores and the poorer percentages of the correctly solved symbols

4. **Test Case 4:** The description of the test case is given in Table 13.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
4	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	28, 35, 45, 55, 63, 65, 75, 85, 95, 100

Table 13: Test Case 4 For Homophonic Substitution Cipher

Results: For test case 4, **t**he graph in Figure 17 presents the results in terms of final score and the graph in Figure 18 presents the results in terms of success rate.

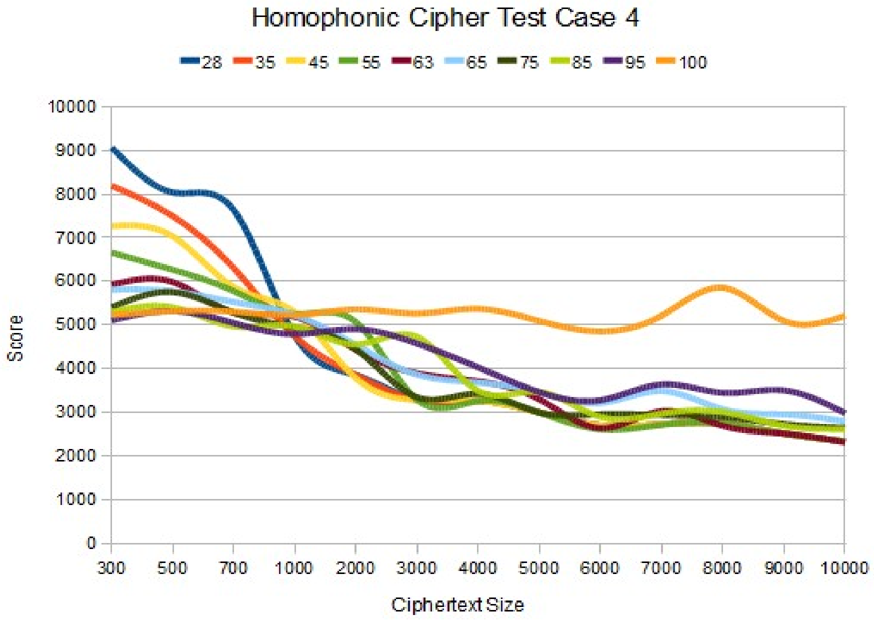


Figure 17: Homophonic Substitution Cipher: Test Case 4 Results - Score

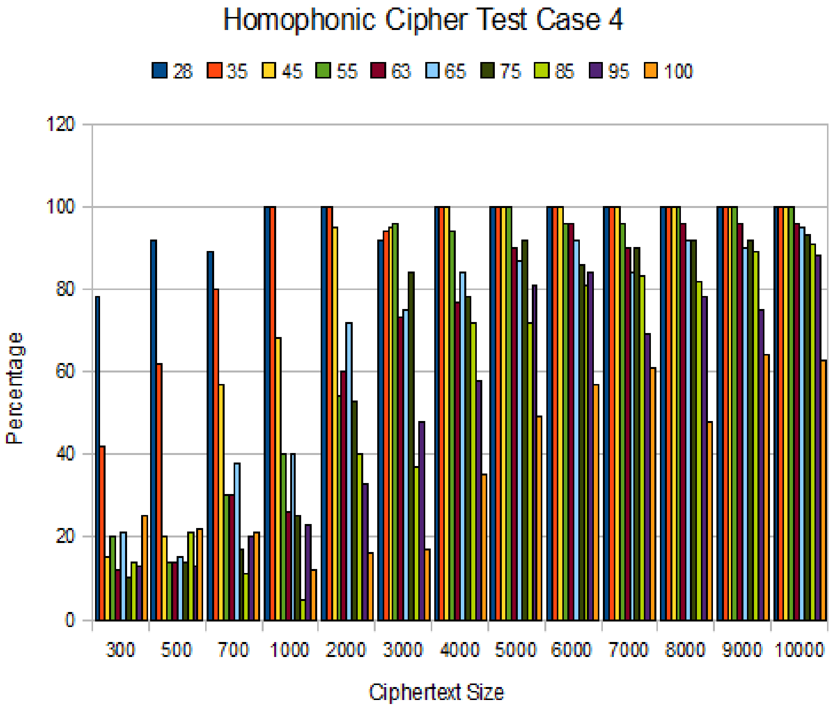


Figure 18: Homophonic Substitution Cipher: Test Case 4 Results – Success Rate

Observations: As this test case is similar to that of Test Case 3, the observations are the same. Only differences here is that, some more instances with higher cipher symbol size are included. From these graphs, it is clearly seen that, as the number of cipher symbol size increases, the score and the percentage of correctly solved cipher symbols decreases.

5. **Test Case 5:** This test case focuses on testing the effect of increasing the number of initial starting points solutions on the ciphertexts with lesser sizes. The description of the test case is given in Table 14.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
5	26 English characters and Space ' '	Inner hill-climbing with 100 initial starting points	300, 500, 700, 1000	28, 35, 45, 55, 63

Table 14: Test Case 5 For Homophonic Substitution Cipher

Results: For test case 5, **t**he graph in Figure 19 presents the results in terms of final score and the graph in Figure 20 presents the results in terms of success rate.

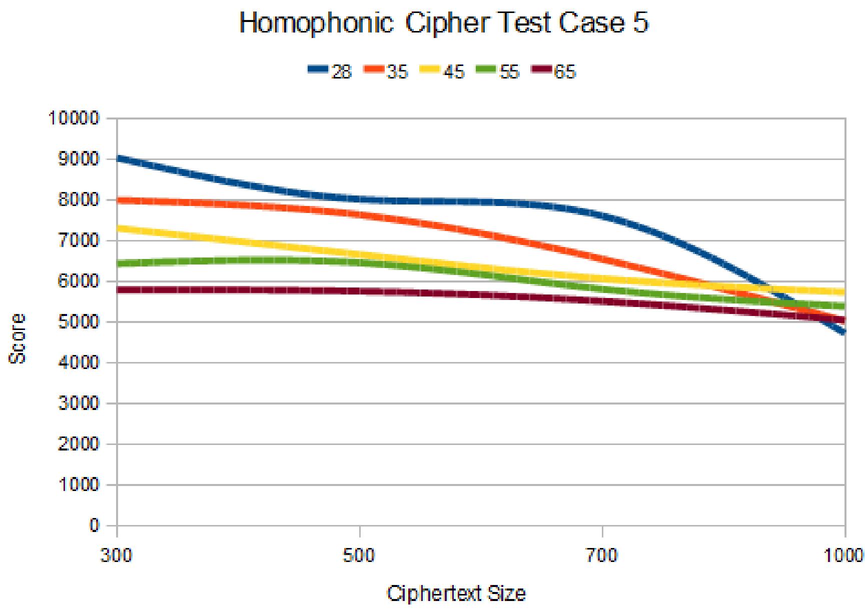


Figure 19: Homophonic Substitution Cipher: Test Case 5 Results - Score

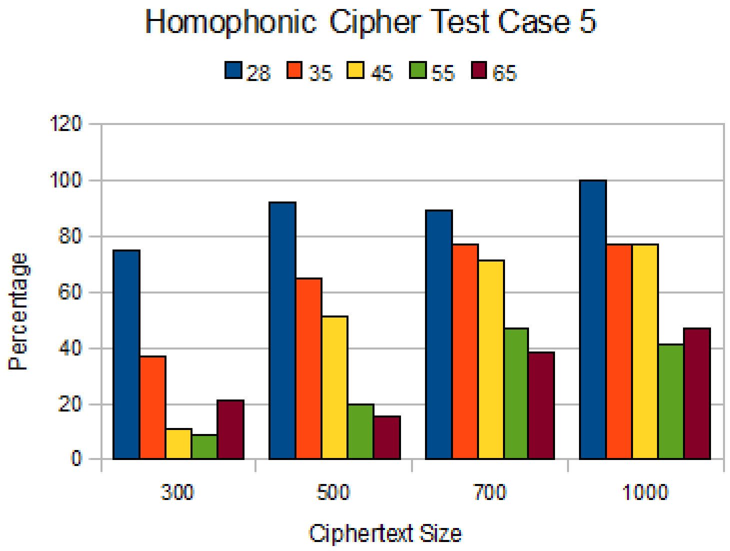


Figure 20: Homophonic Substitution Cipher: Test Case 5 Results – Success Rate

Observations: In this test case, the number of input solutions to *Inner hill-climbing* layer is increased to 100 from 40. The scores and percentages for ciphertexts with lesser sizes are improved slightly because of the increased in the number of input solutions.

6. **Test Case 6:** The description of the test case is given in Table 14.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
6	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points and Outer hill-climbing layer	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	28, 35, 45, 55, 63

Table 15: Test Case 6 For Homophonic Substitution Cipher

Results: For test case 6, **t**he graph in Figure 21 presents the results in terms of final score and the graph in Figure 22 presents the results in terms of success rate.

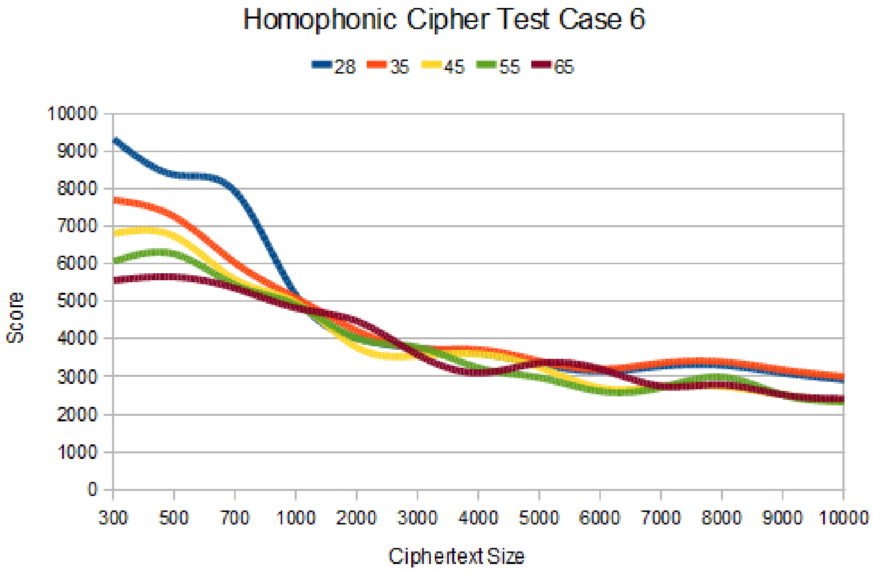


Figure 21: Homophonic Substitution Cipher: Test Case 6 Results - Score

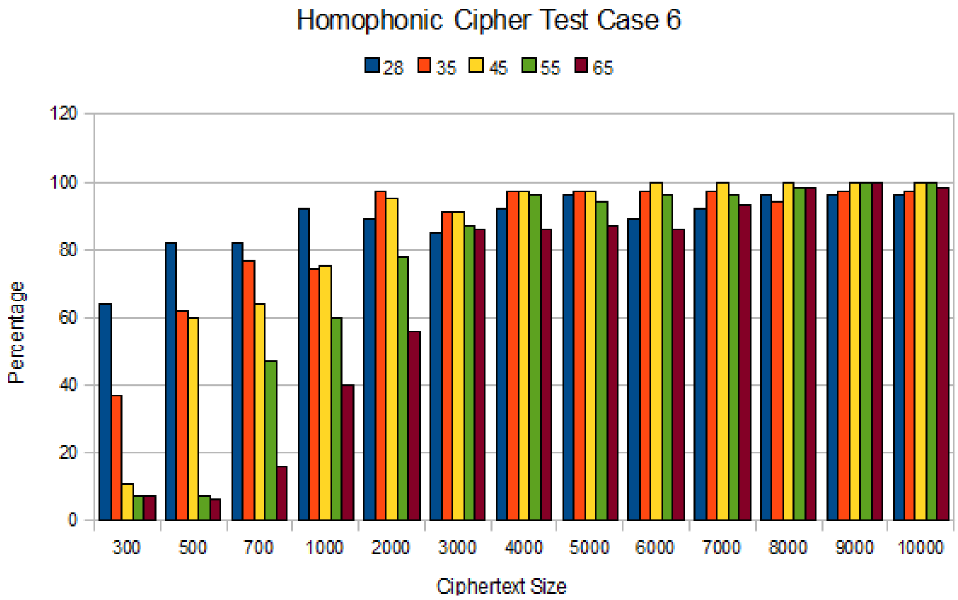


Figure 22: Homophonic Substitution Cipher: Test Case 6 Results – Success Rate

Observations: Including the outer hill-climbing layer gives similar results to that of the test case 3. A neat pattern is seen in the scores of the ciphertext instances. For all of the cipher symbol sizes, the scores reduce drastically as the ciphertext size increases. The only issue in this test case is that, the ciphertexts with lesser sizes display poorer scores and the poorer percentages of the correctly solved symbols.

8 Analysis

After studying the results of the test cases, we found that, factors such as ciphertext size, cipher symbol size, and the numbers of initial starting points play an important role in determining the probability of finding the best feasible solution. Some of these factors apply to both simple and homophonic substitution ciphers and some apply to only homophonic substitution ciphers.

The factors affecting both simple and homophonic substitution ciphers are ciphertext size and number of initial starting points. As the size of the ciphertext increases, the score decreases and the percentage of correctly solved symbols increases. This is mainly because, larger size of ciphertext provides better statistics of the cipher symbol frequencies. Ciphertext size is directly proportional to the percentage of correctly solved symbols. For the number of initial starting points, higher number of input solutions to the *Inner Hill-climbing* layer provide higher number of local optimum solutions to choose from. Thus, higher numbers of initial starting points increase the probability of finding a better solution.

The factor which is specific to homophonic substitution cipher is the cipher symbol size. As the cipher symbol size increases, the plaintext frequencies are more and more flattened in the ciphertext making it more difficult to solve. Lesser the cipher symbol size, higher is the probability of solving more number of cipher symbols correctly. Cipher symbol size is inversely proportional to the percentage of correctly solved symbols.

The Figure 23 displays a 3 dimensional graph summarizing the relation between ciphertext size, ciphertext symbol size, and the success rate. The X-axis represents ciphertext symbol size, the Y-axis represents the ciphertext size, and the Z-axis represents the success rate. It can be clearly seen from the graph that, for lower values of ciphertext size and higher values of ciphertext symbol size, the success rate is lowest. As the ciphertext size increases and the ciphertext symbol size decreases, the success rate increases and stabilizes for ciphertext sizes greater than 6000 and ciphertext symbol sizes less than 55 approximately.

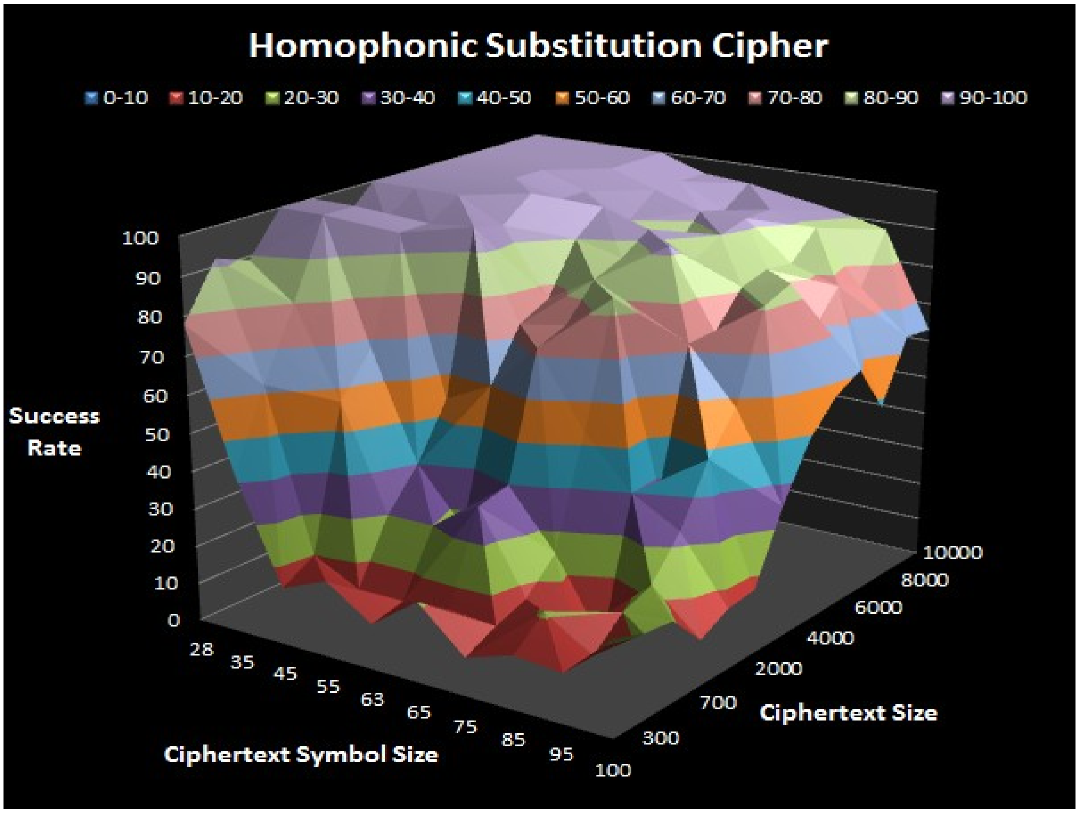


Figure 23: 3D Graph Of Results Summary

9 Zodiac Ciphers

Zodiac was a serial killer in San Francisco Bay Area in 1960-70s [2]. He killed several people mainly in lonely areas. He sent letters, cards, and ciphers to local newspapers such as *“San Francisco Chronicle”, “San Francisco Examiner”* and *“Vallejo Times-Herald”* to take credit for the murders. He claimed to have murdered 37 people. However, the San Francisco Police Department(SFPD) verified only 7 victims (5 killed and 2 inured). He adopted the name Zodiac and never openly revealed his true identity [20]. He did claim that his identity was included in one of the ciphers he created. He created and sent total 4 ciphers to the local newspapers. His first cipher, the Zodiac 408 was broken within a week after getting published in the newspapers. His later ciphers are still not broken and his identity still remains unknown. Zodiac created total 4 ciphers [2]. Zodiac's two famous ciphers are described below.

9.1 Zodiac 408 Cipher

The Zodiac 408 cipher was divided into three parts and each part was sent separately to the local newspapers. This cipher was broken within a week after getting published in the newspapers [16]. The Zodiac 408 cipher is displayed in Figure 24.

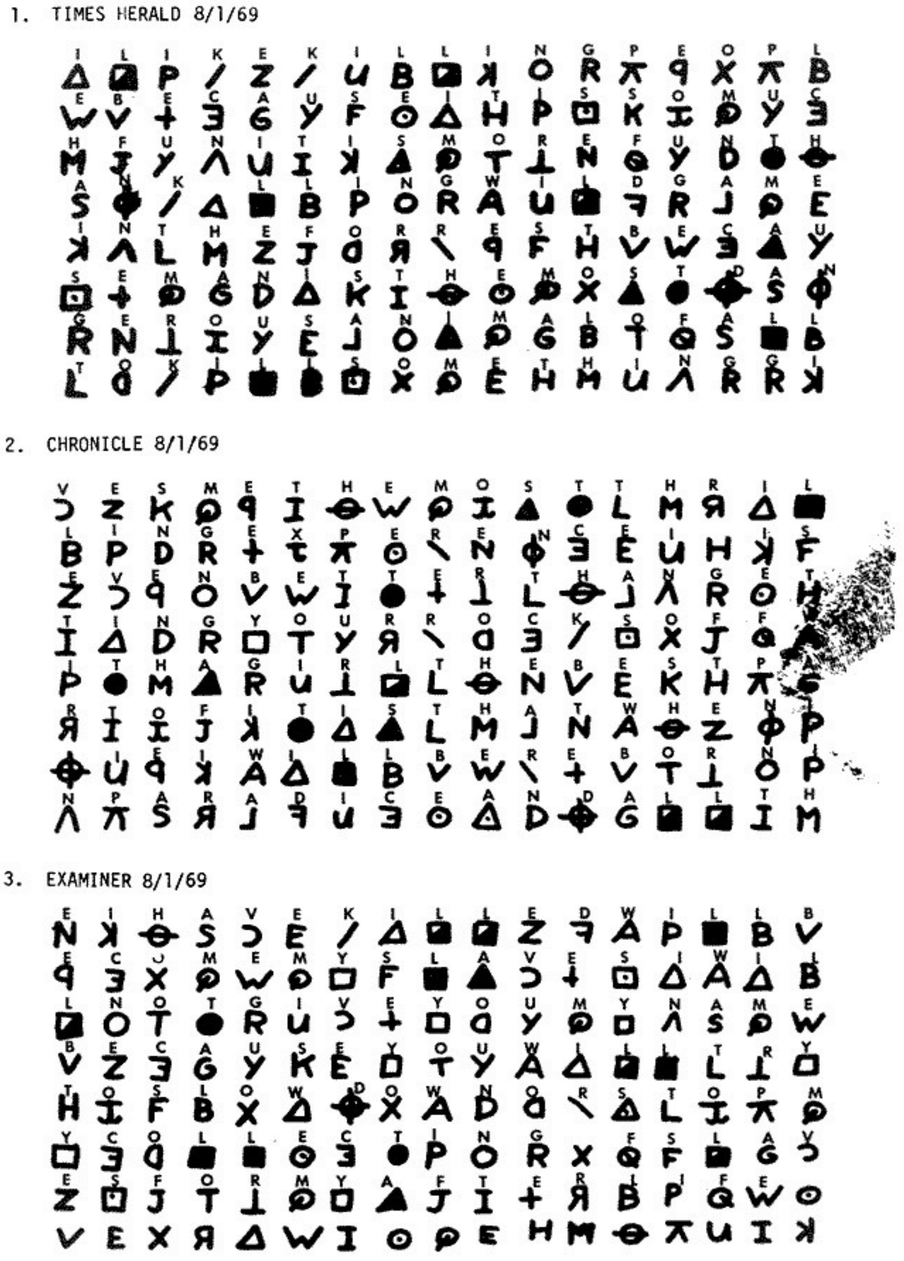


Figure 24: Zodiac 408 Cipher [16]

Solution of the Zodiac 408 cipher [16] is given below

> *“I like killing people because it is so much fun It is more fun than killing wild game in the forrest because man is the most dangerous anamal of all To kill something gives me the most thrilling experience It is even better than getting your rocks off with a girl The best part of it is that when I die I will be reborn in paradice and all the I have killed will become my slaves I will not give you my name because you will try to slow down or stop my collecting of slaves for my afterlife”*

Frequency distribution of the Zodiac 408 cipher is given in the Table 16.

e	I	t	l	o	s	a	n	r	m	h	g	f	u	c	y	w	p	b	v	k	d	x
52	40	35	31	26	23	23	22	16	16	16	12	11	10	10	8	7	7	7	6	6	6	1

Table 16: Letter Frequency Analysis Of Zodiac 408 Cipher [16]

9.2 Zodiac 340 Cipher

The Zodiac 340 cipher is the most famous Zodiac cipher which is still a mystery. This cipher was mailed to local news papers on November 8, 1969. This cipher consisted of 340 characters. The Zodiac 340 Cipher is displayed in Figure 25 [14]

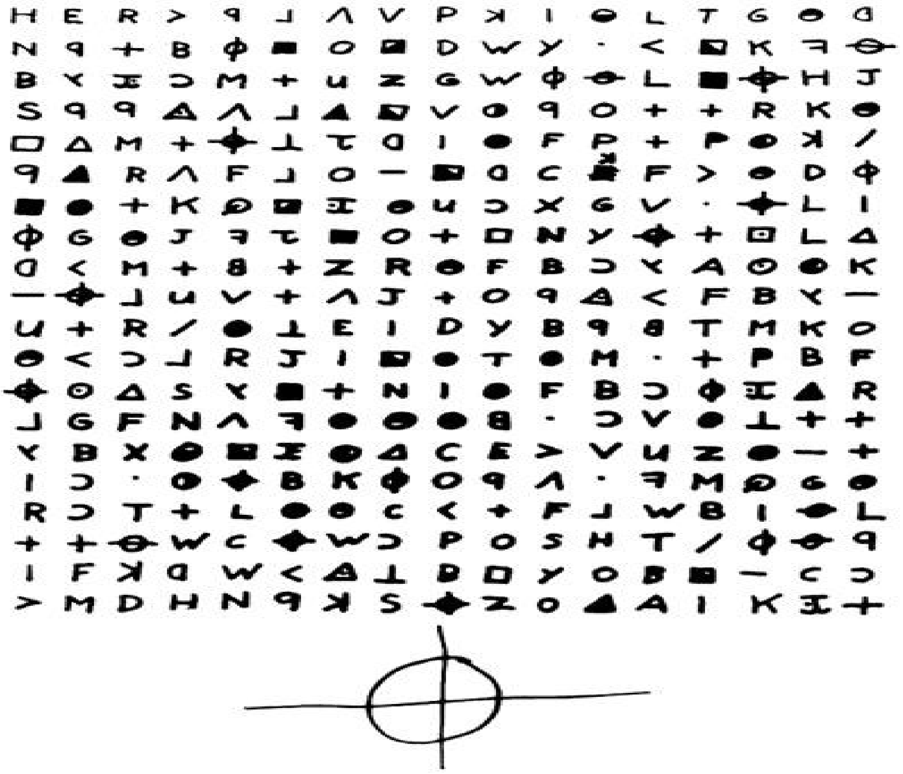


Figure 25: Zodiac 340 Cipher [14]

There have been many attempts made for cracking the Zodiac 340 cipher and some have claimed to have broken the cipher. However, none of the solutions proposed so far have been accepted. Some of the interesting attempts are given as follows.

Christopher Farmer from National Security proposed a solution of Zodiac 340 cipher based on a Japanese play called "Mikado" and the theory of "Radians" [18]. In an another attempt, Mr. Jos Kirps proposed a similarity between Zodiac 340 cipher and a Halloween card sent by Zodiac to one of the newspapers [4]. Mr. Kirps suggested that the cipher could be divided into four parts as shown on the Halloween card and that some of the words given on the card could be repeated from the actual cipher. The Halloween card is displayed in Figure 26.

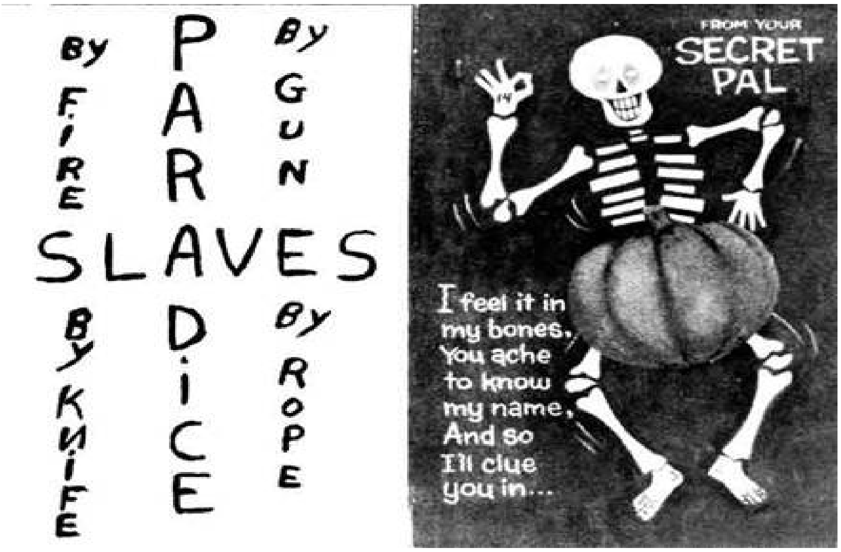

By FIRE
PARADICE
By GUN
SLAVES
By KNIFE
By ROPE
FROM YOUR
SECRET PAL
I feel it in my bones, You ache to know my name, And so I'll clue you in...

Figure 26: Zodiac Halloween Card [1]

9.2.1 Tests & Results

The test cases and results for Zodiac 340 cipher are described in Table 17.

Testcase ID	Plaintext Letter Set	Specific Case	Score	Putative plaintext
1	26 English	Inner hill-climbing	5985	*“presentmentntntntntntntntntntntntntntntnt ntntntntntntntntntntntntntntntntntntntntntnt*

	characters and Space ' '	with 40 initial starting points		*nt in y fisie udqs batcnsaethr iway anulpe ss te celsdiimn rd iloof thnine as mthed cfo he s utinjqg wrxyealn y esoudiriblizndft ihiam hertkven leweiteids thet wimator sbesh nd treme ct t ainehlvd tuii ther g meyhitst tharetoiitexichedoreewat i rallen dstas jy mr int otihe e aniia ol rnd p a as h f t oerbdec ore spis lad k ngi”*
2	26 English characters and Space ' '	Inner hill-climbing with 40 initial starting points and outer hill-climbing layer	5708	*“moouchishathneanr c ldex stiger i ak d waslindome ccoihreszce ornsed ofjrt th healcroitheyer atun ld r xan pasiontlane jde s to qnergd b wont aiveryoh s ie ecogt ay ol fot t cbedreng hoete e di h tove ad t t larohat i n bi s f a phekee ous w y t izo rlecii d ano e n n g ths tin is os he mellicttarsgof ste ey ud m ca oweritra”*
3	26 English characters	Inner hill-climbing with 40 initial starting points	6092	*“ssunolldfrhaemerttotsthiveiwaneryhsieagtebeithehastdooilloedroitturrongtanttheeftforsooulelicetorenrethetrkvareapedaaehtertythitotwatzentngtxtburesaiqjorcaledtlttioinesicetusenshewsoxmgrirnalutheemegatfseajndihttheesataouleetlyerexaadenttispaeeonosndebecthaarasrtiolaygkeruamteerontelishhetthioaiafidsmsthohertininsowisecoangestordabioqhrat”*
4	26 English characters	Inner hill-climbing with 40 initial starting points and outer hill-climbing	5868	*“passturinytistrefotinatlkealdethenngidwineraanstmpbatthruetiqtliishecowimerfthonintystesrouletfryoseeathihakvendoridmstarebertlicolmizsofewiaieseondgxjthemuniirbiltheongenissheatelntatwhleedusbttththwdinnomjoagtiothondavesuroorehehaddiheiignoititorasineheitddqmnhaltrdewaresdtishereiouantnsiinarmadnlaptsanttoyfaehencllnterdswepotyamelexthvi”*

Table 17: Zodiac 340 Cipher Test Cases

9.3 Fake Zodiac Cipher

We used the attack to break a fake zodiac cipher posted on the MysteryTwister website [21]. We designed various test cases for attacking the fake cipher. Along with the standard English language digram statistics; we also considered the digram statistics of Zodiac 408 solution as the expected digram statistics. The size of the fake cipher is 340 and the cipher symbol size is 65. The results are given in Table 18

Testcase ID	Score	Putative plaintext
1	5567	*orevcochinnoasabexdeteoghetrytiejealounleytansismayankerirntiorftdthonaholatthgldotestheduosthspattfenoenassaueperandeelivehishectyemoonsegecahetigatyreerontoqqhtsuthelesfrioichedcridarmembendfothanfarilyoeadsconereficehdithecprgiassseieeimaloesorhotrmdtedhedstorisineratrtagrhifainingthaprearestlomthlshesotniterdithrronccdcanretopothaqclr*
2	4229	*ilidetillingreopldbecaeseittisomalezenitiimofeaunthanjillttngwildgameinthezorrestbecausemanisthenostdangefousaninalosalltodillsomethinggivesmethemostthrillingexqerenceitisdteiwillbereborninpafadiceandalltheieavetilledwillbecomenrslavesiwillnotgiveroemrnamebelauserouwilltrrtoslowdownofstornrlollectingozslavessormraaterliseebeoftetenetheqitt*

Table 18: Fake Zodiac Cipher Test Cases

Using the putative plaintext obtained as a result of testcase ID 2, we could solve the complete cipher manually. The solution is given below:

"I like killing people because it is so much fun it is more fun than killing wild game in the forrest because man is the most dangerous animal of all to kill something gives me the most thrilling experence it is die i will be reborn in paradice and all the i have killed will become my slaves I will not give you my name because you will try to slow down or stop my collecting of slaves for my afterlife ebeorietemethhpiti"

We also modified the outer hill climb to thoroughly check more number of possible frequency distribution mappings. We designed a slow outer hill climb; where instead of having just one round of modifying the adjacent elements of the frequency distribution mapping; we had multiple rounds until no modification in a round produced better results. With this outer hill climb; every attack had atleast one round of frequency distribution modification. The results of the current round decided if the next round was conducted or not. That is; in a given round; if atleast one modification was found which gave better results; the next round was conducted. Thus, multiple rounds were carried out until no modification gave better results. We designed total four test cases to check the effect of modified outer hill climbing module. At this point, since we already knew the actual plaintext, we computed the percentage of correctly solved symbols. The test results with the slow outer hill climb are given in Table 19.

Testcase ID	Description	Percentage
1	Original outer hill climb with standard english statistics	13.00%
2	Slow outer hill climb with standard english statistics	4.00%
3	Original outer hill climb with Zodiac 408 solution statistics	70.00%
4	Slow outer hill climb with Zodiac 408 solution statistics	84.00%

Table 19: Test Results Of Fake Cipher With Slow Outer Hill Climb

As seen in the Table 19, the results of the testcase ID 2, gave worst percentage value. The reason for this behaviour is the standard English language statistics. Since, the digram statistics of the fake cipher do not match with the standard english statistics, the slower outer hill climb tried to bring the putative plaintext of the cipher closer to the standard English, thus effectively making it more different from the actual solution. Therefore, we got lesser percentage value when compared against the actual solution. On the other hand, in test cases 3 and 4; when the Zodiac 408 solution digram statistics were used; we got better results with the slow outer hill climb.

10 Conclusion

We designed and implemented an efficient attack on the homophonic substitution ciphers. The attack is based on the hill-climbing heuristic technique. The proposed algorithm has a multi-layered architecture with three nested loops to solve the challenges imposed by the homophonic substitution ciphers and the hill-climbing technique. The algorithm was successfully tested on simple substitution ciphers and many instances of homophonic substitution ciphers with variable ciphertext sizes and cipher symbol sizes. It gave positive results for more than 90% of the test cases. The algorithm was able to break at least 80% of cipher symbols for the ciphertexts having minimum 1000 characters and maximum 42 cipher symbols. For the ciphertexts having minimum 3000 characters and maximum 75 cipher symbols, the algorithm was able to break at least 85% of cipher symbols.

11 Future Work

The outer hill-climbing loop can be improved by modifying the way iterations are carried out. Instead of having just one round of modifying the adjacent elements of the frequency distribution mapping; a method can be designed such that; every time an instance of frequency distributio mapping produces a better result; the iterations should start again with the first element along with retaining the modification to the frequency distribution mapping. Other ways of improving the outer hill climb could also be devised.

The evaluation of putative plaintext against the expected language of plaintext could be improved by using the trigram or n-gram frequencies; instead of the digram frequencies. For generating random initial starting points, other heuristic methods could also be used; such as Simulated Annealing or Genetic Algorithms [10]. The Simulated Annealing technique can help in generating starting points by using randomized neighborhood search. The Genetic Algorithms can help in constructing starting points by mutating selective local optimum solutions, in order to obtain high quality starting points.

12 References

[1] Announcing the Advanced Encryption Standard(2001). *Federal Information Processing Standards Publication,* v. 197.

[2] 340-cipher – Overview and Examination. Retrieved: November 18, 2011 from website: http://www.zodiackiller.com/mba/zc/69.html

[3] Briggs, K(2004). *English and Latin digram and trigram frequencies.* Retrieved: September 14, 2011, from Website: http://keithbriggs.info/site-map.html

[4] Kirps, J. *Cracking the Zodiac Killer's 340 Cipher.* Retrieved: March 20, 2011, from website http://www.kirps.com/web/main/resources/various/zodiac340/

[5] Mathai, J. *History of Computer Cryptography and Secrecy System.* Retrieved: November 18, 2011 from website: http://www.dsm.fordham.edu/~mathai/crypto.html

[6] *Introduction To Codes, Ciphers, and Codebreaking(2010).* Retrieved: November 22, 2011, from Website: http://www.vectorsite.net/ttcode_01.html#m2

[7] Jakobsen, T(1995). A Fast Method for the Cryptanalysis of Substitution Ciphers. *Cryptologia,* v 19, 265-274

[8] Kahate, A. (2008). *Cryptography and Network Security,* (2nd ed)

[9] Knight, K.(2011). Bayesian Inference for Zodiac and other Homophonic Ciphers. *Association For Computation linguistics,* pages 239-247

[10] Kreher D., Stinson D(1999). *Combinatorial Algorithms*

[11] *Local Obstacle Avoidance and hill-climbing.* Retrieved: November 14, 2011, from website:http://www.gameplaydev.com/2010/08/local-obstacle-avoidance-and-hill-

climbing/

[12] Oranckchak, D.(2008). Evalutionary Algorithm for Decryption of Monoalphabetic Homophonic Ciphers Encoded as Constraint Satisfaction Problem. *ACM,* V. 978-1-60558-131

[13] Olson, E. (2007). Robust Dictionary Attack on Short Simple Substitution Ciphers. *Cryptologia,* V. 31

[14] *Preliminary Report on Project MK-ZODIAC.* Retrieved: November 14, 2011, from website:http://mk-zodiac.com/game.html

[15] *ROT 13 – Simple Substitution Cipher.* Retrieved: November 18, 2011 from website: http://www.tech-faq.com/rot-13.html

[16] *Solved Zodiac 408 ciphers.* Retrieved: May 24, 2011, from Wikipedia website: http://wiki.zodiac-ciphers.dreamhosters.com/wiki/Solved_408-character_cipher

[17] Stamp, M. (2010). *Information Security: Principles and Practice,* (2nd ed)

[18] *The Zodiac 340 Cipher Solved.* Retrieved: March 20, 2011, from http://www.opordanalytical.com/articles1/zodiac-340.htm

[19] Zelina, I (2004). Heuristic Algorithms For Generalized Minimum Spanning Tree Problem. *International Conference of Mathematics and Informatics- ICTAMI.*

[20] *Zodiac Killer Observations.* Retrieved: May 24, 2011, from website: http://mk-zodiac.com/TheMostDangerousGameThatTheZodiacPlayed.html

[21] *Mystery Twister Challenges.* Retrieved: September 24, 2011, from website: http://www.mysterytwisterc3.org/

13 Appendix

13.1 Experiment On Modern Block Cipher AES

We conducted a small experiment on the modern block cipher AES. We took 5 blocks of plaintexts and encrypted them with AES. We then decrypted the ciphertexts with 10 putative keys progressively closer to the actual key. The Table 20 displays the results.

Key	55.00 %	60.00 %	65.00 %	70.00 %	75.00 %	80.00 %	85.00 %	90.00 %	95.00 %	100.00 %
Text 1	57	50	54	51	62	64	50	59	53	100
Text 2	50	50	47	58	56	45	56	50	56	100
Text 3	48	51	64	58	50	51	44	58	33	100
Text 4	45	55	50	50	62	50	56	60	47	100
Text 5	54	64	59	50	56	56	51	57	60	100
Average	**50.8**	**54**	**54.8**	**53.4**	**57.2**	**53.2**	**51.4**	**56.8**	**49.8**	**100**

Table 20: Results Of Experiment On AES Block Cipher

13.2 Homophonic Substitution Cipher Test Cases

7. **Test Case 7:** The description of the test case is given in Table 21.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
7	26 English characters	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26 to 63

Table 21: Test Case 7 For Homophonic Substitution Cipher

Results: For test case 7, **t**he graph in Figure 27 presents the results in terms of final score and the graph in Figure 28 presents the results in terms of success rate.

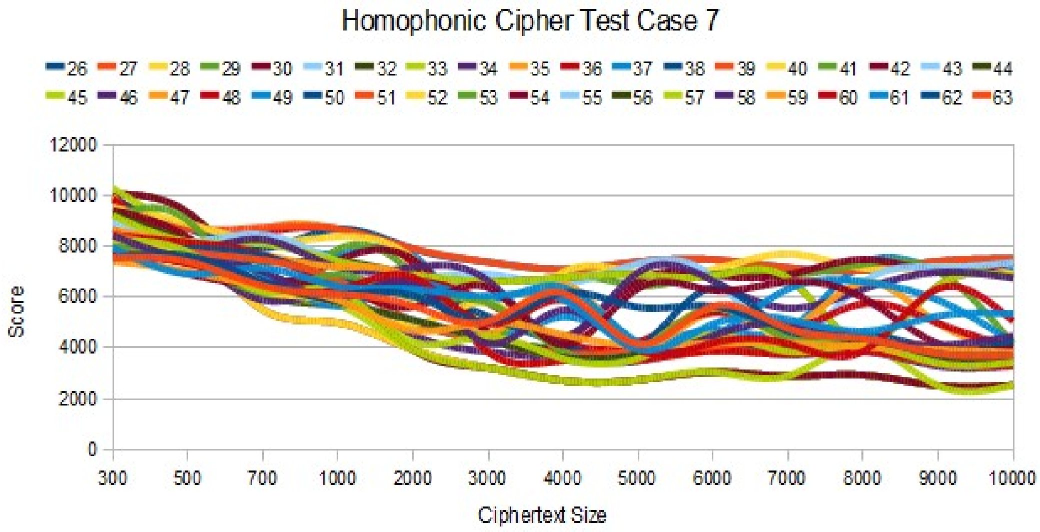


Figure 27: Homophonic Substitution Cipher: Test Case 7 Results - Score

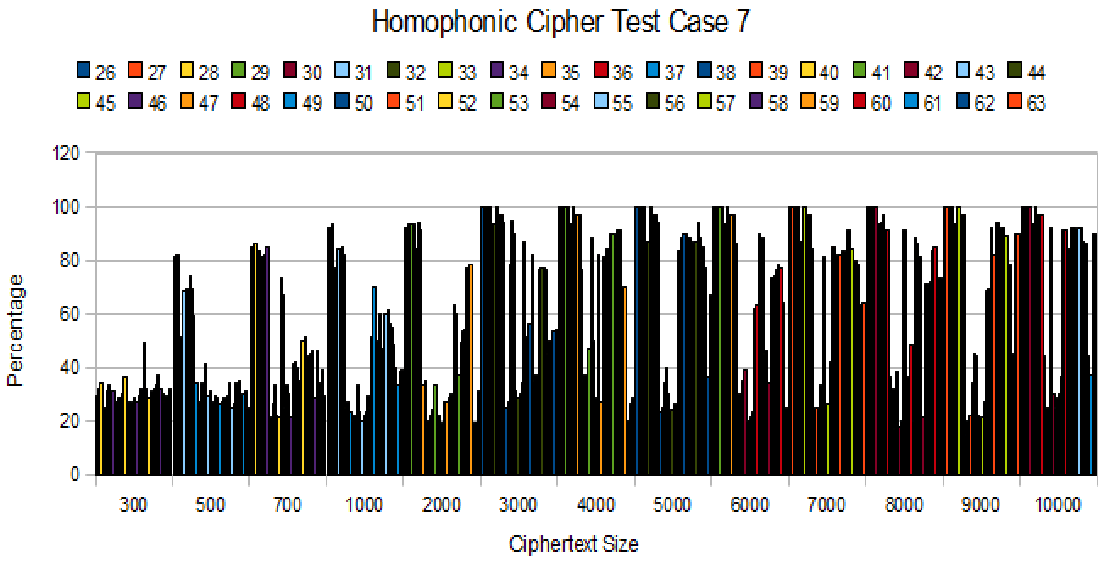


Figure 28: Homophonic Substitution Cipher: Test Case 7 Results – Success Rate

Observations: The observations are quite similar to that of Test Case 1. From the graph, a pattern can be seen in the scores of the various ciphertext instances. For all of the cipher symbol sizes, the scores tend to decrease as the ciphertext size increases. However, there are two major issues seen in these graphs. First, the ciphertexts with lesser sizes display poorer scores and the poorer percentages of the correctly solved symbols and second, for some instances of the cipher symbol sizes, a “sine wave” like behavior is seen in the Figure 26, that is, even with increasing size of ciphertext, scores change periodically from better to worse and from worse to better.

8. **Test Case 8:** The description of the test case is given in Table 22.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
8	26 English characters	Inner hill-climbing with one initial starting point	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27, 35, 45, 55, 63

Table 22: Test Case 8 For Homophonic Substitution Cipher

Results: For test case 8, **t**he graph in Figure 29 presents the results in terms of final score and the graph in Figure 30 presents the results in terms of success rate.

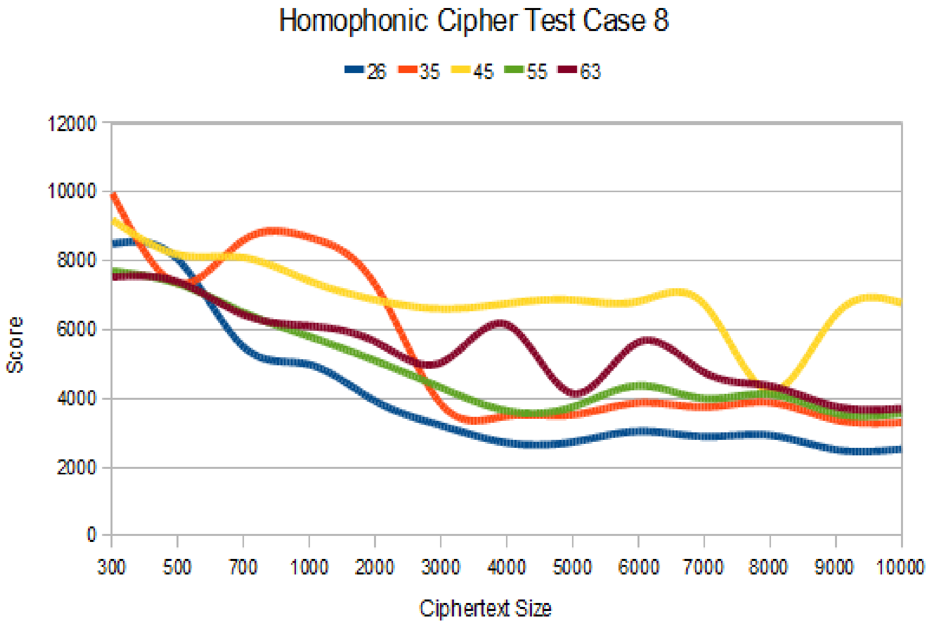


Figure 29: Homophonic Substitution Cipher: Test Case 8 Results - Score

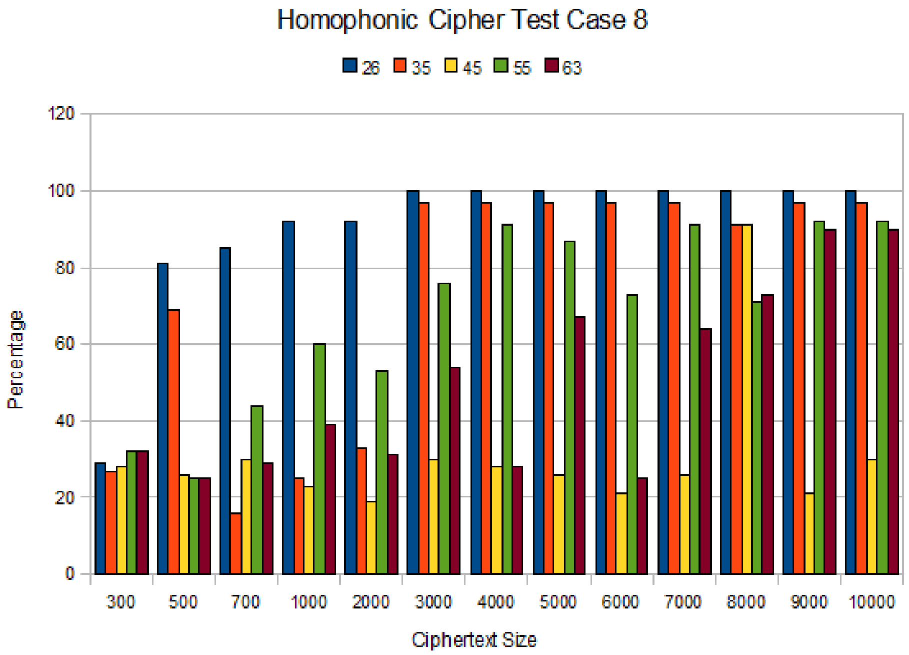


Figure 30: Homophonic Substitution Cipher: Test Case 8 Results – Success Rate

Observations: As this test case is a subset of the Test Case 7, the observations are the same. Only differences here is that, limited instances of cipher symbol sizes make it easier for observations.

9. **Test Case 9:** The description of the test case is given in Table 23.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
9	26 English characters	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26 to 63

Table 23: Test Case 9 For Homophonic Substitution Cipher

Results: For test case 9, **t**he graph in Figure 31 presents the results in terms of final score and the graph in Figure 32 presents the results in terms of success rate.

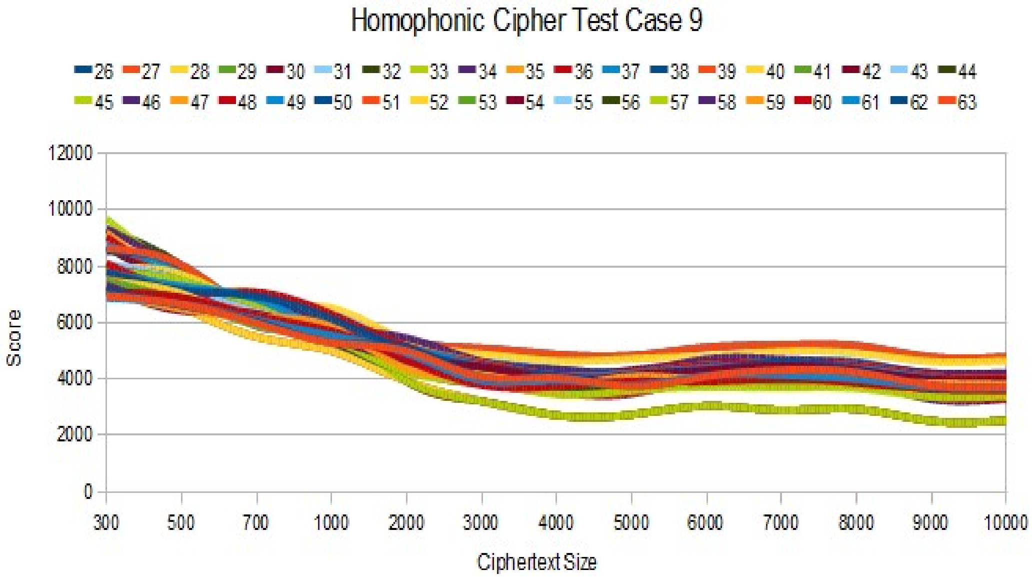


Figure 31: Homophonic Substitution Cipher: Test Case 9 Results - Score

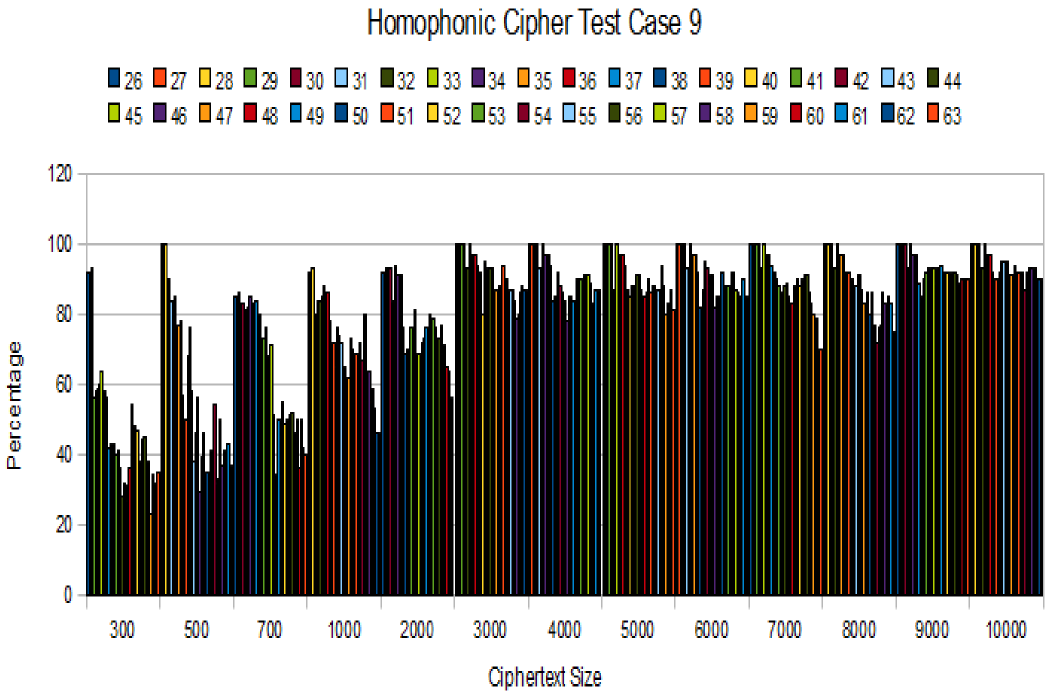


Figure 32: Homophonic Substitution Cipher: Test Case 9 Results – Success Rate

Observations: After increasing the number of input solutions to Inner hill-climbing layer to 40, a noticeable change can be seen in the graphs. A neat pattern is seen in the scores of the ciphertext instances. For all of the cipher symbol sizes, the scores converge into one point as the ciphertext size increases. The only issue in this test case is that, the ciphertexts with lesser sizes display poorer scores and the poorer percentages of the correctly solved symbols

10. **Test Case 10:** The description of the test case is given in Table 24.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
10	26 English characters	Inner hill-climbing with 40 initial starting points	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	26, 35, 45, 55, 63, 65, 75, 85, 95, 100

Table 24: Test Case 10 For Homophonic Substitution Cipher

Results: For test case 10, **t**he graph in Figure 33 presents the results in terms of final score and the graph in Figure 34 presents the results in terms of success rate.

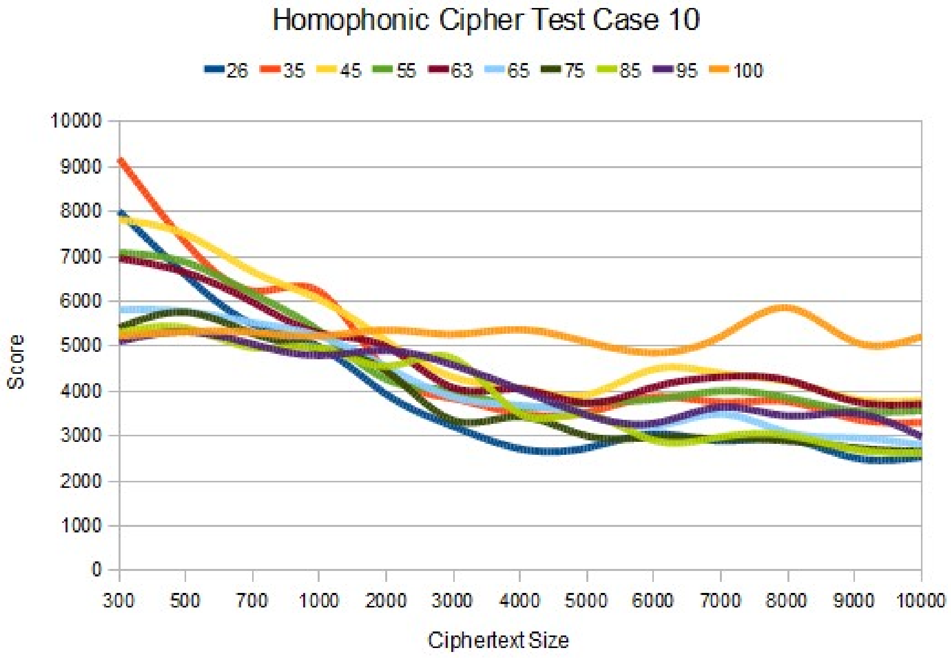


Figure 33: Homophonic Substitution Cipher: Test Case 10 Results - Score

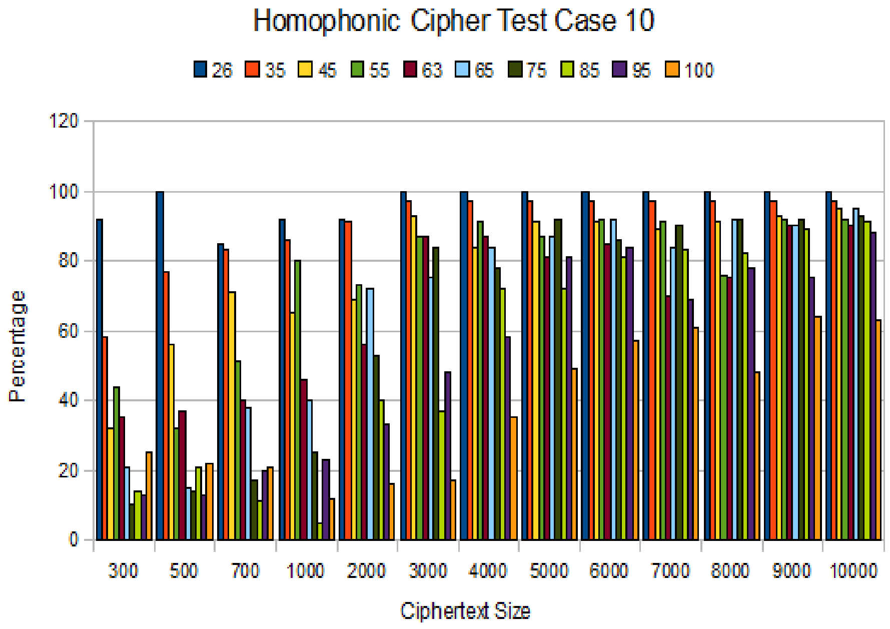


Figure 34: Homophonic Substitution Cipher: Test Case 10 Results – Success Rate

Observations: As this test case is similar to that of test case 9, the observations are the same. Only differences here is that, some more instances with higher cipher symbol size are included. From these graphs, it is clearly seen that, as the number of cipher symbol size increases, the score and the percentage of correctly solved cipher symbols decreases.

11. **Test Case 11:** The description of the test case is given in Table 25.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
11	26 English characters	Inner hill-climbing with 100 initial starting points	300, 500, 700, 1000	27, 35, 45, 55, 63

Table 25: Test Case 11 For Homophonic Substitution Cipher

Results: For test case 11, **t**he graph in Figure 35 presents the results in terms of final score and the graph in Figure 36 presents the results in terms of success rate.

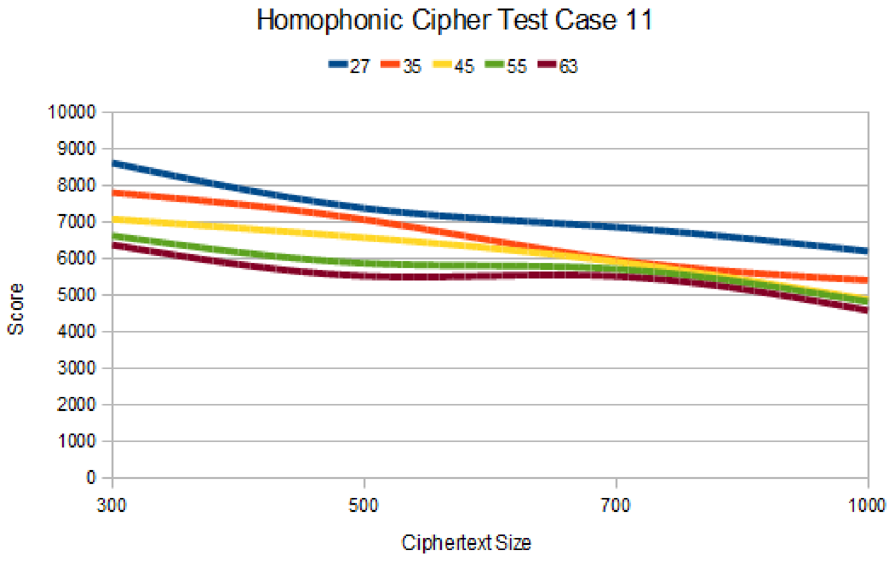


Figure 35: Homophonic Substitution Cipher: Test Case 11 Results - Score

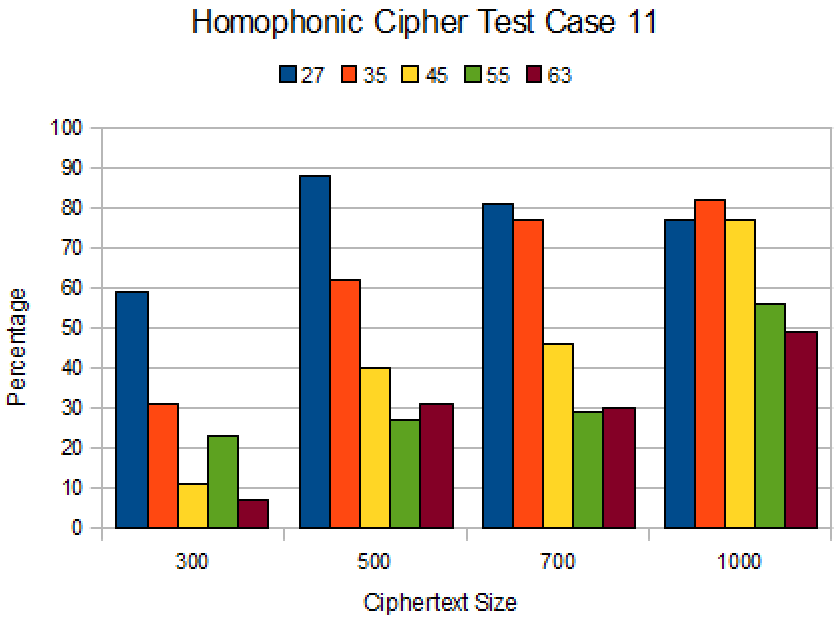


Figure 36: Homophonic Substitution Cipher: Test Case 11 Results – Success Rate

Observations: In this test case, the number of input solutions to *Inner hill-climbing* layer is increased to 100 from 40. The scores and percentages for ciphertexts with lesser sizes are improved slightly because of the increases in the number of input solutions.

12. **Test Case 12:** The description of the test case is given in Table 26.

Testcase ID	Plaintext Letter Set	Specific Case	Ciphertext Size (in Bytes)	Cipher Symbol Size
12	26 English characters	Inner hill-climbing with 40 initial starting points and outer hill-climbing layer	300, 500, 700, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 10000	27, 35, 45, 55, 63

Table 26: Test Case 12 For Homophonic Substitution Cipher

Results: For test case 12, **t**he graph in Figure 37 presents the results in terms of final score and the graph in Figure 38 presents the results in terms of success rate.

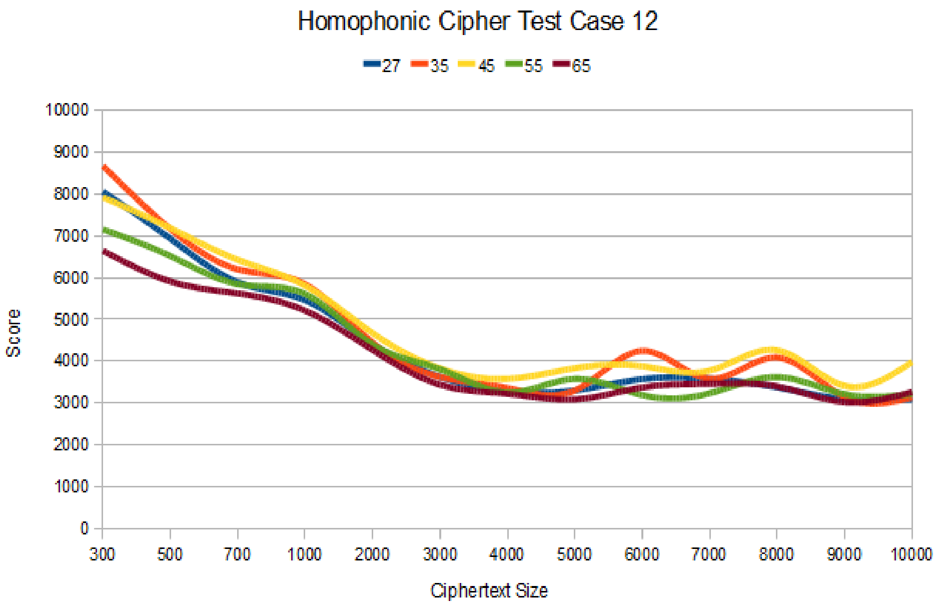


Figure 37: Homophonic Substitution Cipher: Test Case 12 Results - Score

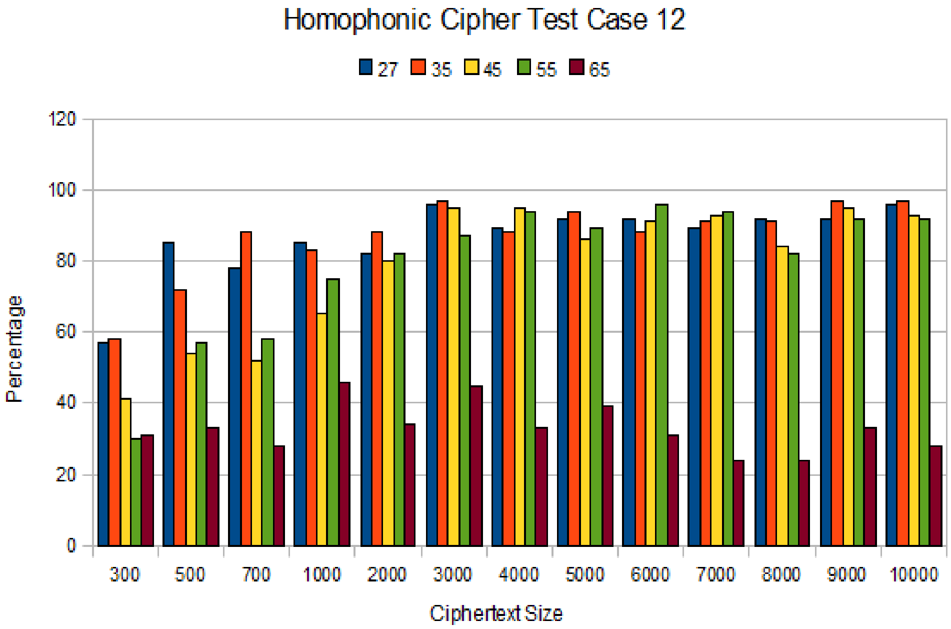


Figure 38: Homophonic Substitution Cipher: Test Case 12 Results – Success Rate

Observations: Including the outer hill-climbing layer gives similar results to that of Test Case 9. A neat pattern is seen in the scores of the ciphertext instances. For all of the cipher symbol sizes, the scores reduce drastically as the ciphertext size increases. The only issue in this test case is that, the ciphertexts with lesser sizes display poorer scores and the poorer percentages of the correctly solved symbols.