



UNIVERSITY OF NAIROBI
SCHOOL OF COMPUTING AND INFORMATICS

USING GENETIC ALGORITHM BASED MODEL IN TEXT
STEGANOGRAPHY

BY

CHRISTINE KAGONYA MULUNDA

P58/74282/2009


August 2012

Submitted in partial fulfillment of the requirements of Masters of Science in Computer
Science

DECLARATION

This research project, as presented on this report is my original work and to the best of my knowledge has not been presented for any other university award.

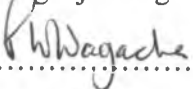
Christine Kagonya Mulunda
P58/ 74282/2009

Signed: 

Date: 15/11/2012

This project has been submitted as part of fulfillment of the requirements for the award of Masters of Science in Computer Science of the School of Computing and Informatics of the University of Nairobi, with my approval as the University Supervisor.

Prof. Peter Wainanjo Wagacha

Signed: 

Date: 15th Nov. 2012

Dedication

This project is dedicated to my husband, Alfayo Oyugi and my daughter, Kayla Akoth, for their tremendous support and understanding when I had to put in long hours and effort into this project.

Abstract

Steganography is the art of hiding information in a cover medium in such a way that the existence of any communication itself is undetectable. It can be applied in open systems such as the internet. There exist a number of steganography tools for embedding secret messages in several cover medium, but the most important property of a cover medium is the amount of data that can be stored inside it, without changing the noticeable properties of the cover, which in this case genetic algorithm approach allows variation in text length. Consequently, there is an increase in sophisticated techniques with which to analyze and recover that information. The cover medium used includes image, audio, video and text. The available text Steganography techniques include format-based method, random and statistical character generation and linguistic method. In this project we present a Genetic Algorithm approach text stenography aimed at increasing robustness and capacity of hidden data. The cover text used is a set of random numbers. First, the secret text/payload is encrypted and then converted into its ASCII form. Genetic algorithm is then applied on the cover text obtained from a set of randomly generated numbers to embed the secret message (ASCII form) into the text data (random numbers). The cover text generated is dependent on the length of the secret message. Once optimal results have been reached the embedding process begins to produce a stego text. An extraction algorithm is applied to get the original secret message. The results show that the proposed approach satisfies security, robustness and hiding capacity requirements.

Key words: Steganography, Genetic algorithm, Cover Medium, Encryption

Acknowledgement

This project would not have been possible without the support of many people. It is with immense gratitude that I acknowledge the support and help of my Supervisor, Prof. Peter Wagacha, for his assistance and feedback during the past few months. I am indebted to Prof. Wagacha as he initially proposed this project when I had little idea on what I wanted to work on beyond '*Genetic algorithm to image steganography*'.

I am also grateful to Mr. Christopher Moturi, Mr. Lawrence Muchemi, Mr. Daniel Orwa and Mr. Joseph Ogutu, for their guidance and positive criticism during project presentation, which helped me to improve.

I am grateful to my family and friends who endured this long process with me always offering support and love.

Last but not least, I thank the Almighty God for giving me the grace to finalize this project.

Table of Content

DECLARATION	1
Dedication	2
Abstract	3
Acknowledgement	4
Table of Content	5
List of Figures	7
Definitions.....	8
CHAPTER ONE: INTRODUCTION.....	9
1.1 Background	9
1.2 Problem Statement	10
1.3 Why Text Steganography?.....	10
1.4 Why GA approach to Text Steganography?	10
1.5 Main Objective.....	10
1.6 Specific Objectives	11
1.7 Research Questions.....	11
1.8 Scope.....	11
1.9 Assumption	11
CHAPTER TWO: LITERATURE REVIEW	12
2.1 Introduction.....	12
2.2 Text Steganography Techniques.....	13
2.3 Steganography Algorithms/tools	14
2.3.1 Texto.....	14
2.3.2 Steganosaurus (Stego)	15
2.3.3 SNOW.....	17
2.3.4 Stegparty	19
2.4 Other Previous Works:.....	20
2.5 Limitations of existing Text steganography tools.....	21
CHAPTER THREE: METHODOLOGY	23
3.1 Introduction.....	23
3.2 System Analysis.....	23
3.1.1 Conceptual framework	23
3.1.1.1 Steganography	24
3.1.1.2 Genetic Algorithm	26
3.1.1.3 Steganalysis	27
3.1.2 Functional Requirements.....	27
3.1.3 Scenarios	28
3.1.4 Use Case Diagrams	30
3.2 Design	31
3.2.1 Interactive Diagram.....	31
3.2.2 Functional Design/ Logical Design.....	33
3.3 Implementation Tools	36
CHAPTER FOUR: RESULTS AND EVALUATION.....	37

4.1	Implementation	37
4.2	Results.....	37
4.3	Analysis of System Results.....	38
CHAPTER FIVE: DISCUSSION AND CONCLUSION		41
5.1	Achievement of Objective	41
5.2	Achievement of Research Questions	41
5.3	Discussion of Results in relation to objectives	41
A)	Robustness	41
	Visual attack.....	42
	Statistical attack	42
	Structural attack	42
B)	Capacity of Hidden Message.....	43
5.4	Comparisons of some Text Steganography Tools	44
5.5	Challenges.....	44
5.6	Conclusion	45
Appendix 1: User Manual.....		48
A)	Steganography.....	48
B)	Steganalysis.....	50
Appendix II: Source Code.....		52

List of Figures

Figure 1: Process flow of Texto Steganography Tool.....	16
Figure 2: Process flow of Stego Steganography Tool.....	17
Figure 3: Process flow of SNOW Steganography Tool.....	19
Figure 4: Process flow of Stegparty Steganography Tool.....	20
Figure 5: Conceptual Framework showing Application of Genetic Algorithm in Text Steganography.....	25
Figure 6: Conceptual Framework of Genetic Algorithm approach.....	27
Figure 7: Conceptual Framework for Steganalysis.....	28
Figure 8: Shows the results obtained from the implementation process.....	38
Figure 9: Bar Graph representing file variance by chromosome length.....	41
Figure 10: GATS interface for encryption.....	49
Figure 11: How to open a saved .txt file.....	49
Figure 12: Conversion of secret message to its ASCII representative.....	50
Figure 13: Generation of cover text.....	50
Figure 14: Generation of stego text.....	51
Figure 15: Opening steganalysis interface.....	51
Figure 16: Opening the saved stego text.....	52
Figure 17: Extraction of the secret message.....	52

Definitions

ASCII – American Standard Code for Information Interchange

Cover Text - Text containing an embedded message.

Cipher text – Refers to encrypted data.

Payload – Secret Message

Cryptography – The art of protecting information by encrypting it into an unreadable format, called cipher text. A secret key is used to decrypt the message into plain text.

Encryption – The translation of data into a secret code.

Decryption – The inverse of encryption

GA - Genetic Algorithm

Plain text – Refers to any message that is not encrypted - also called clear text.

Steganalysis – The art of discovering and rendering useless covert messages.

Steganography - A means of overlaying one set of information ("message") on another (a cover).

Stego/Steno text - The result of combining the cover text and the embedded message.

CFB - cipher-feedback

ICE - Information Concealment Engine

SNOW - Steganographic Nature Of Whitespace

CHAPTER ONE: INTRODUCTION

1.1 Background

Steganography is a technique that establishes a covered information channel in point-to-point connections by embedding or hiding data inside a cover medium in such a way that the existence of any communication itself is undetectable.

Steganographic technologies are a very important part of the future of Internet security and privacy on open systems such as the Internet because important data can be hidden inside a medium so that only the parties intended to get the message knows that a secret message exists. The mostly used medium include text, video, audio or images.

Text Steganography: Is a method of using written natural language to conceal a secret message [Chapman *et al.*, 2001]. Hiding information in plain text can be done in many different ways. One of the technique this type of steganography include the modification of the layout of a text, rules like using every n -th character or the altering of the amount of white space after lines or between words (Huang, Yan). Another possible way of storing a secret inside a text is using a publicly available cover source, a book or a newspaper, and using a code which consists for example of a combination of a page number, a line number and a character number. This way, no information stored inside the cover source will lead to the hidden message. Discovering it relies solely on gaining knowledge of the secret key.

Image Steganography: To hide information, straight message insertion may encode every bit of information in the image or selectively embed the message in “noisy” areas that draw less attention—those areas where there is a great deal of natural color variation. The message may also be scattered randomly throughout the image.

Audio Steganography: In a computer-based audio steganography system, secret messages are embedded in digital sound. The secret message is embedded by slightly altering the binary sequence of a sound file.

Video Steganography: Video files are generally a collection of images and sounds, so most of the presented techniques on images and audio can be applied to video files. The

great advantages of video are the large amount of data that can be hidden inside and the fact that it is a moving stream of images and sounds. Therefore, any small but otherwise noticeable distortions might go by unobserved by humans because of the continuous flow of information.

Some of the areas in which steganography can be applied include:

- 1) Confidential communication and secret data storing
- 2) Protection of data alteration
- 3) Access control system for digital content distribution
- 4) Media Database systems

1.2 Problem Statement

Steganography needs to satisfy perceptual transparency, capacity of hidden data and robustness. The focus of this project was to exploit the use of steganography in textual communication and information since the available text steganography techniques are prone to visual, structural and statistical attacks.

1.3 Why Text Steganography?

The advantage of text steganography over other media like video, audio and image is its smaller memory occupation and simpler communication.

1.4 Why GA approach to Text Steganography?

With the world wide need for secure data transmission over the internet and the failures on most of the currently used algorithms, there was a need to look at the capabilities of Genetic Algorithm as applied to text steganography. This is because genetic algorithm approach tends to focus on longer hidden message, higher text quality and increase on the robustness.

1.5 Main Objective

The aim of this project was to develop a tool by use of genetic algorithm technique on the cover text in order to investigate how to increase the overall rate of hidden data without affecting the quality of the cover text and with reduced probability of message detection.

1.6 Specific Objectives

- a) Investigate and evaluate existing methods of text based Steganography
- b) Introduce genetic algorithm approach to text based Steganography, so as to produce a secure and robust Steganography tool.
- c) Develop a prototype that will represent the use of genetic algorithm in text steganography.
- d) Analyse the developed tool and algorithm used experimentally to find out if this proposed method works properly and is considered to give almost the optimum solution.

1.7 Research Questions

- a) How are online users experiencing or addressing security and privacy issues in message/information transfer?
- b) What are the available algorithms to steganography and how can the use of Genetic Algorithm be used to produce a secure and robust steganography tool?
- c) How will the implementation of genetic algorithm based approach to text steganography reduce the likelihood of visual, structural and statistical attack to embedded messages?

1.8 Scope

This project focus was on research and development of a tool with the capability of hiding a message inside a cover text and transmitting it, without it being detected. Most text steganography tools available lack the robustness and capability of hiding a lot of text hence the research on genetic algorithm to text steganography.

1.9 Assumption

The assumption is that GA's can solve complex environments with non-linear behavior.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

With the widespread use of Internet and wireless networks, and the blooming growth in consumer electronic devices and advances in multimedia compression techniques, multimedia streams are easily acquired nowadays. In an attempt to ensure protection of the a-fore-mentioned multimedia contents and effective hiding of additional data into such digital content, several techniques emerged. Nevertheless, none of the existing schemes can fully shield against all detection attacks. Texts with hidden data are expected to have higher entropy than those without.

There are basically three types of protocols used in Steganography [Katzenbeisser, Fabien and Petitcolas 2000]; Pure Steganography; Secret Key Steganography; and Public Key Steganography.

Pure Steganography is defined as a steganographic system that does not require the exchange of a cipher such as a stego-key. This method of Steganography is the least secure means by which to communicate secretly because the sender and receiver can rely only upon the presumption that no other parties are aware of this secret message. Using open systems such as the Internet, we know this is not the case at all.

Secret Key Steganography is defined as a steganographic system that requires the exchange of a secret key (stego-key) prior to communication. Secret Key Steganography takes a cover message and embeds the secret message inside of it by using a secret key (stego-key). Only the parties who know the secret key can reverse the process and read the secret message. Unlike Pure Steganography where a perceived invisible communication channel is present, Secret Key Steganography exchanges a stego-key, which makes it more susceptible to interception. The benefit to Secret Key Steganography is even if it is intercepted; only parties who know the secret key can extract the secret message.

Public Key Steganography is defined as a steganographic system that uses a public key and a private key to secure the communication between the parties wanting to

communicate secretly. The sender will use the public key during the encoding process and only the private key, which has a direct mathematical relationship with the public key, can decipher the secret message. Public Key Steganography provides a more robust way of implementing a steganographic system because it can utilize a much more robust and researched technology in Public Key Cryptography. It also has multiple levels of security in that unwanted parties must first suspect the use of steganography and then they would have to find a way to crack the algorithm used by the public key system before they could intercept the secret message.

2.2 Text Steganography Techniques

There are several text steganography techniques that have so far been devised, this include:

a) **Format-based methods:** this technique makes use of physical text formatting of text as a place in which to hide information. Generally, this method modifies existing text in order to hide the steganographic text. Insertion of spaces, deliberate misspellings distributed throughout the text, resizing the fonts are some of the many format-based methods being used in text steganography.

b) **Random and statistical generation** is generating cover text according to the statistical properties. This method is based on character sequences and words sequences. The hiding of information within character sequences is embedding the information to be appeared in random sequence of characters. This sequence must appear to be random to anyone who intercepts the message.

Statistical character generation involves taking the statistical properties of word-length and letter frequency in order to create “words” which will appear to have the same statistical properties as actual words in a given language. The hiding of information within word sequences, the actual dictionary items can be used to encode one or more bits of information per word using a codebook of mappings between lexical items and bit sequences, or words themselves can encode the hidden information.

c) Linguistic method considers the linguistic properties of generated and modified text, frequently uses linguistic structure as a place for hidden messages. Syntactic method is a linguistic steganography method where some punctuation signs like comma (,) and full-stop (.) are placed in proper places in the document to embed a data. This method needs proper identification of places where the signs can be inserted. Another linguistic steganography method is semantic method. In this method the synonym of words for some pre-selected are used. The words are replaced by their synonyms to hide information in it.

2.3 Steganography Algorithms/tools

Listed below are some of the common Steganography Algorithms in use:

2.3.1 Texto

Texto [Maher K.] is a rudimentary text steganography program to facilitate the exchange of binary data. It uses a simple substitution cipher which transforms *uuencoded* or *pgp ASCII-armoured ASCII* data, especially encrypted data into English sentences so that the text will look apparently reasonable during data transmission. Each symbol is replaced by nouns, verbs, adjectives, and adverbs in the preset sentence structures without punctuation or "connecting" words through English sentences. However, not all of the words in the resulting English are significant to the Texto program. Usually, the output of Texto is close enough to normal English text that it will slip by any kind of automated scanning.

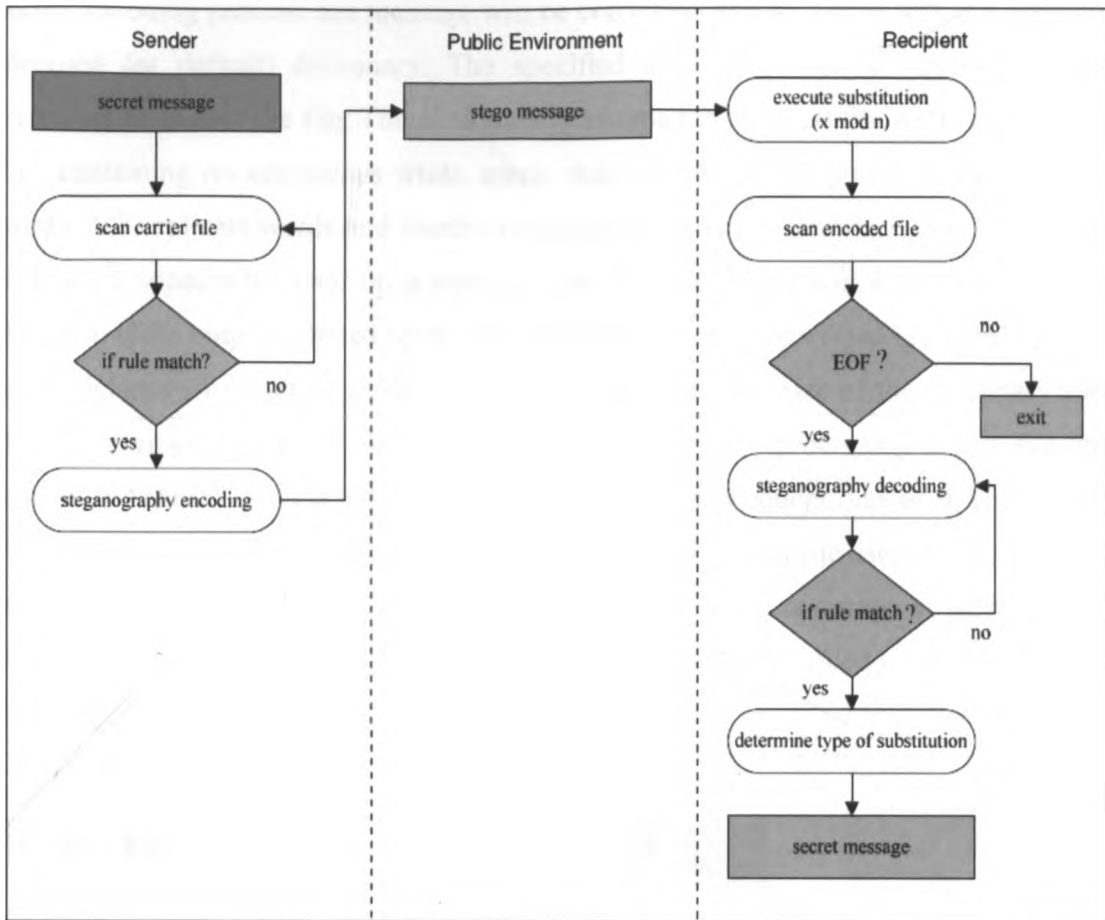


Figure 1: Process flow of Text Steganography Tool.

2.3.2 Steganosaurus (Stego)

Walker J. (1997) introduced Steganosaurus also known as Stego which uses a line-shift coding method (plain text steganography utility) which encodes a binary file into a gibberish text based on either a spelling dictionary or words taken from a text document. The output of Stego converts any binary file into nonsense text based on a dictionary from a source document. The output of stego is nonsense but statistically resembles text in the language of the dictionary supplied. A human reader will instantly recognize it as gibberish while to eavesdroppers; the encrypted messages may consider it to be unremarkable, especially if a relatively small amount of such text appears within a large document. Stego makes no attempt, on its own, to prevent the message from being read. It is the equivalents of a code book with unique words as large as the dictionary. Text

created by stego uses only characters in the source dictionary or document. It means that during encoding process, the message will be converted into an output text file using the specified (or default) dictionary. The specified file called 'dictfile' is used as the dictionary to encode the file. The dictionary is assumed to be a text file with one word per line, containing no extraneous white space, duplicate words, or punctuation within the words. All duplicate words and words containing punctuation characters are deleted, and each word appears by itself on a separate line. The Stego text will look less obviously gibberish if the output is based upon template sentence structures filled in by dictionaries. The efficiency of encoding a file as words depends upon the size of the dictionary used and the average length of the words in the dictionary. Preprocessing a text file into dictionary format allows it to be loaded much faster in subsequent runs of stego. Another file namely 'textdict', is created to build the dictionary for input file used during encoding or decoding process. The 'textdict' is scanned and words, consisting of alphanumeric characters, are extracted. Duplicate words are automatically discarded to prevent errors in encoding and decoding processes.

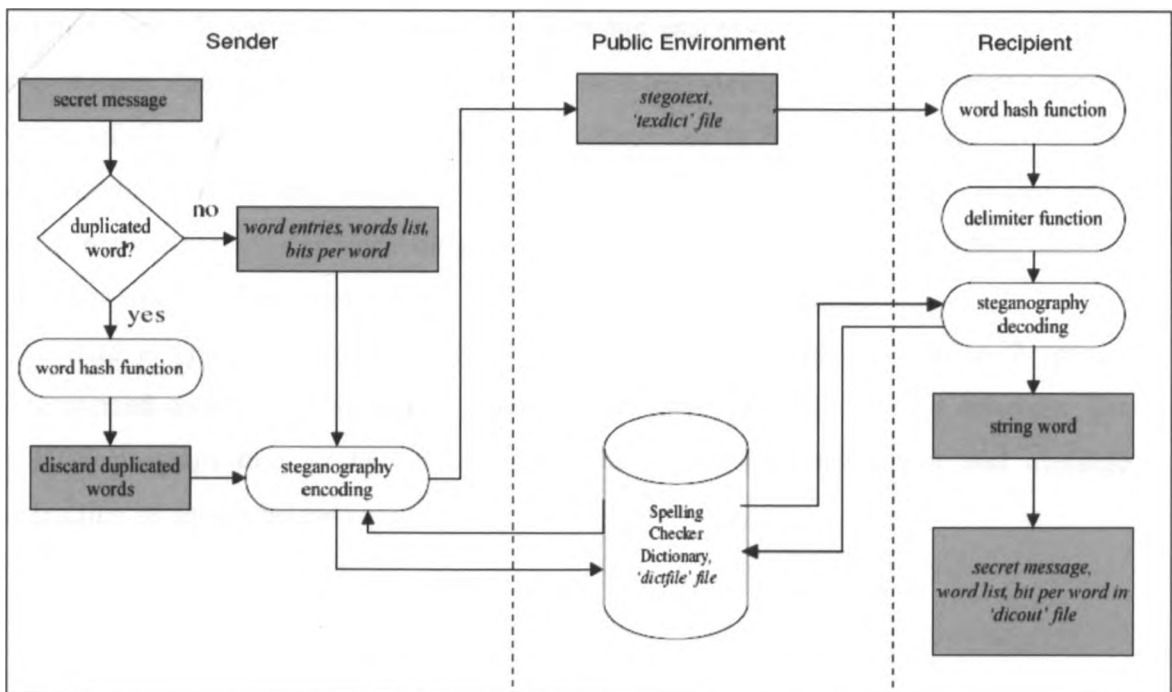


Figure 2: Process flow of Stego Steganography Tool

Stego can then be applied to the encrypted output, transforming it into seemingly innocuous text for transmission, so it can be sent by sender through media, such as electronic mail, which cannot transmit binary information. If the medium used to transmit the output of stego, *'textdict'* cannot correctly deliver such data; the recipient will be unable to reconstruct the original message. To avoid this problem, the sender can either encode the data before transmission or use a dictionary which contains only characters which can be transmitted without loss. The decoding process by receiver to recover the original message, *'dictout'*, must be carried out using the same dictionary as encoding process because the ability to recognize gibberish in text is highly language dependent. Usually, the default dictionary is the system spelling checker dictionary. However, this dictionary is not standard across all systems.

2.3.3 SNOW

Steganographic Nature Of Whitespace or SNOW [Kwan 1998]; is a program for concealing messages and extracting messages in ASCII text file. This feature coding method conceals messages by appending tabs and spaces (known as whitespace) at the end of lines. Tabs and spaces are invisible to most text viewers, hence the steganographic nature of this encoding scheme. This allows messages to be hidden in the ASCII text without affecting the text visual presentation. Since trailing spaces and tabs occasionally occur naturally, their existence should not be deemed sufficient to immediately alert an observer who stumbles across them.

The data is concealed in the text file by appending sequences of up to 7 spaces, interspersed with tabs. This usually allows 3 bits to be stored in every 8 columns. The SNOW program runs in two modes which are message concealment and message extraction as shown below.

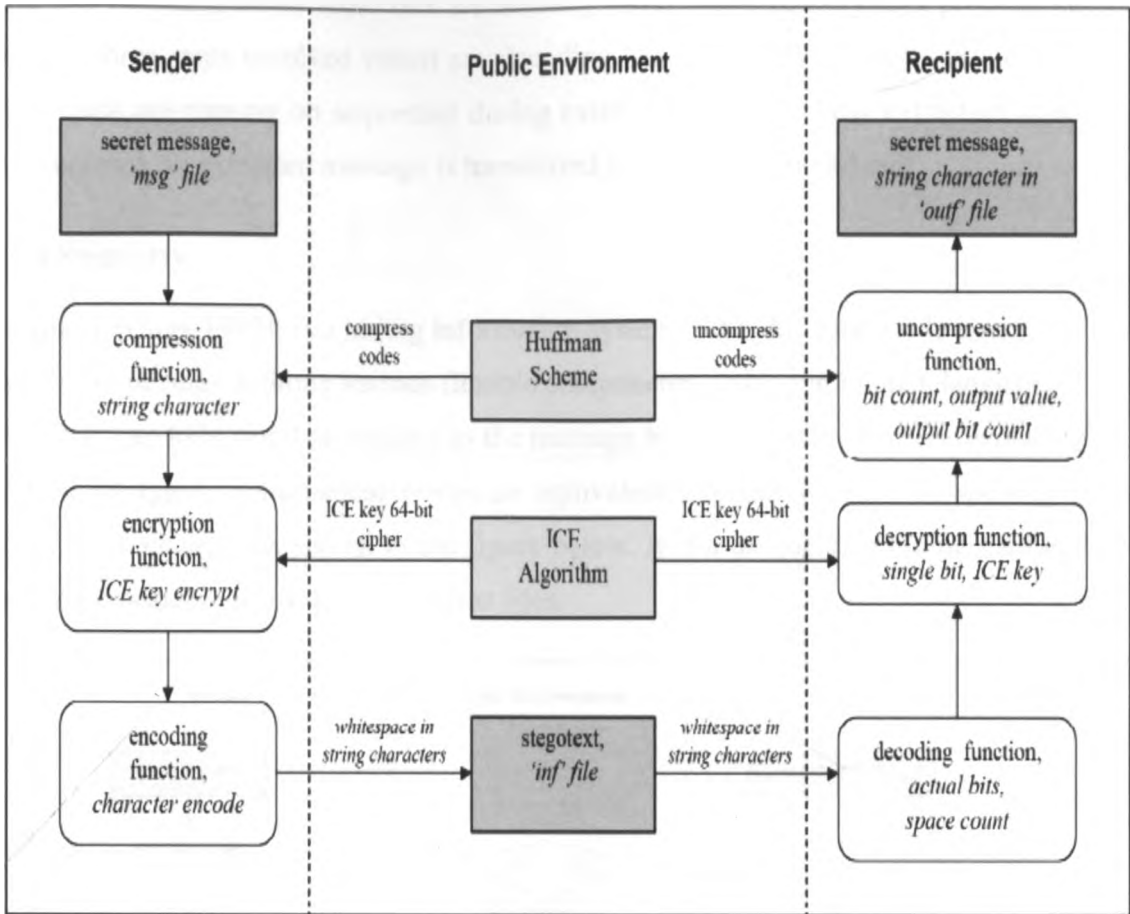


Figure 3: Process flow of SNOW Steganography Tool

There are three important steps involve in the concealing process which are;

- i. *Compression* - used a rudimentary Huffman encoding scheme where the tables are optimized for English text. This was chosen because the *whitespace* encoding scheme provides very limited storage space in some situations, and a compression algorithm with low overhead was needed.
- ii. *Encryption* - used an ICE encryption algorithm with 64-bit block cipher. It runs on a 1-bit *cipher-feedback* (CFB) mode, which is quite inefficient (requiring a full 64-bit encryption for each bit of output).
- iii. *Encoding scheme* – at the beginning of a message, a tab is added immediately after the text on the first line where it will fit. Tabs are used to separate the blocks of spaces. A tab is not appended to the end of a line unless the last 3 bits coded to zero spaces, in which

case it is needed to show some bits are actually there. While in extracting process, there are also three steps involved which are decoding, decryption and decompression. All of these steps are running on sequential during extraction process. After extraction process is completed, an extracted message is transferred to output text called *outf*.

2.3.4 Stegparty

Stegparty [Hugg 1999]; is a hiding information system that hides data inside a text file by using a set of rules defining various flexible components within the English language.

Stegparty can hide small alterations to the message by matching the text and replacing it with small typos, grammatical errors, or equivalent expressions such as spelling and punctuation changes as shown in the figure below. It is a unique data hiding method by creating misspellings inside original text files.

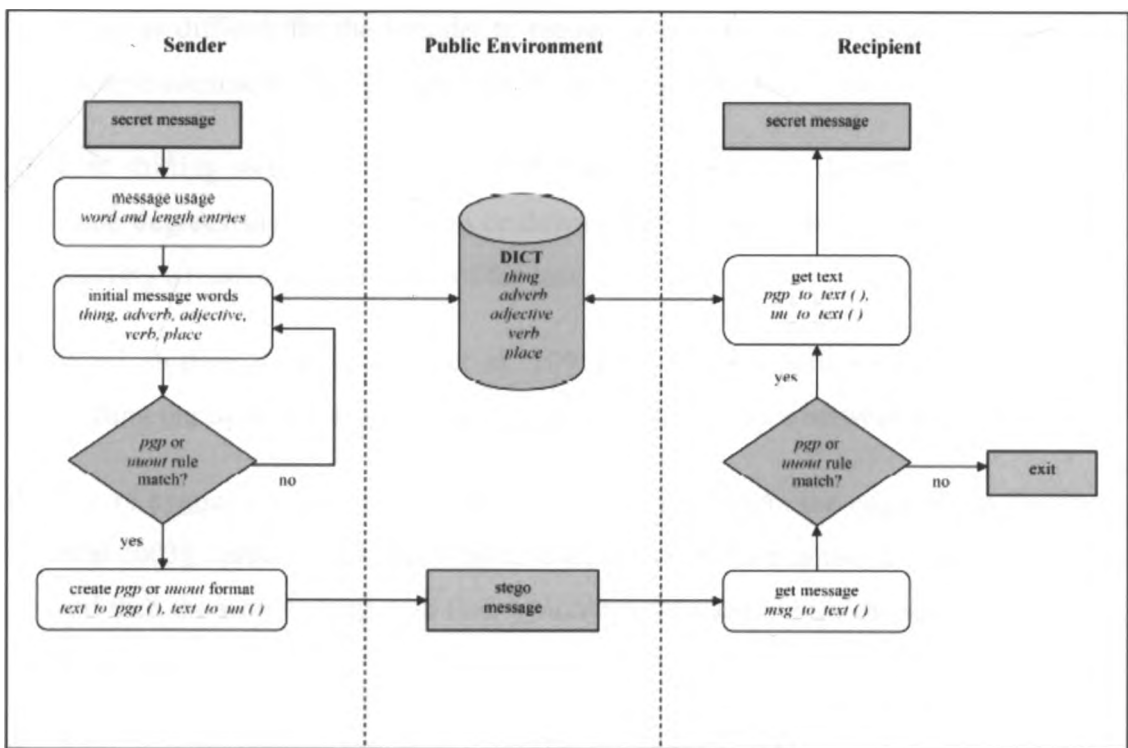


Figure 4: Process flow of Stegparty Steganography Tool

2.4 Other Previous Works:

Other previous works on text Steganography include;

2.4.1 Use of specific characters from words [Moerland 2003]; in this method, some specific characters from certain words are selected and are used to hide the secret information. The first character of the first word of each paragraph can be used to hide a secret message one character at a time such that by placing these characters side by side, we get the whole message.

Moerland also discussed about using punctuation marks. The idea behind this approach is to utilize the presence of punctuation marks like comma (,), semi colon (:), quotes (,, “) etc. in the text for encoding a secret message. The use of punctuation marks is quite common in the normal english text and hence it becomes difficult for the intruder to recognize the presence of secret message in the text document. This accounts for the security of the technique.

2.4.2 Line shifting method [Low et al. 1995]; where the lines of the text are shifted to some degrees say 1/300 inch up or down and then the information is hidden by creating a hidden unique shape of the text.

2.4.3 Word shifting method [Low et al. 1995]; here, the information is hidden by shifting the words horizontally or by changing the distance between the words.

2.4.4 Use of synonyms of certain words to hide the message in the english text [Niimi et al 2003]; certain words from the text are selected, their synonyms are identified and then the words along with their synonyms are used to hide the secret message in the text.

2.4.5 Adding extra white-spaces in the text [Huang and Yan 2001]; white spaces can be placed at the end of each line, at the end of each paragraph or between the words.

2.4.6 Persian/Arabic text [M. Shirali-Shahreza 2006]; and Urdu/Arabic text [Memon et al. 2008]; one of the characteristics of these languages is the abundance of points in its letters. One point letters can be used to hide the information by shifting the

position of a point a little bit vertically high with respect to the standard point position in the text.

- 2.4.7 Hindi text Steganography [Alla and Prasad 2009]; this technique is based on the fact that each language has its own characteristics. Every language is formed of combinations of one or more vowels and consonants. These vowels and consonants and the combination of the two, forms the basis of this hindi text steganography technique. This technique makes use of two elements: simple letters (pure vowels and pure consonants) and compound letters (combinations of vowels, consonants, vowels and consonants).
- 2.4.8 Hiding secret message in the English text by using different spellings of the words [Shirali-Shahreza 2008]; most words have different spelling in UK and US. For example "dialog" has different terms in UK (dialogue) and US (dialog). This difference in spellings forms the basis of steganography.
- 2.4.9 Emoticon based text Steganography [Wang et al 2009]; emoticons are emotional icons that are used in online chatting. These emoticons express the feeling or mood of the persons communicating with each other.

2.5 Limitations of existing Text steganography tools

The existing tools for text steganography have been prone to visual, structural, syntactic, semantic and statistical attacks, hence transfer of confidential message remain unsecure. One of the text steganography tool deals with removing hidden messages from a plain text by rewriting and reformulating the contents.

The character position schemes will no longer work because the words have been changed, and the same is valid for the differentiations in white spacing, since the text will have a new layout.

Usage of a publicly available cover source cannot easily be altered. There is no effective way of cracking this method, except for intercepting the secret key.

With several limitations existing within the existing text steganography tools, genetic algorithm was used on text steganography because of its insusceptibility to the above mentioned attacks (i.e. visual, structural, syntactic, semantic and statistical attacks) and its ability to give an almost optimal solution.

CHAPTER THREE: METHODOLOGY

3.1 Introduction

This project used objectory use case approach which was proposed by I. Jacobson (1994). This involved identification of functional requirements from which use case artefacts were developed, after which, the dynamic and static behaviour of the system were analysed and modelled. The modelling of static behaviours was done through identification of objects and classes which were represented using Unified Modeling Language (UML) diagrams. The dynamic aspects of the system were modelled using sequence, interactive, state diagrams and collaboration diagrams. Thereafter, implementation was done, where the algorithm was implemented using genetic algorithm approach. The implementation of genetic algorithm in text steganography is represented in figure 5 below.

3.2 System Analysis

3.1.1 Conceptual framework

Figure 5 below shows the conceptual framework for the genetic algorithm approach to text steganography. It demonstrates the flow of the secret message from encryption, to application of genetic algorithm operators on the cover text to produce a stego text. Random numbers generated from the ASCII representation of the secret message are used to generate the initial population. During reproduction, selection and crossover operations are performed to generate new offsprings. Fitness function is performed at the initial population to get the fit parents to reproduce and the offspring. The children/offsprings are introduced to the population and again fitness function is performed to discard the unfit individuals in the population. The procedure is repeated until the optimum solution is found, then the embedding process begins to produce the stego text. Steganalysis is the reverse to get the secret message from the stego text.

3.1.1.1 Steganography

This figure 5 below demonstrates the conceptual framework for steganography process.

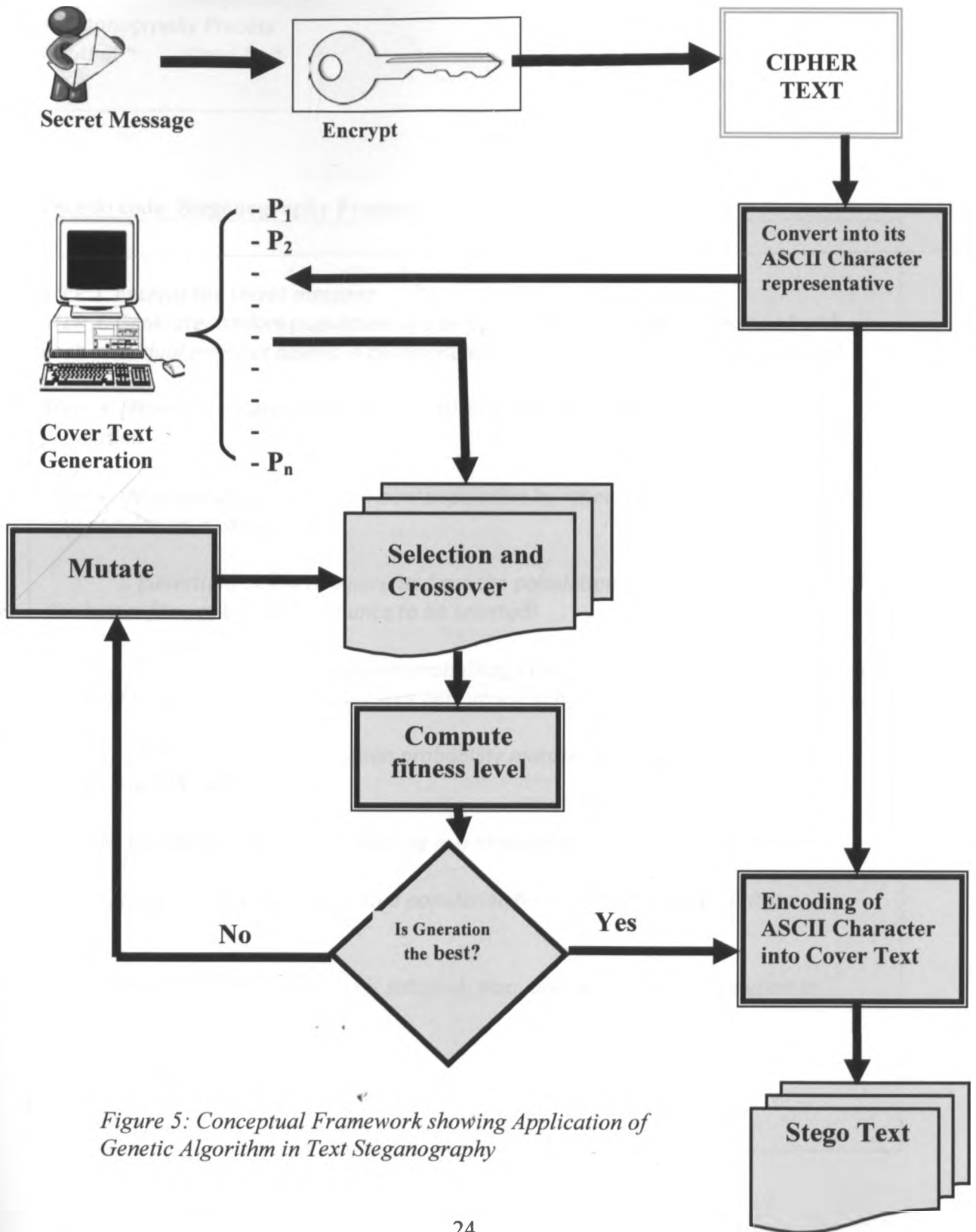


Figure 5: Conceptual Framework showing Application of Genetic Algorithm in Text Steganography

Input ~ Secret Message
Encryption key
>Steganography Process
>Output ~ Stego Text

Pseudo code: Steganography Process

>

STEP 1. Encrypt the secret message

STEP 2: Generate random population of size L (L-length of the Secret Message) with each individual member having n chromosomes (suitable solutions for the problem)

STEP 3. [Fitness] Evaluate the fitness $f(x)$ of each chromosome individual in the population

STEP 4. [New population] Create a new population by repeating following steps until the new population is complete

- i. [Selection] Select two parents from the population with the best fitness level (the better fitness, the bigger chance to be selected)*
- ii. [Crossover] With a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.*
- iii. [Mutation] With a mutation probability mutate new offspring at each locus (position in chromosome).*
- iv. [Accepting] Place new offspring in a new population*

STEP 5. [Replace] Use new generated population for a further run of algorithm

STEP 6. [Test] If the end condition is satisfied, stop, and return the best solution in current population

STEP 7. [Loop] Go to step 4

3.1.1.2 Genetic Algorithm

The figure 6 shows the conceptual framework for the genetic algorithm used on the cover text. It shows how the population is generated and fitness function applied to the individuals, the point at which crossover and mutation operators are introduced until the optimal solution is found.

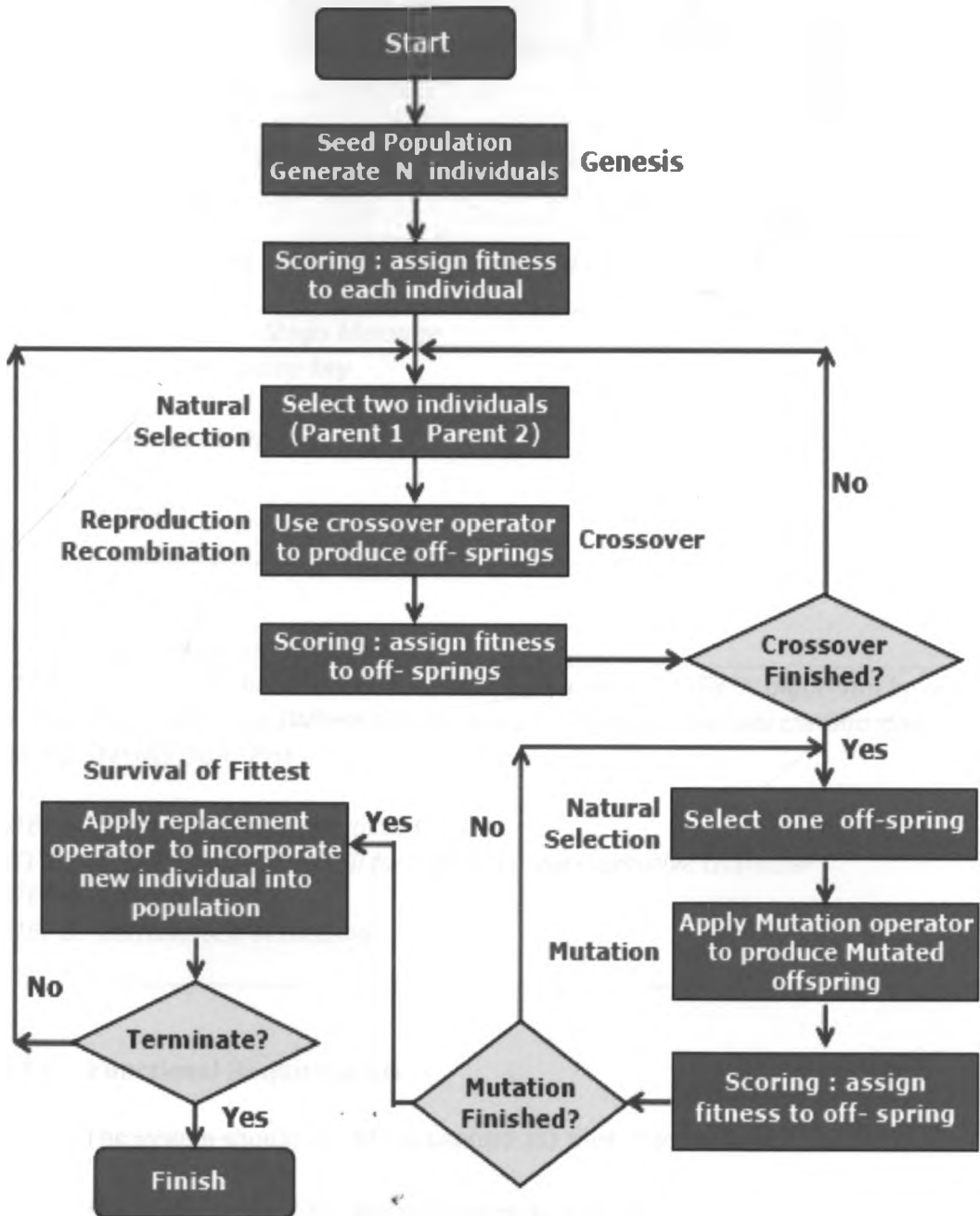


Figure 6 : Conceptual Framework of Genetic Algorithm approach

3.1.1.3 Steganalysis

Figure 7 demonstrates the conceptual framework for steganalysis process.

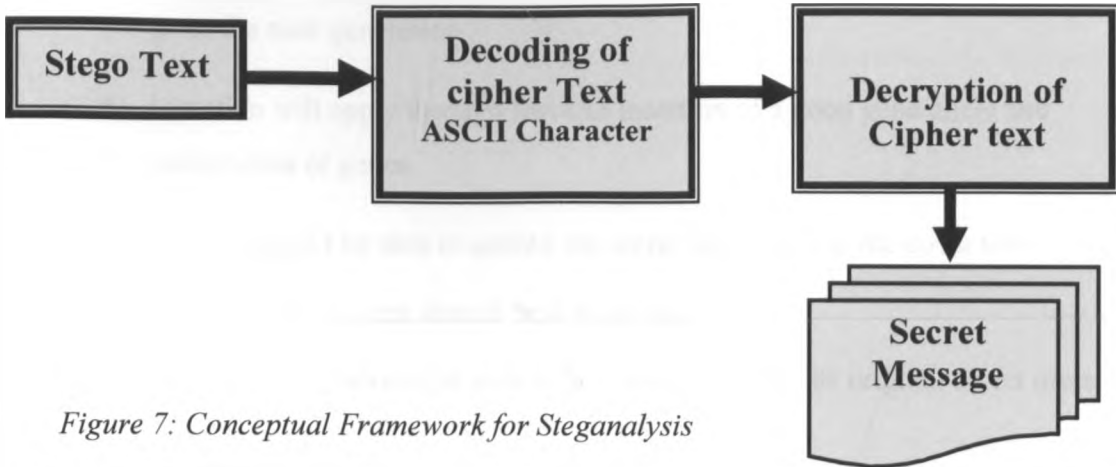


Figure 7: Conceptual Framework for Steganalysis

- Input ~ Stego Message
Decryption key
- Steganalysis Process
- Output ~ Secret Message

Pseudo code: Steganalysis Process

STEP 1: Retrieve the hidden text using the First in First Out (FIFO) algorithm by selecting n value from stego text (Where n is the number of of an individual chromosomes used during STEGANOGRAPHY)

STEP 2: Extract the ASCII characters

STEP 3: Convert from the ASCII format to its representative character

STEP 4: Decrypt

STEP 5: Retrieve Secret Message

3.1.2 Functional Requirements

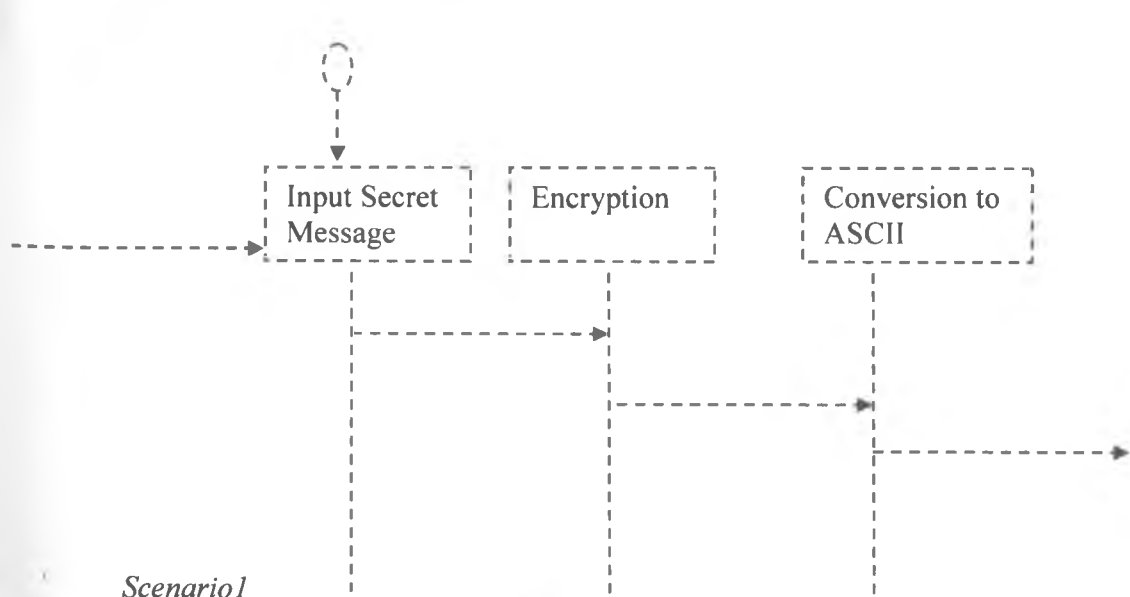
- 1) The system should be able to capture .txt files (Secret file)
- 2) The system should be able to encrypt/decrypt files
- 3) Convert the secret file contents, from characters to ASCII

- 4) The system should be able to generate random numbers which will be use as population for the GA algorithm
- 5) It should be able to perform a fitness function to get the best individuals that will go to the next generation
- 6) Mutation will apply through forceful insertion of a good gene/allele and substitution of genes
- 7) System should be able to embed the secret message into the cover text
- 8) Output of the system should be a stego text.
- 9) The stego text should be able to be reverted back to its original secret message

3.1.3 Scenarios

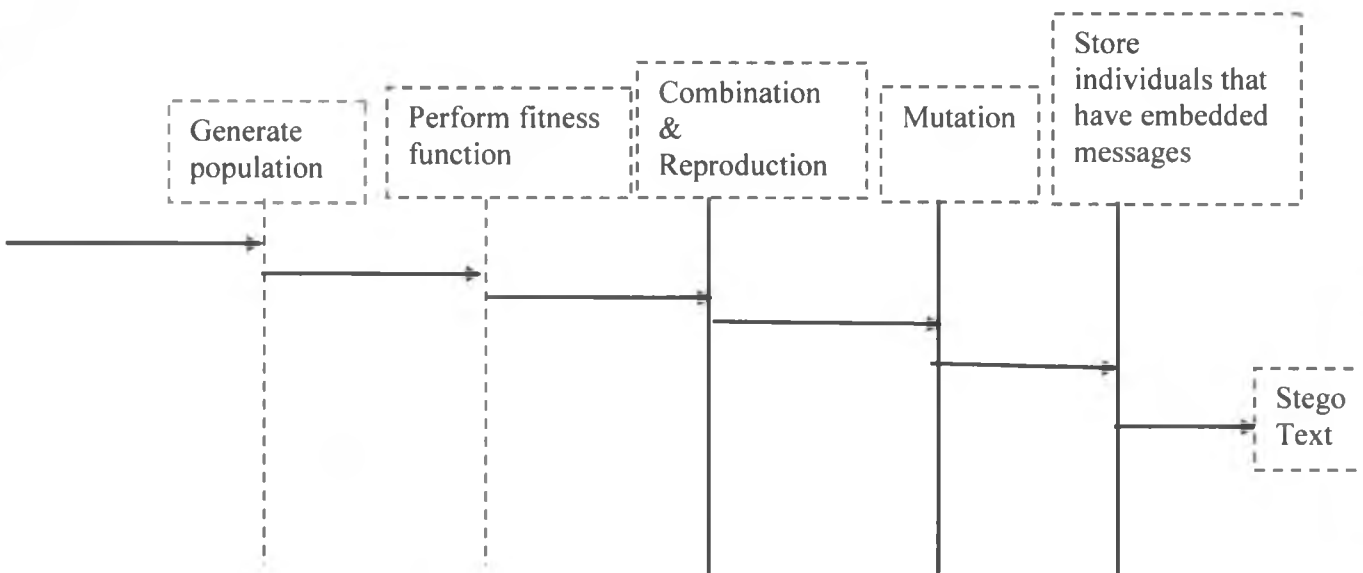
S.1: Input the Secret Message

- Encryption
- Conversion to ASCII



S 2: Application of Genetic Algorithm

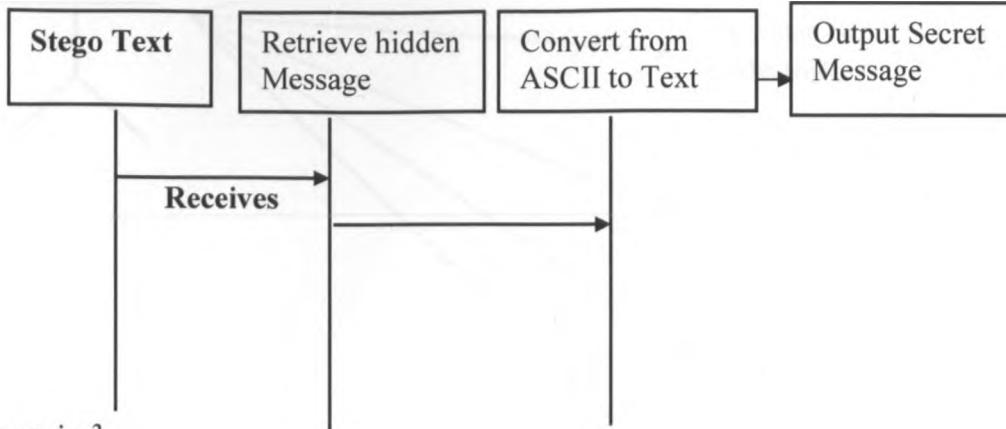
- The system will generate a first set of random numbers this will act as the initial population
- It will be in blocks of 6 i.e. one chromosome has 6 genes/allele (in this case)
- This is dependent on the length of the secret text. The longer the secret text/message the larger the generation of the population
- Fitness function will be performed on individual chromosome
- The two most fit will combine and produce children
- Embedding of the first character of the secret message will be done
- Substitution with the last character in the individual chromosome will be done
- Stacking of the embedded chromosomes whereby the first in will be first out during retrieval process



Scenario 2

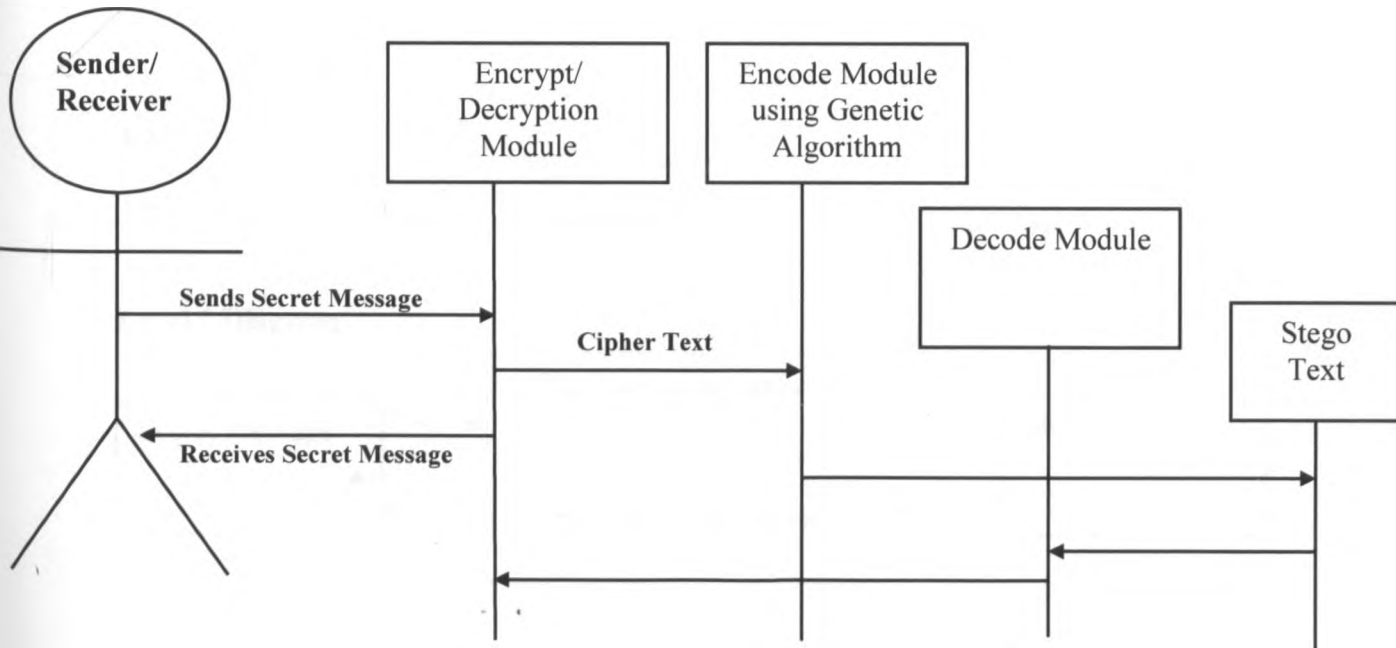
S 3: Steganalysis

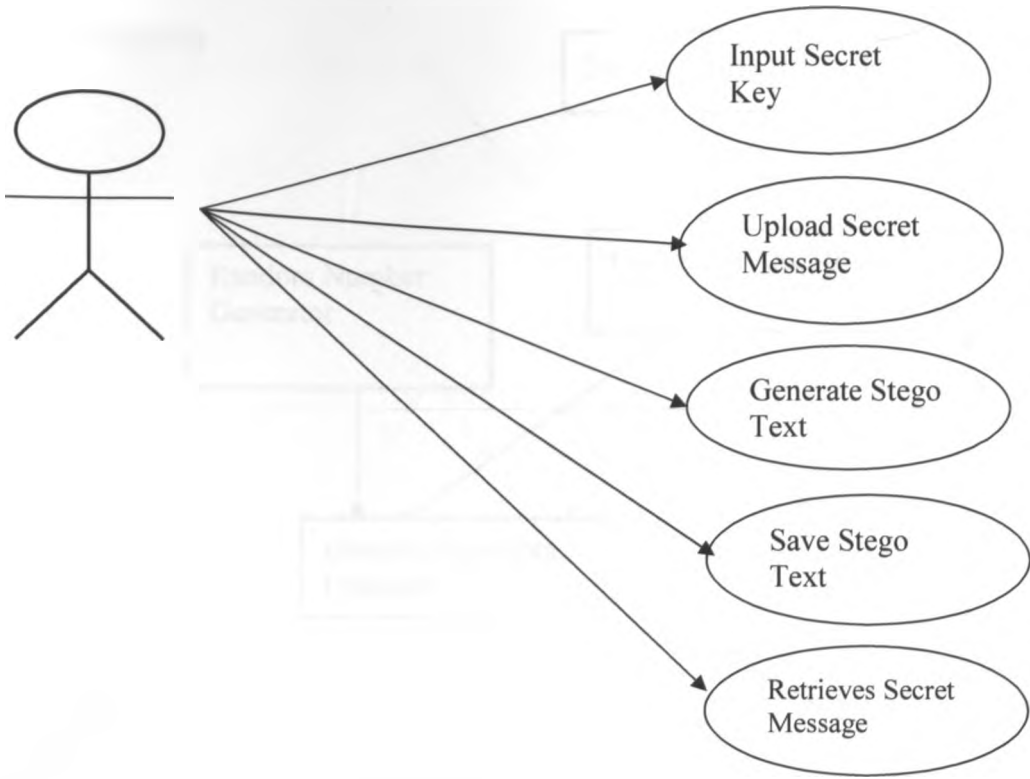
- Gets the stego text
- Retrieves the hidden message
- Converts it from ASCII into text



Scenario 3

3.1.4 Use Case Diagrams

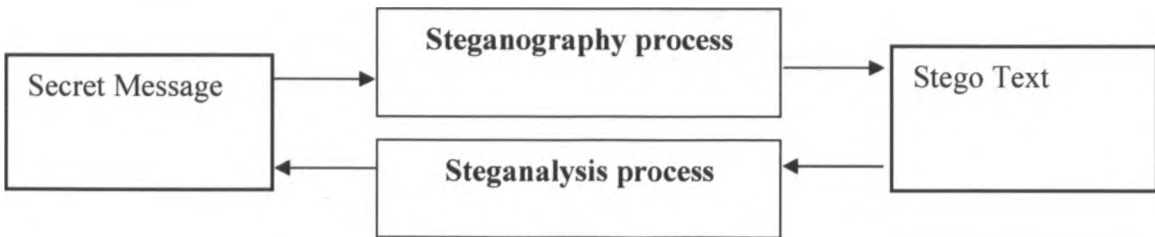




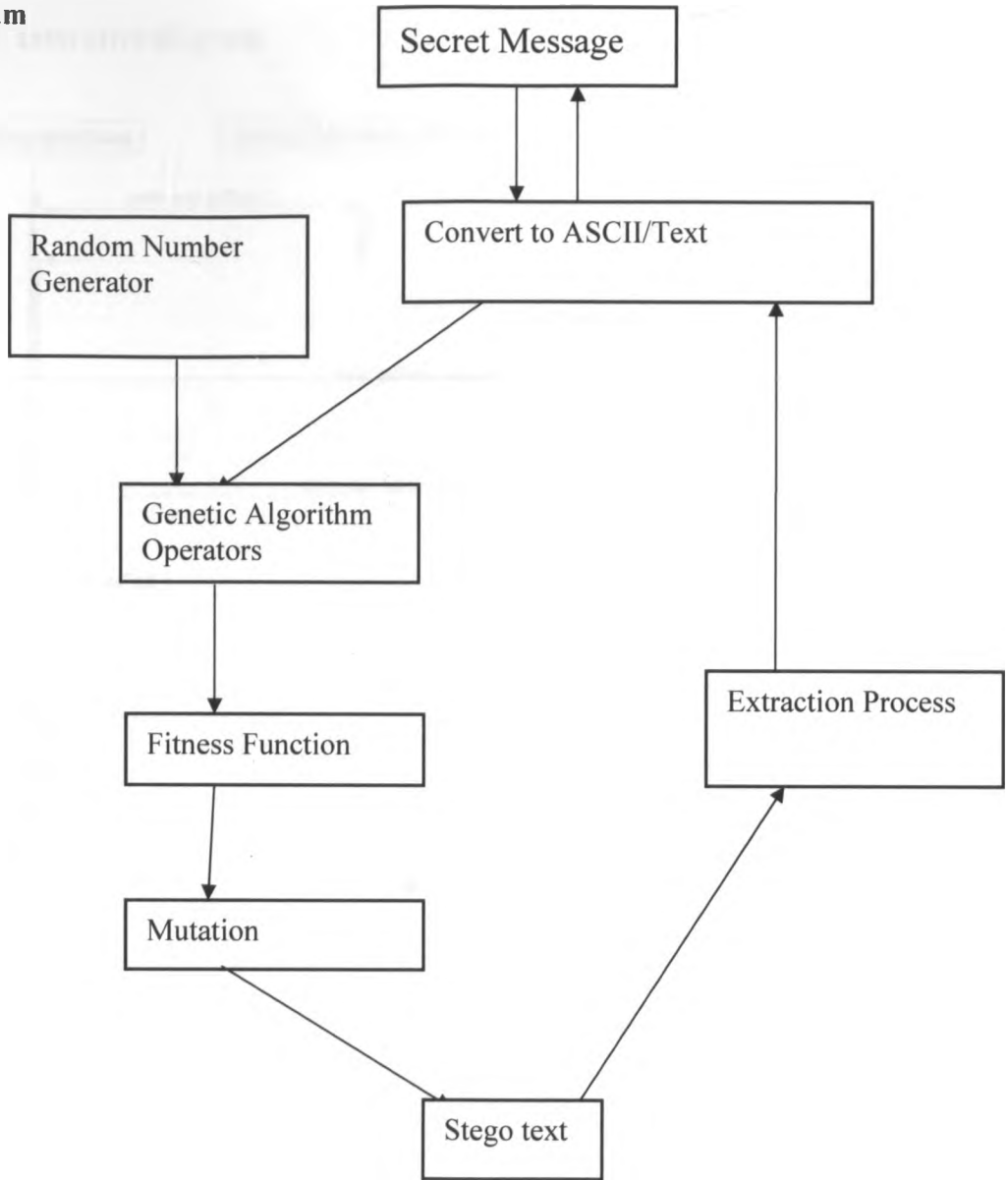
3.2 Design

3.2.1 Interactive Diagram

Level 1 Diagram

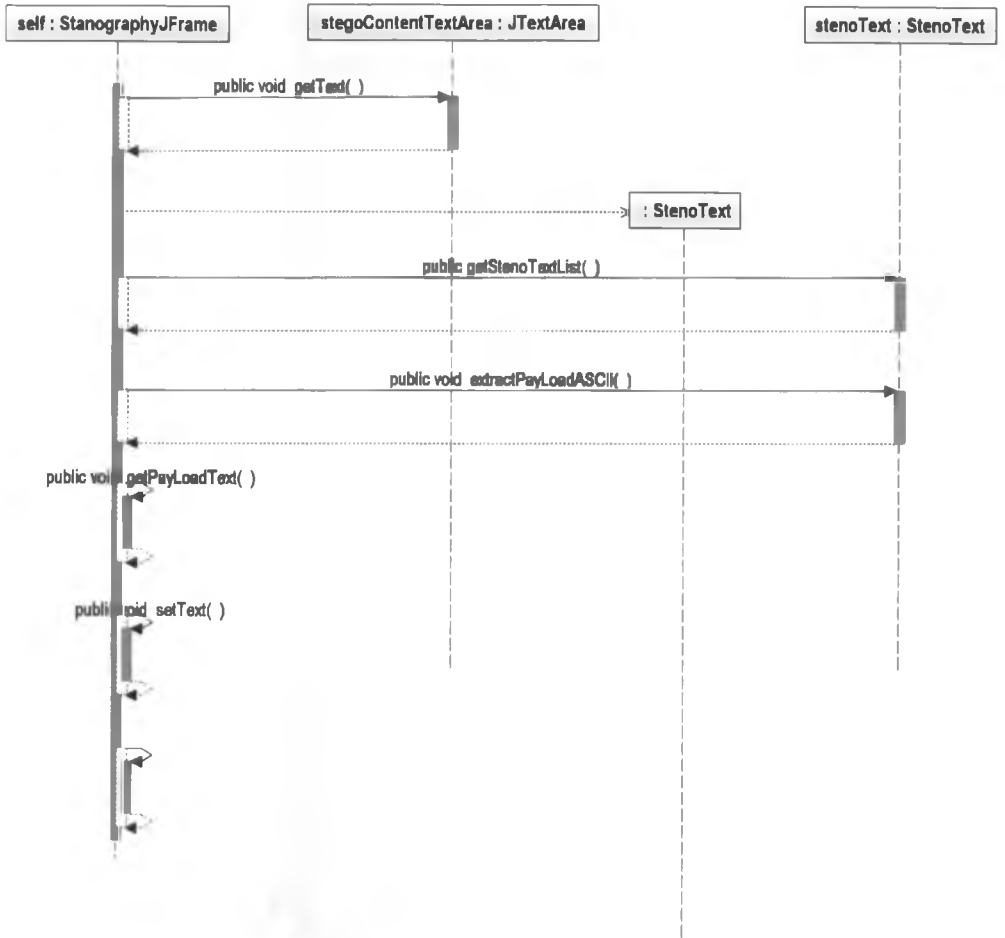


Level 2 Diagram

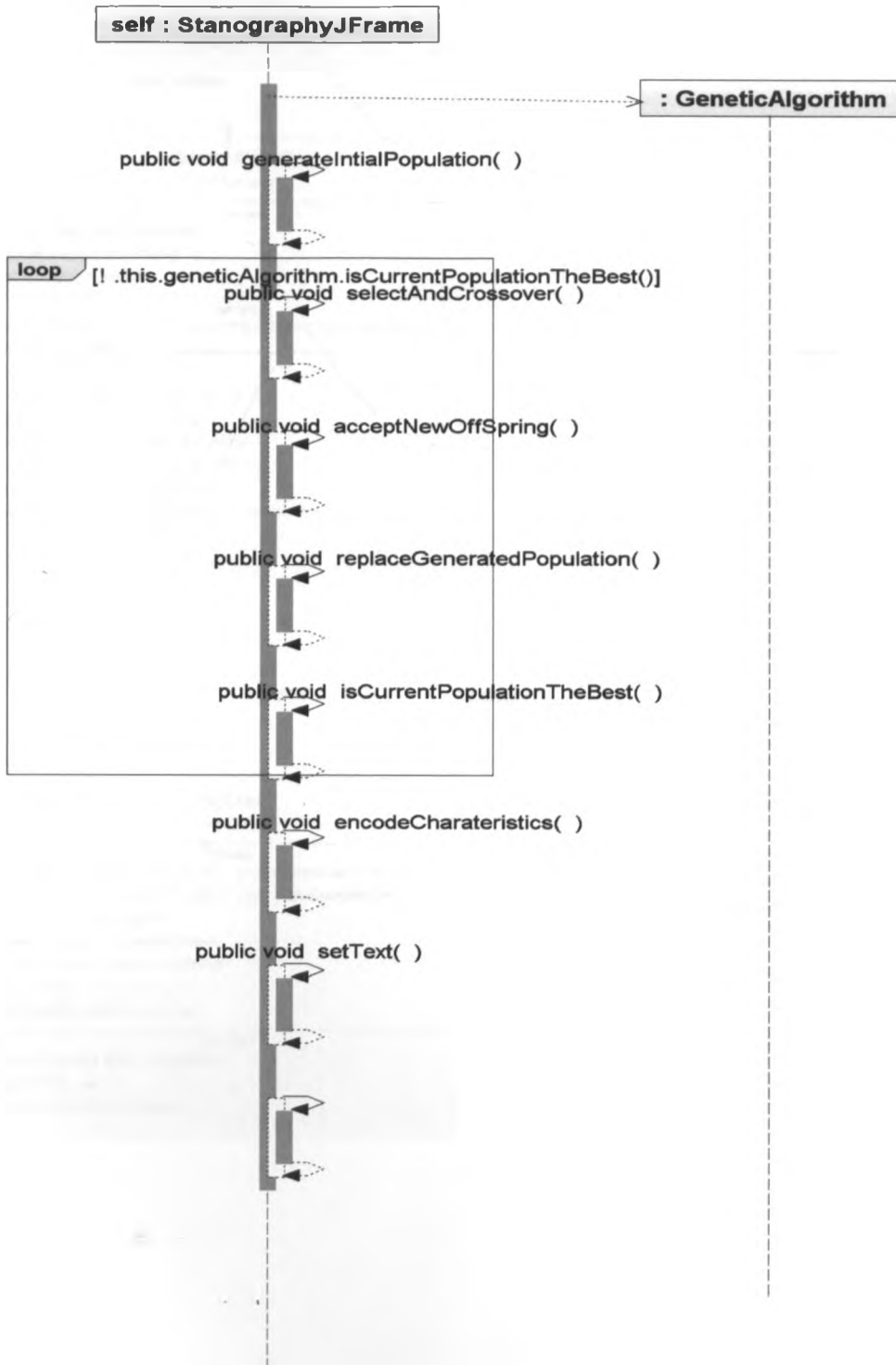


3.2.2 Functional Design/ Logical Design

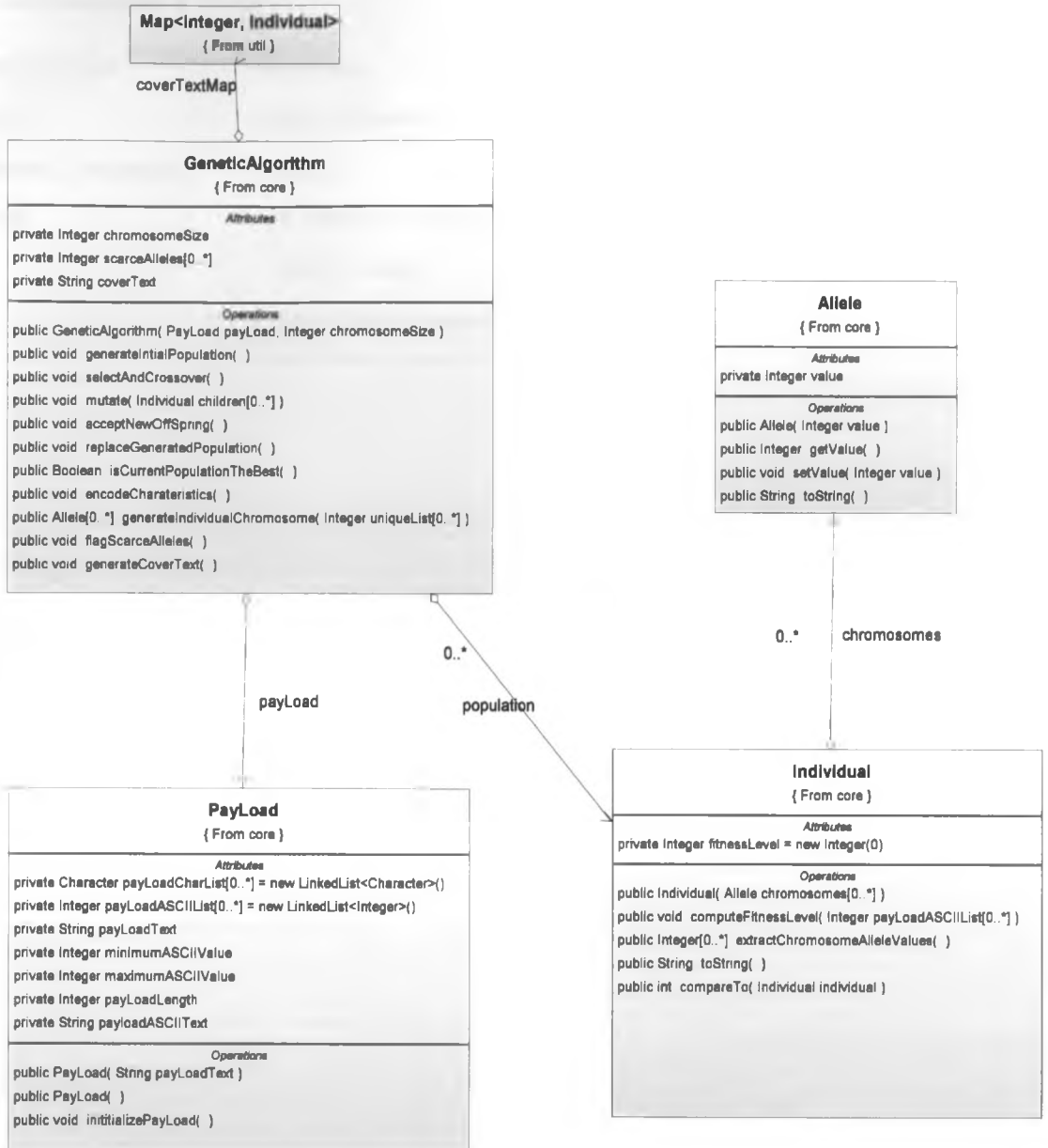
Level 1: Interactive diagram



Level 2: Iterative diagram



Level 2: Class Diagram



3.3 Implementation Tools

Operating System: Windows
Approach: Object Oriented
Programming Language: - Java
Case Tools
- NetBeans Version 7.0.1 IDE
- GUI: Swings
- Java Development Kit Version 7

CHAPTER FOUR: RESULTS AND EVALUATION

4.1 Implementation

The most important properties of a cover medium is security of information, robustness and amount of data that can be stored inside it, without changing the noticeable properties of the cover.

The implementation of genetic algorithm was done on the cover text; in this case, a set of random numbers was used. The generated cover text depends on the length of the secret message. Once optimal results have been achieved the embedding process begins, to output stego text. An extraction algorithm will be applied to reverse to the original secret message.

4.2 Results

The figure below shows the steganography process of the cover text being passed into the embedding function with the secret message to encode resulting in a stego text containing the hidden message. A key is often used to protect the hidden message. This key is usually a password, so this key is also used to encrypt and decrypt the message before and after embedding.

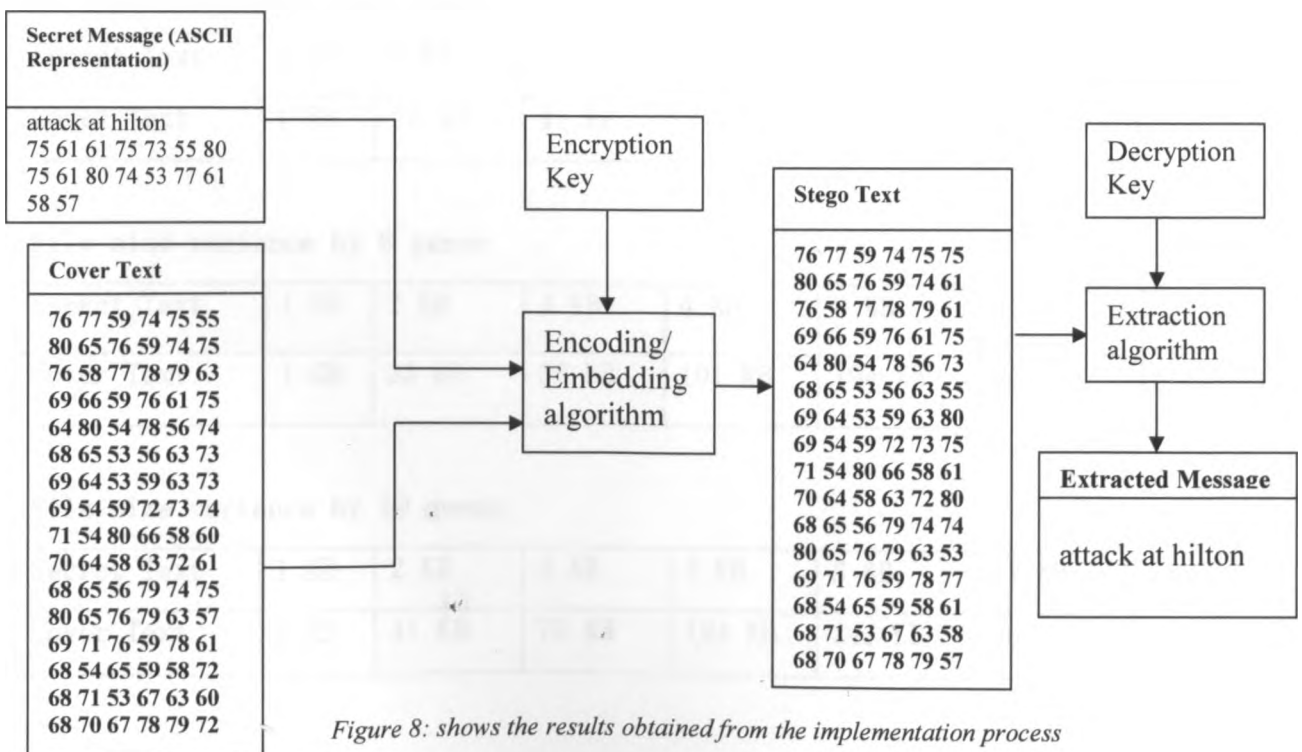


Figure 8: shows the results obtained from the implementation process

4.3 Analysis of System Results

The system results were looked at in terms of security, capacity of hidden message and robustness. The approach used was found to satisfy both security aspects, hiding capacity requirements and minimal embedding time. It generated the stego text with minimum degradation and is not revealing to people about the existence of any hidden data, therefore maintaining its security. The analysis was done in two ways:

- a) Varying the size of Secret message
- b) Varying the chromosome length

An increase in the size of the secret message shows that there is an increase in the size of the generated cover text. A chromosome of 4 bits or 4 genes was used on a population of 1KB of file size to generate a cover text of 1KB while a chromosome of 20 bits or 20 genes on population of 1KB of file size generated a cover text of 3KB.

File size variance by 4 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	1 KB	16 KB	28 KB	49 KB	95 KB	145 KB

File size variance by 6 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	1 KB	25 KB	42 KB	74 KB	143 KB	219 KB

File size variance by 8 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	1 KB	33 KB	56 KB	101 KB	194 KB	297 KB

File size variance by 10 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	2 KB	41 KB	70 KB	124 KB	241 KB	367 KB

File size variance by 12 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	2 KB	49 KB	84 KB	149 KB	289 KB	441 KB

File size variance by 14 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	2 KB	58 KB	98 KB	176 KB	340 KB	515 KB

File size variance by 16 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	2 KB	65 KB	112 KB	299 KB	386 KB	589 KB

File size variance by 18 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	2 KB	74 KB	126 KB	225 KB	435 KB	663 KB

File size variance by 20 genes

Secret Text	1 KB	2 KB	3 KB	4 KB	8 KB	12 KB
Cover Text	3 KB	82 KB	140 KB	250 KB	483 KB	737 KB

This graph below shows that when the chromosomes size was varied, there was a difference in size of the cover text generated.

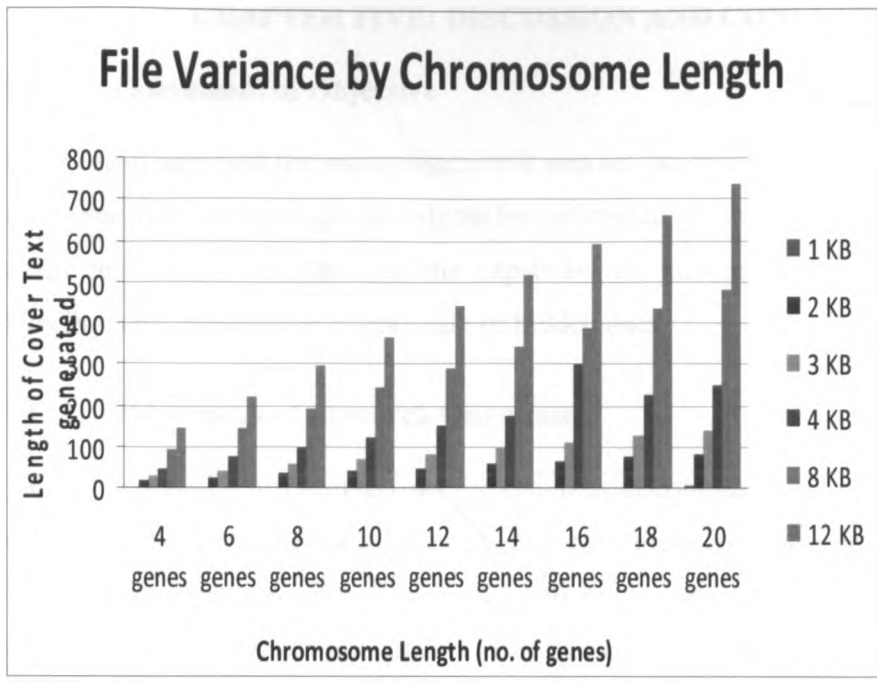


Figure 9: Bar Graph representing file size variance by chromosome length

From the results obtained, it was found that best population size depends on the length of encoded message. That is a chromosome with 4 bits/genes, the population should be say 4 also a chromosome of 20 bits/genes should have a population size of 20.

In relation to the embedding time, the results revealed that given a chromosome of 4 bits/genes on a population size of 4 the embedding time would be much faster than say the same no. of genes on a larger population size.

i.e. $4 \text{ bits/genes} * \text{Population size } (4) = 16$

When the numbers of chromosomes were increased to 20 bits/genes on the population size of 4 the embedding time was higher as compared to that of 4 bits/genes.

i.e. $20 \text{ bits/genes} * \text{Population size } (4) = 80$

This means that for best performance and/or speed of finding a solution, the population size should be almost equal to the size of chromosome.

CHAPTER FIVE: DISCUSSION AND CONCLUSION

5.1 Achievement of Objective

This project achieved the main objective it was set out to carry, this was to develop a tool to be used to implement genetic algorithm technique on the cover text so as to produce a secure and robust tool that has the capability of reducing the probability of message detection and increase the overall rate of hidden data.

5.2 Achievement of Research Questions

Online users find it difficult to trust the channels of communicating secret messages. This is because the available techniques are prone to attackers who intercept the message to reveal it hence no security. This project therefore set out to investigate the available algorithm used to secure messages and in this case Steganography. Text Steganography was therefore chosen from other cover mediums (video, image, audio), because of its smaller memory occupation and simpler communication. Genetic algorithm approach is not prone to visual, structural and statistical attack because of its use of random numbers and generation of random numbers between the minimum and maximum values of the secret message. Also there are two layers of security being used; use of playfair encryption method and then conversion of secret message to ASCII to generate the cover medium to be used to embed the secret message bits/numbers

5.3 Discussion of Results in relation to objectives

5.3.1 Comparison of existing text based Steganography and the technique used in this project, in relation to robustness and capacity of hidden message.

A) Robustness

Robustness is the ability of a hidden message not to be detected either through visual, semantic or statistical attack. The steganographic techniques available are prone to these attacks, unlike genetic algorithm approach which makes use of random numbers, this is explained below:

Visual attack

The genetic algorithm approach used in this project is not prone to visual attacks because of its use of numbers. This is not the case for Format-Based technique that deals with modifications of existing text in order to hide the steganographic text by resizing of fonts, insertion of spaces or non-displayed characters, deliberate misspellings distributed throughout the text and resizing the fonts among others. Insertion of spaces where extra space(s) between words is used, one space means that the transmitted information bit is \0", and two spaces mean \1", which can easily be detected. Deliberate misspellings when writing words, such as: *\How is you" to \How iz you"*. The presence of errors in a document may raise curiosity by someone intercepting the message.

Statistical attack

Character generation takes the statistical properties of word-length and letter frequency to create "words" (with no lexical value) which will appear to have the same statistical properties as actual words in a given language. These words might convince a computer which is only doing statistical analysis (and this is much less likely now that we are in an age where enormous dictionaries can be used to check the validity of words), but has clear problems with modern computer systems in terms of appearing suspicious.

Word Sequence: classifying words and noticing extremely unlikely patterns (e.g. too many verbs or determiners in a row, no prepositions) within sequences of a certain length may be enough to alert the attacker to anomalous behaviour.

Structural attack

Linguistic based methods deal with both modifications of syntax and semantics of words and sentences to hide messages. Grammar-checkers used by modern word processors may be helpful tools in discovering ungrammatical texts. While legitimate ungrammatical texts certainly exist, given a certain context and threshold, such methods could be used to flag texts with no syntactic structure for further attention.

An example of changing the syntax nature of a sentence:

The boy was chased by a dog.

The dog was chased by a boy.

The problem with interchanging words is that in the long run the sentences might not make sense, hence prone to an attacker.

An example of changing the semantic nature of a sentence:

Tom surrendered

Tom gave up

Here, the word surrendered is interchanged with words that have the same meaning. For example if the characters to be hidden is 101, then the option is to look for similar words that can store 101, assuming in this case *Tom gave up* (*mep*) has the equivalent bits to store the secret word.

B) Capacity of Hidden Message

The existing techniques are tedious to embed long messages and in some cases the meaning of the cover message change completely until no sense can be made out of it. This is not likely to occur in the approach proposed in this project because; one the computer does all the work once the parameters for getting the population, fitness and mutation values are pre-determined. Format-Based Method can not be used to hide a long message as it is cumbersome and with its high affinity for visual attack, it can not be effective. On the other hand, using linguistic method will mean having a very long cover text that is prone to both syntactic and semantic errors and therefore also not very effective to use.

5.3.2 Introduce genetic algorithm approach to text based Steganography, so as to produce a secure and robust Steganography tool.

Genetic algorithm was implemented on the cover text generated from the minimum and maximum values of the secret message (ASCII representatives). Before this, the secret message was encrypted using the Playfair encryption method, to add an extra layer of security to the message.

5.3.3 Develop a prototype that will represent the use of genetic algorithm in text Steganography.

A tool was developed to demonstrate the application of genetic algorithm to text Steganography. The tool was developed and implemented using Java as a platform on a Windows machine.

5.3.4 Analyse the developed tool and algorithm used experimentally to find out if this proposed method works properly and is considered to give almost the optimum solution.

Several experimental results obtained from the tool confirm that it produces the correct results intended and also depending on the chromosome length to the size of secret message, there is an increase in performance (in terms of speed) in getting optimal solutions.

5.4 Comparisons of some Text Steganography Tools

****GATS – Genetic Algorithm Based Text Steganography Tool**

	GATS	wbStego	SNOW	Stego
Use of encryption/ decryption key	Yes	Yes/No	Yes/No	Yes
Cover file	System generated	Not System generated	Not System generated	Not System generated
File types	.txt	Image, pdf, txt	-	-
Visibility of secret message	Not visible	Not visible	visible	Not visible
Type of encryption	Playfair	Various	ICE- Information Concealment Engine 64 BIT private key	-
platform	JAVA: WIN	WIN : Delphi	C/C++: DOS WIN	C: DOS

5.5 Challenges

The first challenge faced was creating a robust genetic algorithm approach to represent the problem. The approach was to accept random changes such that fatal errors or noise results do not consistently occur. To achieve this numbers in form of integers was used, where each number represented some aspect of a candidate solution and mutation was consequently introduced randomly. The method of obtaining the fitness value was looked into carefully and frequently evaluated to ensure that the solution obtained equates to a better solution for the given problem.

Secondly, the size of the population, the rate of mutation and crossover, the type and strength of selection needed to be chosen with care. A small population size would not explore enough of the solution space to consistently find good solutions. If the rate of genetic change is too high or the selection scheme is chosen poorly, the population would enter into the problem of local min-max or otherwise termed as premature convergence, hence producing wrong results. For example, given an individual that is more fit than most of its competitors emerging early on in the course of the run, it may reproduce so abundantly that it drives down the population's diversity too soon, leading the algorithm to converge on the local optimum that that individual represents rather than searching the fitness landscape thoroughly enough to find the global optimum (Forrest, 1993; Mitchell, 1996). This problem was found to be common in small populations, where even chance variations in reproduction rate may cause one genotype to become dominant over others. To overcome this challenge the approach used in this project used random numbers both at the initial generated population and during mutation where a gene in a child is selected randomly for mutation. Also a fitness value was used during selection process of parent for crossover, where only the fit individuals are allowed to crossover hence only optimal solutions achieved.

5.6 Conclusion

The outcome of the system evaluation showed that the genetic algorithm approach used in this project is not prone to visual attacks because of its use of numbers. This project introduces the use of genetic algorithm in text steganography. Effective optimization, security and robustness are achieved. The experimental results showed that this approach works properly and is considered to give almost the optimum solution within a small amount of time.

Future work can be focussed on exploring other search heuristics algorithm with an aim of improving the efficiency of the proposed algorithm in terms of robustness and capacity of hidden message.

In addition, this technique can be extended to other types of files.

REFERENCE

- Alla K. and Prasad R.S.R (2009) An Evolution of Hindi Text Steganography: Sixth International Conference on Information Technology New Generations, 2009 (ITNG '09), Digital Object Identifier: 10.1109/ITNG.2009.41, 2009, Page(s): 1577 - 1578.
- Bhattacharyya S., Banerjee I. and Sanyal G. (2010) A Novel Approach of Secure Text Based Steganography Model using Word Mapping Method (WMM): International Journal of Computer and Information Engineering.
- Davida G., Chapman M. and Rennhard M. (2001) A practical and effective approach to large-scale automated linguistic Steganography: In Proceedings of the Information Security Conference.
- Forrest and Stephanie (1993) Genetic algorithms: Principles of Natural Selection as applied to Computation Science.
- Huang D. and Yan H. (2001) Inter word Distance Changes Represented by Sine Waves for Watermarking Text Images: IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 12, December 2001, pp. 1237-1245.
- Jacobson I. (1994) Object Oriented Software Engineering: Use Case Approach, published by Addison-Wesley in 1994.
- Katzenbeisser S., Fabien A.P. and Petitcolas (2000) Information Hiding: Techniques for Steganography and Digital Watermarking.
- Krenn J.R. (2004) Steganography and Steganalysis.
- Kwan M. (1998) SNOW [online] <http://www.darkside.com.au/snow/index.html> (Accessed date /March/2012).
- Low S.H., Maxemchuk N.F., Brassil J.T., and O'Gorman L. (1995) Document marking and identification using both line and word shifting: Proceedings of the Fourteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '95), vol.2, 2-6 April 1995, pp. 853 - 860.
- Lou D.C., Liu J.L. & Tso H.K. (2008) Evolution of Information – Hiding Technology. In Nemati H. (Ed.): Premier Reference Source – Information Security and Ethics: Concepts, Methodologies, Tools and Applications, Volume 1, Chapter 1.32. New York: Information Science Reference.
- Maher K., Texto [online] <http://www.ijtc.com/Security/stegtools.htm> (Accessed 23/March/2012)
- Memon J.A., Khowaja K., and Kazi H., Evaluation of steganography for Urdu /Arabic text: Journal of Theoretical and Applied Information Technology, pp 232-237.
- Mitchell and Melanie (1996) An Introduction to Genetic Algorithms: MIT Press.

Moerland T. (2003) Steganography and Steganalysis.

Nameer N. (2007) Hiding a Large Amount of Data with High Security Using Steganography Algorithm: Applied Computer Science, Faculty of Information Technology, Philadelphia University, Jordan.

Niimi M., Minewaki S., Noda H., and Kawaguchi E. (2003) A Framework of Text-based Steganography Using SD Form Semantics Model: Pacific Rim Workshop on Digital Steganography 2003, Kyushu Institute of Technology, Kitakyushu, Japan, July 3-4, 2003.

Pori L.Y. and Delina B. (2008) Information Hiding-A New Approach in Text Steganography: Faculty of Computer Science and Information Technology University of Malaya, Kuala Lumpur, Malaysia. Applied Computer & Applied Computational Science (ACACOS '08), Hangzhou, China, April 6-8.

Samir B., Tuhin P. and Raychoudhury A. (2010) Genetic Algorithm Based Substitution Technique of Image Steganography: Journal of Global Research in Computer Science, Volume 1, No.5.

Shaifizat Mansor, Roshidi Din & Azman Samsudin, Analysis of Natural Language Steganography: International Journal of Computer Science and Security (IJCSS), Volume (3): Issue (2) 113.

Shirali-Shahreza M. (2008) Text Steganography by Changing Words spelling: Computer Science Department Sharif University of Technology Tehran, IRAN.

Shirali-Shahreza M. (2008) Text Steganography by Changing Words Spelling: International Journal of Advanced Communication Technology, 2008 (ICTACT 08), Volume: 3, Digital Object Identifier: 10.1109/ICTACT.2008.4494159, 2008, Page(s): 1912 - 1913.

Shirali-Shahreza M.H., and Shirali-Shahreza S., A (2006) New Approach to Persian/Arabic Text Steganography: Proceedings of 5th IEEE/ACIS international Conference on Computer and Information Science and 1st IEEE/ACIS.

Wang Z.H., Chang C.C., Kieu D., and Li M.C. (2009) Emoticon-based Text Steganography in Chat: Second Asia-Pacific Conference on Computational Intelligence and Industrial applications.

Walker J. (1997) Steganosaurus [online] <http://www.fourmilab.to/stego/> published by Walker J. (Accessed 23/March /2012)

Zamani M., Manaf A.A., Ahmad R. B., Zeki A. M. and Abdullah S. (2009) A Genetic-Algorithm-Based Approach for Audio Steganography: World Academy of Science, Engineering and Technology.

Appendix 1: User Manual

A) Steganography

Step 1: Open Program and go FILE-->>Steganography.



Figure 10: GATS interface for encryption

For encryption use a key that has no repetitive characters in it e.g. CHARLES

Step 2: Dialog box to open the secret message stored in .txt file



Figure 11: how to open a saved .txt file

Step 3: Conversion of the Secret Message into its ASCII representation

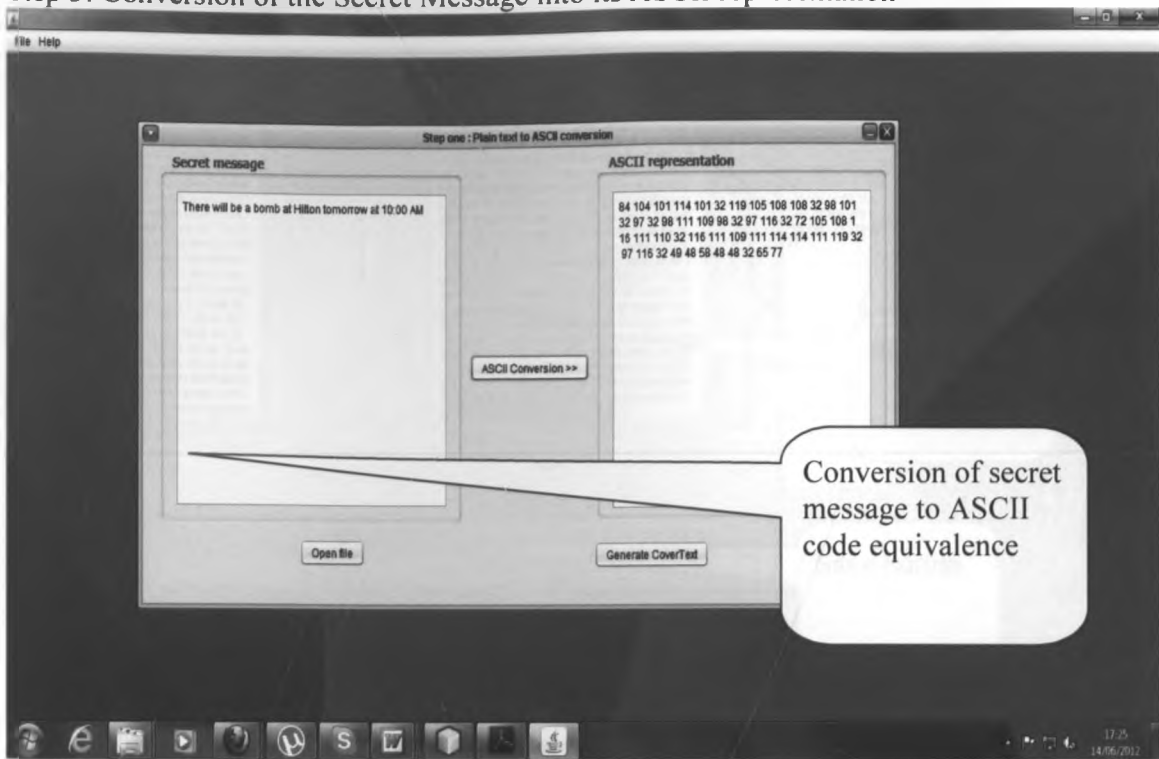


Figure 12: conversion of secret message to its ASCII representative

Step 4: Generate Cover Text



Figure 13: generation of cover text

Step 5: Generate Stego Text

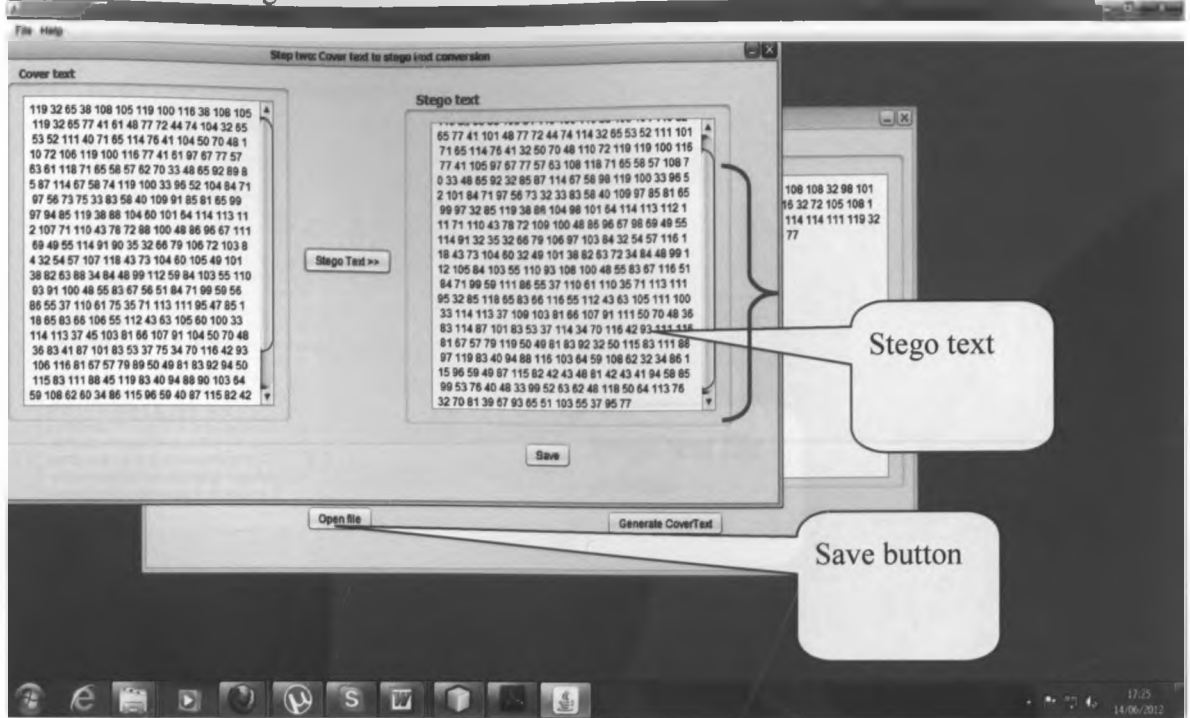


Figure 14: generation of stego text

B) Steganalysis

Step 1: Open the Steganalysis application. File-->> Text-->>Steganalysis



Figure 15: opening steganalysis interface

Step 2: Dialog box to open the Stego Text file

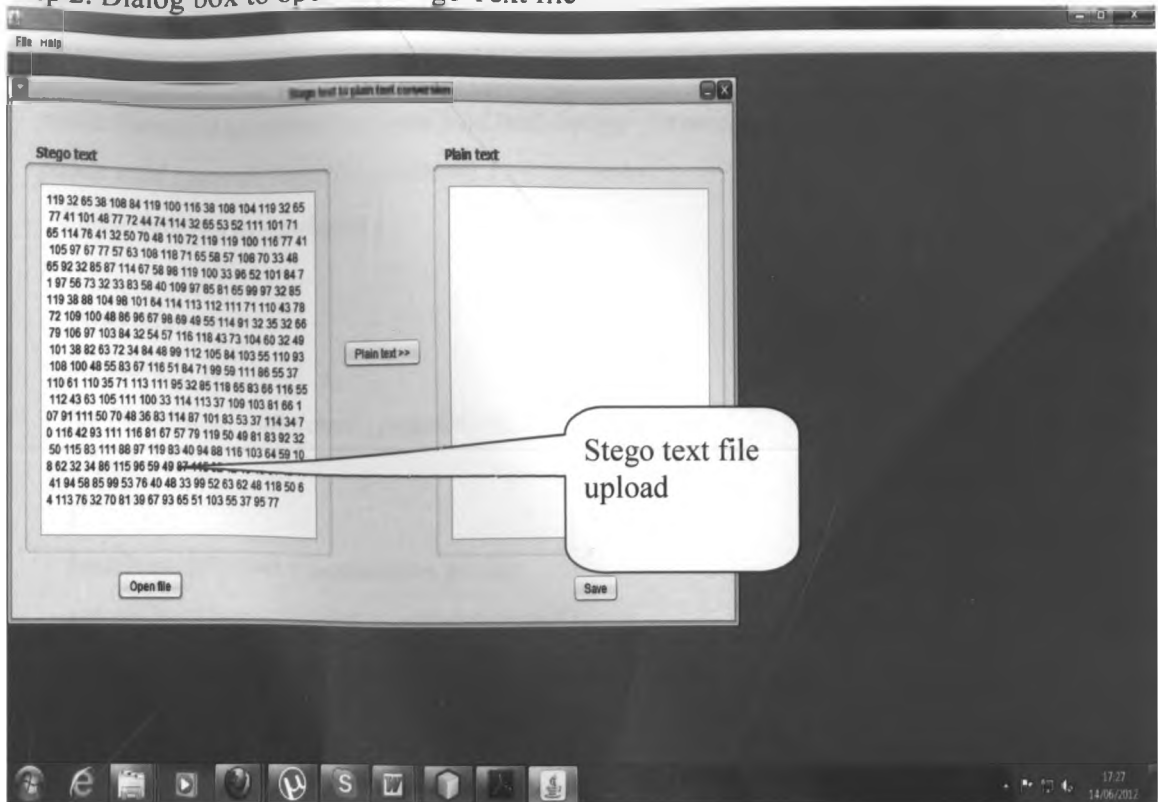


Figure 16: opening the saved stego text

Step 3: Conversion of Stego text to Plain Text

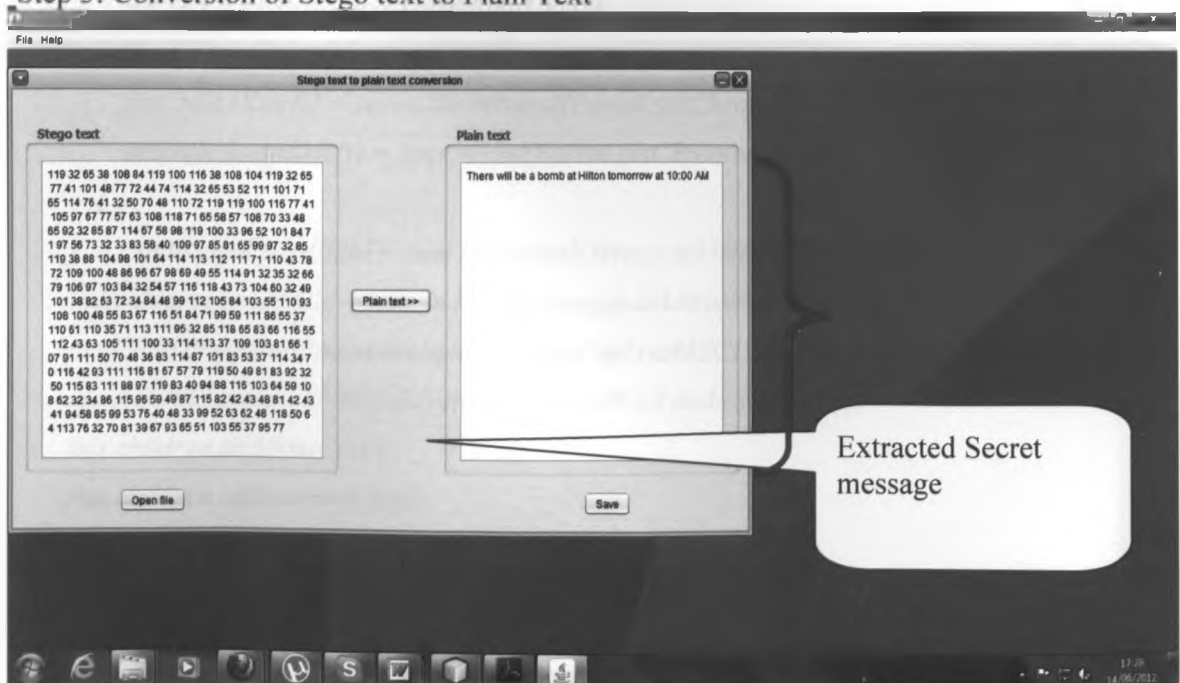


Figure 17: extraction of the secret message

Appendix 11: Source Code

```
/** GA ALGORITHM IMPLEMENTATION
public GeneticAlgorithm(PayLoad payLoad, Integer chromosomeSize) {
public void generateInitialPopulation() {
    Collections.sort(population);
}
    generation++;
    int index = 1;
    for (Individual individual : population) {
        index++;
    }
    bestOverallParent = population.get(0);
    secondBestOverallParent = population.get(1);
    Allele[] firstChildAllele = new Allele[chromosomeSize];
    Allele[] secondChildAllele = new Allele[chromosomeSize];
{
    if (i < chromosomeSize / 2) {
        firstChildAllele[i] = bestOverallParent.getChromosomes().get(i);
        secondChildAllele[i] = secondBestOverallParent.getChromosomes().get(i);
    } else {
        firstChildAllele[i] = secondBestOverallParent.getChromosomes().get(i);
        secondChildAllele[i] = bestOverallParent.getChromosomes().get(i);
    }
}
    Individual firstChild = new Individual(Arrays.asList(firstChildAllele));
    Individual secondChild = new Individual(Arrays.asList(secondChildAllele));
    firstChild.computeFitnessLevel(payLoad.getPayloadASCIIList());
    secondChild.computeFitnessLevel(payLoad.getPayloadASCIIList());
    this.children.add(firstChild);
    this.children.add(secondChild);
    mutate(children);
} public void mutate(List<Individual> children) {
    }
    childAllele.add(allele.getValue());
```

```

    }
    scarceAlleles.get(scarceAlleleIndex.intValue());
        childAllele.add(scarceAlleles.get(scarceAlleleIndex.intValue()));
    for (Integer value : childAllele) {
        childAlleleList.add(new Allele(value));
    }
    child.setChromosomes(childAlleleList);
    child.computeFitnessLevel(this.payLoad.getPayLoadASCIIList());
    mutatedChildren.add(child);
    this.scarceAlleles.remove(scarceAlleleIndex.intValue());
}
if (mutatedChildren.size() == 1 && scarceAlleles.size() == 0) {
    mutatedChildren.add(children.get(1));
}
this.children.clear();
this.children = mutatedChildren;
}
} public void acceptNewOffSpring() {
    this.population.addAll(children);
}
public void replaceGeneratedPopulation() {
    int index = 0;
    for (Individual individual : population) {
        index++;
    }
} public Boolean isCurrentPopulationTheBest() {
{
    return false;
} else {
    return true;
}
} public void encodeCharateristics() {
    } this.population = finalGeneration;
} chromosomes.add(new Allele(value));
} return chromosomes; }

```