M.Tech. (Computer Science) Dissertation Series

# Distinguishing between competing crypto-algorithms for the known Ciphertext case: a statistical approach

A dissertation submitted in partial fulfillment of the requirement for the M.Tech. (Computer Science) degree of the Indian Statistical Institute

*By*

Deepak Kumar

Under the supervision of

**Prof. Bimal Roy**
Applied Statistics Unit

INDIAN STATISTICAL INSTITUTE
203, Barrackpore Truck Road
Kolkata 700 108
11th July 2005

## Certificate of Approval

This is to certify that the dissertation titled *Distinguishing between competing crypto-algorithms for the known ciphertext case : a statistical approach* submitted by Deepak Kumar, towards the partial fulfillment of the requirements for the Degree of M.Tech in Computer Science at Indian Statistical Institute, kolkata, embodies the work under my supervision. His work is satisfactory.

Bimal Roy

(Supervisor)

15.7.05

Professor
APPLIED STATISTICS UNIT
INDIAN STATISTICAL INSTITUTE
203, Barrackpore Trunk Road,
Kolkata-700108

(External Examiner)

## ACKNOWLEDGEMENT

I take this opportunity to express my deep sense of gratitude and indebtedness to my guide Prof. Bimal Roy of Applied Statistics Unit, Indian Statistical Institute, Kolkata for his generous and whole hearted support to me in completing my M.Tech dissertation work. I am also highly grateful to Prof. Ayanderanath Basu, Applied Statistical Institute, Kolkata for his guidance.

I also wish to place on record my sincere thanks the Research Scholars of Applied Statistics Unit for the kind help and cooperation that they extended to me throughout the tenure of my dissertation.

I would like to express my profound gratitude to all other teachers of Indian Statistical Institute, Kolkata and sincere thanks to all of my classmates who were the great source of inspiration during the M.Tech programme.

**(Deepak Kumar)**

# Contents

**1 Preliminaries** **3**

1.1 Statistical and mathematical terms . . . . . 3

1.1.1 Probability . . . . . 3

1.1.2 Chi-square distance function . . . . . 4

1.1.3 Frequency based randomness test . . . . . 4

1.1.4 Primitive polynomial . . . . . 5

1.2 Cryptology related terms . . . . . 5

1.2.1 Cryptology . . . . . 5

1.2.2 Block cipher and stream cipher . . . . . 6

1.3 Classification related terms . . . . . 7

1.3.1 Pattern Classification . . . . . 7

1.3.2 Classification approaches . . . . . 7

**2 Introduction to the problem** **9**

2.1 Classifier design . . . . . 9

2.2 Adjusting parameters in classifier design . . . . . 11

**3 Transition from simpler problem to harder problem** **15**

3.1 General assumptions . . . . . 15

3.2 Problem 1 . . . . . 16

3.3 Problem 2 . . . . . 17

3.4 Problem 3 . . . . . 19

3.5 Problem 4 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21

**4 Brief detail of used Ciphers 25**

4.1 Block Ciphers . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

4.1.1 Vigenere Cipher . . . . . . . . . . . . . . . . . . . . . . . . . . 25

4.1.2 Fiestel Cipher with a simple Fiestel function simpler than DES . . 26

4.1.3 SPN Cipher based on 8 bit boxes with six rounds and with a non linear S box function . . . . . . . . . . . . . . . . . . . . . . . . 27

4.2 Stream Ciphers . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29

4.2.1 LFSR based linear Stream Cipher on GF(2) . . . . . . . . . . . . 29

4.2.2 non LFSR based Stream Cipher . . . . . . . . . . . . . . . . . . 30

4.2.3 Nonlinear Filter Generator of length 64 on GF(4) based on LFSR 30

4.2.4 VMPC cipher . . . . . . . . . . . . . . . . . . . . . . . . . . . 31

**5 Experimental Results 32**

5.1 Problem 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

5.1.1 Frequency based randomness test . . . . . . . . . . . . . . . . . 32

5.2 Problem 2 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

5.2.1 Frequency based randomness test . . . . . . . . . . . . . . . . . 33

5.2.2 Test data using two keys of LFSR . . . . . . . . . . . . . . . . . 34

5.3 Problem 3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 36

5.3.1 Step 1 of Learning Phase . . . . . . . . . . . . . . . . . . . . . . 36

5.3.2 Step 2 of Learning Phase . . . . . . . . . . . . . . . . . . . . . . 37

5.4 Problem 4 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38

5.4.1 Step 1 of Learning Phase . . . . . . . . . . . . . . . . . . . . . . 38

5.4.2 Step 2 of Learning Phase . . . . . . . . . . . . . . . . . . . . . . 39

**6 Final Remarks 40**

# List of Tables

5.1 Frequency based randomness test . . . . . . . . . . . . . . . . . . . . . . . 33
5.2 Chi square distance: row represents class 1 and column represents class 2 34
5.3 Chi square distance: row represents class 1 and column represents class 2 34
5.4 Chi square distance: row represents $1st$ class and column represents $2nd$ class . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
5.5 Chi square distance: row represents $1st$ class and column represents $2nd$ class . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
5.6 Chi square distance: row represents $1st$ class and column represents $2nd$ class . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
5.7 Chi square distance: rows represents $1st$ class and column represents $2nd$ class . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37
5.8 Distribution of class prediction (in percentage) by classifier . . . . . . . . 37
5.9 Distribution of class prediction (in percentage) by classifier . . . . . . . . 38
5.10 Distribution of class prediction (in percentage) by classifier . . . . . . . . 38
5.11 Chi square distance: rows represents $1st$ class and column represents $2nd$ class . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
5.12 Distribution of class prediction (in percentage) by classifier . . . . . . . . 39

# List of Figures

4.1 Fiestel Networks . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

4.2 DES encryption diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

4.3 SPN Network with 3 rounds and 4 S boxes . . . . . . . . . . . . . . . . . . 28

4.4 LFSR key stream generator . . . . . . . . . . . . . . . . . . . . . . . . . . 29

## Abstract

In this report, we have described some statistical approaches for the classification of ciphertext in terms of the algorithm used to encrypt it. We have tried to do cryptanalysis using "known ciphertext only" attack. The above mentioned problem can be defined as follows.

"Given a finite set of possible algorithms and a ciphertext with an unknown origin, determine the algorithm which created this ciphertext by distinguishing among a finite set of possible algorithms."

Since, there is no existing statistical literature to solve above mentioned problem, we tackled it from the very beginning. Our effort can be described in brief as follows.

**Formulation of Problem** In the first step, we successfully formulated our problem in the unambiguous and simple manner(section 2).

**Design of Classification Strategy** We viewed this problem as a classification problem (section 2.1). Note that, in an area like cryptology where secrecy and confidentiality are the essence, misclassifying the origin of a ciphertext can lead to a potential disaster. It is desirable that our classifier must have very little probability of misclassification. Since it is very difficult to design a classifier with almost 0% misclassification probability, we slightly modified commonly used classification approach to overcome such harsh restriction.

**Identification of Different Parameters** In general, parameters are terms associated to strategy, which is used to solve given problem. For example, in our Classifier design strategy, parameters are "Choice of training ciphertext", "Design choice for training set and similarity measure" and so on. Till now, we identified 6 different parameters (section 2.2).

**Fixing parameters** To handle a problem at hand, we must fix parameters. To fix it, we used observations obtained in experiment. We also applied our intuition and statistical known behavior of cryptosystems to fix it. For example, we allowed key mixing after getting observation, by experiment, that key may affect the performance of classifier (section 2.2).

**Analysis of behavior of cipher keys** Intuitively, we felt that the key may had significant role in our process. We had analyzed its behavior and found that their were instances when the variation due to the key could hamper the distinguishing process (section 3.3).

**Design classifier using different learning schemes** In this report, we used statistical measures to design classifiers. In particular, we experimented using frequency based learning scheme. We analyzed the effectiveness of "One character based frequency analysis" (section 5.3) and "two characters based frequency analysis" (section 5.4). Also, see (section 2.2). We obtained encouraging result in this analysis (section 3.4 and 3.5).

In this report, we presented four problems. First two problems were meant to analyze the behavior of frequency pattern, key etc. In last two problems, we applied knowledge gained from this analysis for classifier design.

# Chapter 1

# Preliminaries

## 1.1 Statistical and mathematical terms

### 1.1.1 Probability

The term "Probability" is derived from the word "probable". Probable is one of several words applied to uncertain events or knowledge, being more or less interchangeable with uncertain, and doubtful, likely, risky, hazardous, depending on the context. Chance, odds, and bet are other words expressing similar notions. The theory of probability attempts to quantify the notion of probable event (or proposition).

Some useful probability laws are as follows (Refer [1]):

1. The probability of an event is a number between 0 and 1.
2. The probability of an event and its complement must add up to 1.
3. The joint probability of two events is the product of the probability of one of them and the probability of the second, conditional on the first

### 1.1.2 Chi-square distance function

Let n balls be tossed into cells $1, 2...K$ independently such that $p_i$ is the probability for any given ball to land in cell $i$, $i = 1, 2...K$. Let $n_i$ be the number of balls in cell $i$, $i = 1, 2...K$. Then the expected number of balls in cell $i$ ($i = 1..K$) will be

$$E(n_i) = np_i$$

Now suppose that we hypothesize values for $p_1, p_2, ....p_K$ and calculate the expected value for each cell. Certainly, if our hypothesis is true, then the cell counts $n_i$ shouldn't deviate greatly from their expected values $np_i$ ($i = 1, 2...K$). So, we can define the following test statistics, which is a function of the squares of the deviations of the observed counts from their expected values, weighted by the reciprocals of their expected values.

$$X^2 = \sum_{i=1}^{K} \frac{[n_i - E(n_i)]^2}{E(n_i)}$$

Note that here $n_1 + n_2 + ..... + n_K = n$ and degree of freedom is $K - 1$.

Above formulae is known as Pearson's Chi-square distance function (Refer [2]). Equivalently, above formulae can be written as follows.

$$X^2 = n \sum_{i=1}^{K} \frac{[d_i - e_i]^2}{e_i}, \text{ where } d_i = \frac{n_i}{n} \ \ and \ \ e_i = \frac{E(n_i)}{n}$$

Note that there are many other chi-square functions in literature. One such function is likelihood ratio chi-square. It is defined as follows.

$$X^2 = \sum_{i=1}^{K} d_i log(\frac{d_i}{e_i}) \text{ where } d_i = \frac{n_i}{n} \ \ and \ \ e_i = \frac{E(n_i)}{n}$$

### 1.1.3 Frequency based randomness test

Our frequency test uses Pearson's Chi-square distance function for checking randomness of a bit pattern. Note that in the case of monobit test, the value of $K$ is 2. We also applied multibit test. In most cases we used 5 bits. Note that, in this case, $K$ is 32.

We can also apply this approach to find relative distance between two patterns. Here also, our frequency test uses Pearson's chi- square distance function described as below.

$$Chi - Square(F1, F2) = \sum_{i=1}^{K} \frac{(F1_i - F2_i)^2}{F1_i}$$

where $F1_1, F1_2, ...$ are expected frequencies and $F2_1, ...$ are observed frequencies. And $K - 1$ is the degree of freedom.

Note that the chi-square distance function is not symmetric. It is easy to verify that above distance function satisfies two properties.

1. $Chi - Square(F1, F2) \; > \; 0$
2. $Chi - Square(F1, F2) \; = \; 0$ iff $F1 \; = \; F2$

Note that, considering above two properties, Chi Square distance function can be viewed as statistical distance and not as mathematical metric. It is popularly known as "Divergence". Please note that, we continue to call it as distance function in this report.

### 1.1.4 Primitive polynomial

A primitive polynomial is the minimal polynomial of a primitive element of the extension field $GF(p^m)$ (refer [3]). Here $p$ is a prime number and $m$ is +ve integer. Root of primitive polynomial, also known as primitive element, behaves as generator of $GF(p^m)$. A polynomial of degree $n$ is said to be irreducible, if it can't be factorized into two or more non trivial polynomials of degree less than $n$.

## 1.2 Cryptology related terms

### 1.2.1 Cryptology

Cryptology is the study of cryptography and cryptanalysis (refer [4]) (refer [5]).

Cryptography is, traditionally, the study of means of converting information from its normal, comprehensible form into an incomprehensible format, rendering it unreadable without secret knowledge  the art of encryption. All modern cryptography is based on the use of algorithms to scramble (encrypt) the original message, called plaintext, into unintelligible babble, called ciphertext. The operation of the algorithm requires the use of a key.

The study of how to circumvent the use of cryptography is called cryptanalysis, or codebreaking. A cryptanalyst might appear to be the natural adversary of a cryptographer, and to an extent this is true. There are a wide variety of cryptanalytic attacks, and it is convenient to classify them. One distinction concerns what an attacker can know and do in order to learn secret information, e.g. does the cryptanalyst have access only to the ciphertext (Known as ciphertext only attack)? Does he also know or can he guess some corresponding plaintexts (Known plaintext attack)? Or even: Can he choose arbitrary plaintexts to be encrypted (Chosen plaintext attack)?

### 1.2.2 Block cipher and stream cipher

A cipher is an algorithm for encryption and decryption (refer [4]) (refer [5]).

A block cipher is a method of encrypting text (to produce ciphertext) in which a cryptographic key and algorithm are applied to a block of data (for example, 64 contiguous bits) at once as a group rather than to one bit at a time. The DES and AES algorithms are examples of block ciphers.

A stream cipher is a method of encrypting text (to produce ciphertext) in which a cryptographic key and algorithm are applied to each binary digit in a data stream, one bit at a time. Encryption consists of XORing the plaintext bits with the corresponding bits of the keystream; decryption consists of XORing the ciphertext bits with the corresponding keystream bits. This means that a single bit of ciphertext error results in a single bit of plaintext error; this property is useful when the transmission error rate is high.

Stream ciphers are used in applications where plaintext comes in quantities of unknowable length - for example, a secure wireless connection. If a block cipher were to be used in this type of application, considerable bandwidth would end up being wasted by padding, since block ciphers cannot work on blocks shorter than their block size.

## 1.3 Classification related terms

### 1.3.1 Pattern Classification

Pattern is defined as description of object. Pattern class (or simply class) is class of similar patterns. Pattern consists of attributes which are useful for the problem in hand. These attributes are called features (refer [7]) (refer [8]).

The goal of classification is to learn a mapping from the feature space, X, to a label space, Y. This mapping, f, is called a classifier. In the classification, we have access to training samples with known classes. Formally, the problem can be stated as follows.

Given training data, $\{(x_1, y_1), (x_2, y_2), ...(x_n, y_n), \}, x_i \in X \ and \ y_i \in Y, \ i = 1...n$, produces a classifier $f : X \rightarrow Y$ which maps an object $x \in X$ to its classification label $y \in Y$.

### 1.3.2 Classification approaches

There are many approaches for pattern classification. Two approaches known as "Prototype method" and "$K - NN$ method" are as follows.

**Prototype approach** Let $m$ be number of classes. Note that $m$ is known apriori. In the learning phase, we determine single representative pattern for each class. These representatives are known as prototypes. To classify a new pattern with unknown class, we match it with each prototype. We assign it to the class whose pattern is closest. The distance function may be chosen depending on problem domain.

$K - NN$ **approach** Let $K$ be a $+ve$ number. Let $m$ be number of classes. Note that m is known apriori. In the learning phase, we determine many sample patterns for each class. To classify a new pattern with unknown class, we match it with these sample patterns. The distance function may be chosen depending on problem domain. We sort these distances in ascending order. Let $d_1, d_2..., d_K$ be $K$ smallest distances. Now, we count the frequency of each class among these $K$ smallest distances. We assign it to the class whose frequency is highest. Note that, tie may be broken arbitrarily.

# Chapter 2

# Introduction to the problem

Most known attacks assume the knowledge of algorithm used for the encryption process. But in practice, often it is the case that we don't have exact idea about what algorithm has been used. In the era of internetworking, this issue has become more important. We have tried to tackle this problem through a statistical analysis. Above mentioned problem can be defined as follows.

"Given a finite set of possible algorithms and a ciphertext with an unknown origin, determine the algorithm which created this ciphertext by distinguishing among a finite set of possible algorithms."

## 2.1 Classifier design

We formulated it as a classification problem i.e. we assume existence of labelled sample data. This assumption is practical, since encryption algorithms are public. Our approach in brief is as follows.

In the learning phase, significantly large chunk of labelled sample data (ciphertext) has been used to create frequency tables. Each class (algorithm) has its own representative frequency table, popularly known as prototype. This phase has been used to train our classifier.

In the classification phase, for the given ciphertext, we generate its own frequency table. Then, this table is matched with the existing prototypes and the ciphertext is classified to class whose prototype is the closest. In this report, We have used a statistical distance measure known as the Pearson's Chi-Square distance measure in some appropriate sense.

In an area like cryptology where secrecy and confidentiality are the essence, misclassifying the origin of a ciphertext can lead to a potential disaster. It is desirable that our classifier must be very accurate. In other words, it must have a small probability of misclassification. Since it is very difficult to design a classifier with almost 0% misclassification probability, we have slightly modified our above approach without loss of generality. Our modification is as follows.

Let the number of possible algorithms be $K$. Then if a ciphertext chunk with an unknown origin is randomly (with uniform probability distribution) classified, the prior probability of recovering the correct algorithm is $1/K$. Let our classifier's probability of correct classification be $p$.

Also, let probability of misclassification into each individual (incorrect) class be different from $p$.

Mathematically, let $C_1, C_2, ....C_n$ be $n$ classes. And, let, $S$ be the ciphertext to be classified. Let $C_i$ be its correct class, then

$$Prob(S \in C_j|S) \neq p \quad \forall j \neq i$$

To obtain correct class, we proceed as follows.

We break sufficiently large chunk of ciphertext, to be classified, into many small pieces. We apply above classifier to these pieces of ciphertexts and note down its classification results. Then, we calculate probability of membership of ciphertext for each class. We assign ciphertext to the class in which probability is closest to $p$.

Note that this scheme gives almost negligible error of misclassification with few number of experiments, if, for each class, correct classification probability is not very close to misclassification probability to each incorrect class. In other words, probability of membership of ciphertext in correct class, $p$, should be well separated from each probability of membership of ciphertext in incorrect classes.

This scheme can be further simplified if the following relationship holds.

$$Prob(S \in C_j|S) < p \quad \forall j \neq i$$

In this case, instead of calculating the probability of membership, we can classify the ciphertext to the class which is assigned maximum number of times.

Above modification is quite applicable to real use. We also got classifier which satisfies these conditions.

## 2.2 Adjusting parameters in classifier design

During the classification process, we came across many parameters. These parameters are as follows.

1. **Choice of training ciphertext**

   The ciphertext should be representative of class. In other words, its corresponding frequency pattern (table) should correspond, in particular, to the frequency pattern of the ciphertext class. One issue is that training ciphertext size shouldn't be too small. We checked with many ciphertext chunks and fixed it according to performance of classifier. We have following observations.

   (a) The size of ciphertext depends on the classification problem in hand. In particular, it depends on algorithms selected for classification.

   (b) It also depends on the kind of frequency analysis used. We have experimented with two classification schemes. First classification scheme uses single character frequency analysis. The second classification scheme uses two character classifier. Note that, in this report, we viewed ciphertext as stream of 5 bit characters. Note that, in the single character frequency analysis, number of different frequencies are 32 and, in the case of two character frequency analysis, number of different frequencies are 32 × 32. We applied Pearson's Chi-square test (section 2.2) in both case.

   It is clear that ciphertext size of longer size is needed in the case of two character frequency analysis.

2. **Choice of number of classes**

   It is generally obtained from the application domain. In the initial phase of our analysis, we have selected 2 classes. This analysis was done for capturing the behavior of classifier for simpler case. After getting encouraging result, we proceeded with 4 popular cryptosystems. Then, we tried with 3 seemingly secure cryptosystems.

3. **How to deal with keys**

   We know that keys play an important role in variation of ciphertext frequency. For a known cryptosystem (LFSR based Stream Cipher), we generated two different ciphertexts using two different keys (section 3.3). Through the frequency analysis, it is observed that their frequency patterns are well separated (section 5.2). Here Chi square distance function was used. To avoid the problem of key effect, we adopted the method of key mixing. In the case of frequency analysis, we merged the ciphertexts obtained using a number of different keys. This can be viewed as averaging of frequency patterns. Here we used 4 random keys for generating ciphertexts. All these ciphertexts were merged using concatenation operation. Note that number of keys may be adjusted as needed.

4. **Choice of frequency analysis method**

   There are two issues here.

   (a) We can analyze using binary level frequency analysis. We have rejected this analysis observing following comments.

      **observ1** Since in this case, pattern has only two features namely frequency of $0's$ and $1's$, it is very difficult to capture representative pattern.

      **observ2** We know that for random behavior both $0's$ and $1's$ must be equally probable. This is an initial test for randomness. Most of the cryptosystems nowadays fulfill this test. So, obtained frequency table will be most likely not well separable.

So, we decided to use combination of 5 bit characters for our analysis purpose. It has 32 different frequencies. Please note that in our analysis, we have dealt with text documents containing 26 English characters and 6 other symbols. So, taking 5 bit character has advantage that we can view encrypted (cipher-text) document as a scrambled form of plain text document, which is quite natural.

(b) We tried with single character and multiple character frequency analysis. In the single character, the frequency pattern contains 32 features. In the two character frequency analysis, it contains 32 × 32 different features. In our experiment, we observed slightly better classifier performance in the case of two character frequency analysis. We have interpreted these observations in the following manner.

Since these cryptosystems are not truly random under these tests (one and two character frequency analysis), they exhibited some distinguishable pattern. Also, it indicates that cryptosystems under consideration are weaker with respect to two character frequency analysis.

## 5. Design choice for training set and similarity measure

We used prototype method. In this method, we determined prototypes of each class using training sample. For the classification of new sample, we compare its frequency pattern with frequency patterns of prototypes of each class. Another approach can be as follows.

We can store many frequency pattern for each class. For the classification of any new pattern, we can use K-NN classification strategy.

As a similarity measure, we used following form of chi-square function.

$$Chi - Square(F1, F2) = \sum_{i=1}^{n} \frac{(F1_i - F2_i)^2}{F1_i}$$

where $F1_1, F1_2, \ldots$ are expected frequencies and $F2_1, \ldots$ are observed frequencies. And $n - 1$ is the degree of freedom.

Note that there are many other similarity measures present in the literature (section 1.1.2). For example, other similarity measure such as likelihood ratio chi-square may be used.

6. **Choice of ciphertext size to be classified**

The choice of ciphertext size should be such that it must capture the value of all its feature (frequencies in this case). We tried with 1000 character, 2000 characters and 5000 character ciphertexts. Note that one character has 5 bits. We observed that higher size of ciphertext resulted in slightly better result. In this case of 1000 characters and 5000 characters, the improvement was more clear. Note that the choice of ciphertext size depends on the particular problem at hand.

# Chapter 3

# Transition from simpler problem to harder problem

## 3.1 General assumptions

During formulation of our analysis strategy, we made following general assumptions.

1. we dealt with text documents containing 26 English characters and 6 other symbols. These symbols are a-z, comma(,), semicolon(;), fullstop(.), space( ),exclamation mark(!) and question mark(?). This assumption allows us to view text document as stream of 5 bit characters. Encrypted (ciphertext) document can also be viewed as scrambled form of text document. Please note that this restriction is reasonably fair (observe that semantic of a text document is preserved by this set of symbols). While customizing meaningful text documents, we observed that it didn't need significant amount of modification.

2. We have dealt with meaningful text documents.

## 3.2 Problem 1

We started with relatively simple problem. In this experimental problem, classes are as follows.

1. $1^{st}$ class is Vigenere cryptosystem. It is a block cipher
2. $2^{nd}$ is an stream cipher whose key generating recurrence relation is as follows.

$$X_m = (aX_{m-1} + b)\%N \quad \text{where} \quad gcd(a, N) = 1 \;\; and \; X_0 \text{ is given}$$

In this problem, we applied one character based frequency analysis. Note that, in this case, frequency pattern has 32 features. In the beginning, we analyzed frequency pattern of ciphertexts obtained from above mentioned two cryptosystems. We experimented using following parameters.

1. Size of each ciphertext chunk is 1000 characters of 5 bits.
2. We didn't use key mixing. In other words, each ciphertext chunk was obtained using one key only.
3. We applied one character based frequency analysis.
4. We used chi-square distance function whose formula is as follows (section 1.1.2).

$$Chi - Square(F1, F2) = \sum_{i=1}^{n} \frac{(F1_i - F2_i)^2}{F1_i}$$

where $F1_1, F1_2, ...$ are expected frequencies and $F2_1, ...$ are observed frequencies. And $n-1$ is the degree of freedom. $n$ is the problem of different 5 bit permutations, with repetitions, of $0's$ and $1's$.

We analyzed frequency patterns as follows.

1. We applied frequency based randomness test and observed that both cryptosystems had significantly different behavior. Note that we repeated this test for many different chunks.

2. We compared frequency patterns corresponding to these two classes. We observed that their frequencies were well separated.

We interpreted above observation as an indication of possibly good classifier. We obtained 100% correct classification.

## 3.3 Problem 2

Here we experimented with 2 different classes. In this experimental problem, we replaced Vigenere cryptosystem with relatively more secure LFSR based stream cipher. For the completion sake, we describe 2 classes below.

1. $1^{st}$ class is a stream cipher whose key generating recurrence relation is as follows.

$$X_m = (aX_{m-1} + b)\%N \quad \text{where } gcd(a, N) = 1 \quad \text{and } X_0 \quad \text{is} \quad \text{given}$$

2. $2^{nd}$ class is a stream cipher using LFSR (Linear Feedback Shift Register) as random key generator. Here we have used $GF(2)$ as base field.

In this problem also, we applied one character based frequency analysis. Note that in this case, frequency pattern has 32 features. In the beginning, we analyzed frequency pattern of ciphertexts obtained from above mentioned two cryptosystems. We experimented using following parameters.

1. Size of each ciphertext chunk is 1000 character of 5 bits.

2. We didn't use key mixing. In other words, each ciphertext chunk was obtained using one key only.

3. We applied one character based frequency analysis.

4. We used 8 bit LFSR, 16 bit LFSR and 24 bit LFSR respectively.

5. We used chi-square distance function whose formula is as follows [1.1.2].

$$Chi - Square(F1, F2) = \sum_{i=1}^{n} \frac{(F1_i - F2_i)^2}{F1_i}$$

   where $F1_1, F1_2, ...$ are expected frequencies and $F2_1, ...$ are observed frequencies. And $n-1$ is the degree of freedom. $n$ is the problem of different 5 bit permutations, with repetitions, of $0's$ and $1's$.

In the first step, we analyzed frequency patterns as follows.

1. We applied frequency based randomness test and observed that both cryptosystems had significantly different behavior. Note that we observed that 16 bit LFSR and 24 bit LFSR behaved nearly random whereas 8 bit LFSR and stream ciphers behaved as non random. Note that we repeated this test for many different chunks.

2. We compared frequency patterns corresponding to these two classes. We observed that frequencies corresponding to stream cipher were well separated (easily distinguishable) from that of 8 bit, 16 bit and 24 bit LFSR based stream ciphers respectively.

We interpreted above observation as an indication of possibly good classifier. We obtained 100% correct classification.

In the next step, we experimented to analyze behavior of key in the classification process. For this purpose, we slightly modified our above experiment to include one more class. In this $3^{rd}$ class, we used different initial key but same LFSR having same primitive polynomial. We obtained following observations.

1. In the case of 8 bit LFSR, frequency patterns corresponding to two classes, namely class 2 (LFSR with one key) and class 3 (LFSR with other key) are well separated.

2. We compared two classes at a time. We observed that in the case of 8 bit LFSR, both classes 2 and class 3 are more closer to class 1 (stream cipher) than their mutual distance.

We interpreted above observations as follows.

Key may play a significant role in the classification process and it may badly affect its performance. For the experiment, we used "test ciphertext chunks" encrypted with 8 bit LFSR using same single key. We observed that many chunks were misclassified to class 1. To neutralize effect of key, we proposed key mixing.

## 3.4 Problem 3

Here we experimented with 4 different popular and seemingly very secure cryptosystems. We considered 2 stream ciphers and 2 block ciphers. We applied knowledge obtained by experiment with Problem 1 (section 3.2) and Problem 2 (section 3.3), to design classifier for this problem. Classes are as follows.

1. $1^{st}$ class is VMPC cryptosystem of level 1. It is a stream cipher (section 4.2.4).

2. $2^{nd}$ class is Nonlinear Filter Generator of length 64 on $GF(4)$ based on LFSR. It is a stream cipher. It used 64 degree primitive polynomial (section 4.2.3).

3. $3^{rd}$ class is SPN(Substituation Permutation Network) based on 6 rounds with 8 blocks each of 8 bits and with a non linear S-box function. It is a block cipher (section 4.1.3).

4. $4^{th}$ class is Fiestel Cipher (32 bit + 32 bit; two blocks) with a simple Fiestel function simpler than DES (section 4.1.2).

In this problem, we applied one character based frequency analysis . Note that in this case, frequency pattern has 32 features. We experimented using following parameters.

1. We experimented with training ciphertext chunk of different size. We experimented with moderate size (approx. $100,000$ character chunk) and big size (approx. $1,600,000$) respectively.

2. We experimented with different size test ciphertext chunks. We experimented with small size (1000 charater), and moderate sizes(2000 character and 5000 character respectively). Please note that a character is made up of 5 bits.

3. We used key mixing. In other words, each training ciphertext chunk was obtained by concatenation of four (sub)chunks. Each (sub)chunk is obtained using different key. Note that it increased the size of training chunk by 4.

4. We applied one character based frequency analysis.

5. As mentioned in the introduction part, we used prototype method for classification.

6. We used chi-square distance function whose formula is as follows [1.1.2].

$$Chi - Square(F1, F2) = \sum_{i=1}^{n} \frac{(F1_i - F2_i)^2}{F1_i}$$

   where $F1_1, F1_2, ...$ are expected frequencies and $F2_1, ...$ are observed frequencies. And $n-1$ is the degree of freedom. $n$ is the problem of different 5 bit permutations, with repetitions, of $0's$ and $1's$.

We obtained following observations.

1. We applied frequency based randomness test and observed that all 4 cryptosystems had significantly different behavior. Note that we observed that class 1, class 2 and class 3 behaved nearly random whereas class 4 behaved as non random.

2. We pairwise compared frequency patterns corresponding to these 4 classes. We observed that relative Chi Square distances among Class 1, class 2 and class 3 were not significantly high. We also observed that frequencies corresponding to class 4

were well separated (easily distinguishable) from that of class 1, class 2 and class 3 respectively.

3. We observed that size of test ciphertext chunk affects the performance of classifier. We experimented with 1000 character length ciphertext chunk. We observed significant improvement in the result when we experimented with 5000 character length ciphertext chunk. Also, note that improvement in the result over the result by using 2000 character length ciphertext chunk was significant. But this jump was not very high.

4. We observed that size of training chunk affects the classifier performance. We experimented with 100,000 character chunk and found it insufficient to capture representative frequency pattern for class 4.

5. We observed that, on average, percentage of correct classification, p, by our classifier was around 30%. Note that it is higher than 0.25 (25% in percentage), which is probability of correct classification if a classifier randomly classifies.

6. We observed that probability of misclassification into each individual (incorrect) class were different from p. But they were not well separated. We observed that class 1 was not well separated to other classes. But rest classes were mutually well separated.

We interpreted last two observations as follows.

Classifier's probability of correct classification is not high. For the better classification, we can use the observation 4. For further improvement, we can use modified classifier design stretegy (section 2.1). Note that, ties will be resolved arbitrarily. Due to possibility of ties, we may obtain less than 100% correct classification.

## 3.5 Problem 4

Here we experimented with 3 different popular and seemingly very secure cryptosystems. These cryptosystems are same as in above problem (section 3.4) excluding Fiestel Cipher.

For the completeness sake, classes are as follows.

1. $1^{st}$ class is VMPC cryptosystem. It is a stream cipher (section 4.2.4).

2. $2^{nd}$ class is Nonlinear Filter Generator of length 64 on $GF(4)$ based on LFSR. It is a stream cipher. It used 64 degree primitive polynomial (section 4.2.3)

3. $3^{rd}$ class is SPN(Substituation Permutation Network) based on 6 rounds with 8 blocks each of 8 bits and with a non linear S-box function. It is a block cipher (section 4.1.3).

We experimented for two character based frequency analysis. Note that in this case, frequency pattern has $32 \times 32$ features. We experimented using following parameters.

1. We experimented with training ciphertext chunk of different size. We experimented with moderate size (approx. $900,000$ character chunk) and big size (approx. $6,000,000$) respectively.

2. We experimented with different size test ciphertext chunks. We experimented with small size (1000 character), and moderate sizes(2000 character and 5000 character respectively). Please note that a character is made up of 5 bits.

3. We used key mixing. In other words, each training ciphertext chunk was obtained by concatenation of four (sub)chunks. Each (sub)chunk is obtained using different key. Note that it increased the size of training chunk by 4.

4. we applied two character based frequency analysis.

5. As mentioned in the introduction part, we used prototype method for classification.

6. We used chi-square distance function whose formula is as follows [1.1.2].

$$Chi - Square(F1, F2) = \sum_{i=1}^{n} \frac{(F1_i - F2_i)^2}{F1_i}$$

where $F1_1, F1_2, ...$ are expected frequencies and $F2_1, ...$ are observed frequencies. And $n-1$ is the degree of freedom. $n$ is the problem of different 5 bit permutations, with repetitions, of $0's$ and $1's$.

We obtained following observations.

1. We applied frequency based randomness test and observed that all 3 cryptosystems had significantly different behavior. Note that we observed that all classes behaved fairly random.

2. We pairwise compared frequency patterns corresponding to these 3 classes. We observed that frequencies corresponding to these classes were well separated (easily distinguishable) among each other.

3. We observed that size of test ciphertext chunk affects the performance of classifier. We experimented with 1000 character length ciphertext chunk. We observed significant improvement in the result when we experimented with 5000 character length ciphertext chunk. Also, note that improvement in this result over the result by using 2000 character length ciphertext chunk was significant. But this jump was not very high.

4. We observed that percentage of correct classification of test chunks originating from class 1, class 2 , class 3 were about 50%. Note that it is significantly higher that 1/3 (approx. 33.34% in percentage), which is probability of correct classification if a classifier randomly classifies.

5. We observed that probability of misclassification into each individual (incorrect) class were significantly lesser than p. We also observed that they were very well separated.

We interpreted last two observations as follows.

We interpreted above observation as an indication of possibly good classifier. We can use modified classification strategy for getting almost 100% correct classification. Note that here we can use majority instead of calculating actual probability (section 2.1).

# Chapter 4

# Brief detail of used Ciphers

## 4.1 Block Ciphers

### 4.1.1 Vigenere Cipher

The Vigenere cipher is a method of encryption that uses a series of different Caesar ciphers based on the letters of a keyword. In a Caesar cipher, each letter of the alphabet is shifted along some number of places. For example, in a Caesar cipher of shift 3, A would become D, B would become E and so on. The Vigenere cipher consists of using several Caeser ciphers in sequence with different shift values.

For example, suppose the encipherer wishes to encrypt a plaintext:

$$ATTACKATDAWN$$

The encipherer chooses a keyword and repeats it until it matches the length of the plaintext, for example, the keyword, "$LEMON$".

$$LEMONLEMONLE$$

The first letter of the plaintext, A, is enciphered using the alphabet in row L, which is the first letter of the key. Similarly, for the second letter of the plaintext, the second letter of the key is used; the letter at row E and column T is X. The rest of the plaintext is enciphered in a similar fashion:

*Plaintext* : ATTACKATDAWN
*Key* : LEMONLEMONLE
*Ciphertext* : LXFOPVEFRNHR

### 4.1.2 Fiestel Cipher with a simple Fiestel function simpler than DES

Fiestel cipher is an iterated cipher. An iterated cipher is one that encrypts a plaintext block by a process that has several rounds. In each round, the same transformation function is applied to the data using a subkey. The set of subkeys are usually derived from the user provided secret key by a key schedule (refer [4]).

In the Fiestel cipher, the text being encrypted is split into two halves. The round function $f$ is applied to right half using a subkey and the output of $f$ is exclusive- ORed with the left half. The two halves are then swapped. Each round follows the same pattern except for the last round where there is no swap. [Figure 4.1] describes the Fiestel network.

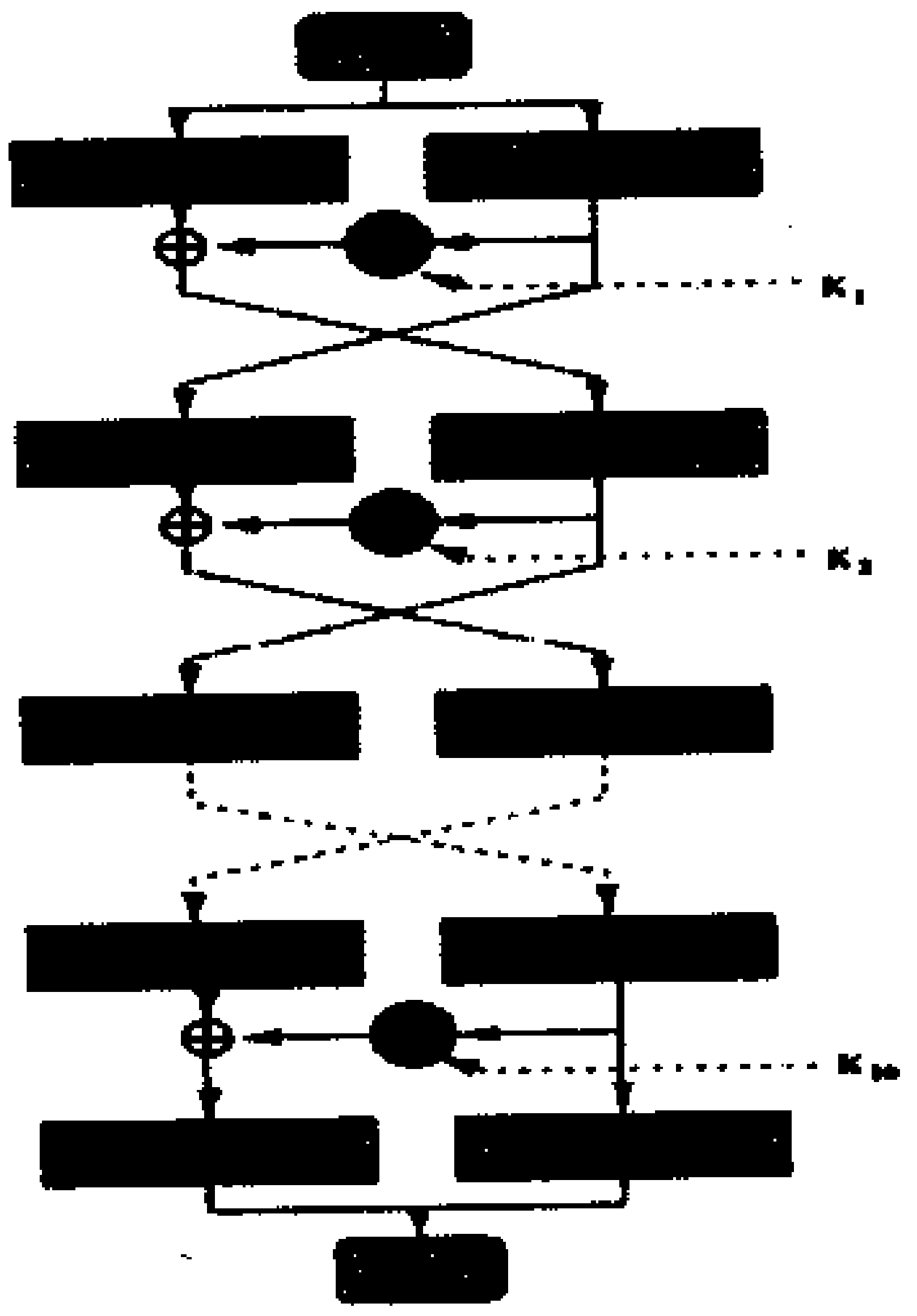

Figure 4.1: Fiestel Networks

DES is a typical 16 round 64 *bit* Fiestel Cipher. The sketch of its enciphering computation is as given in [figure 4.2]. Note that DES uses 8 $S$ boxes.

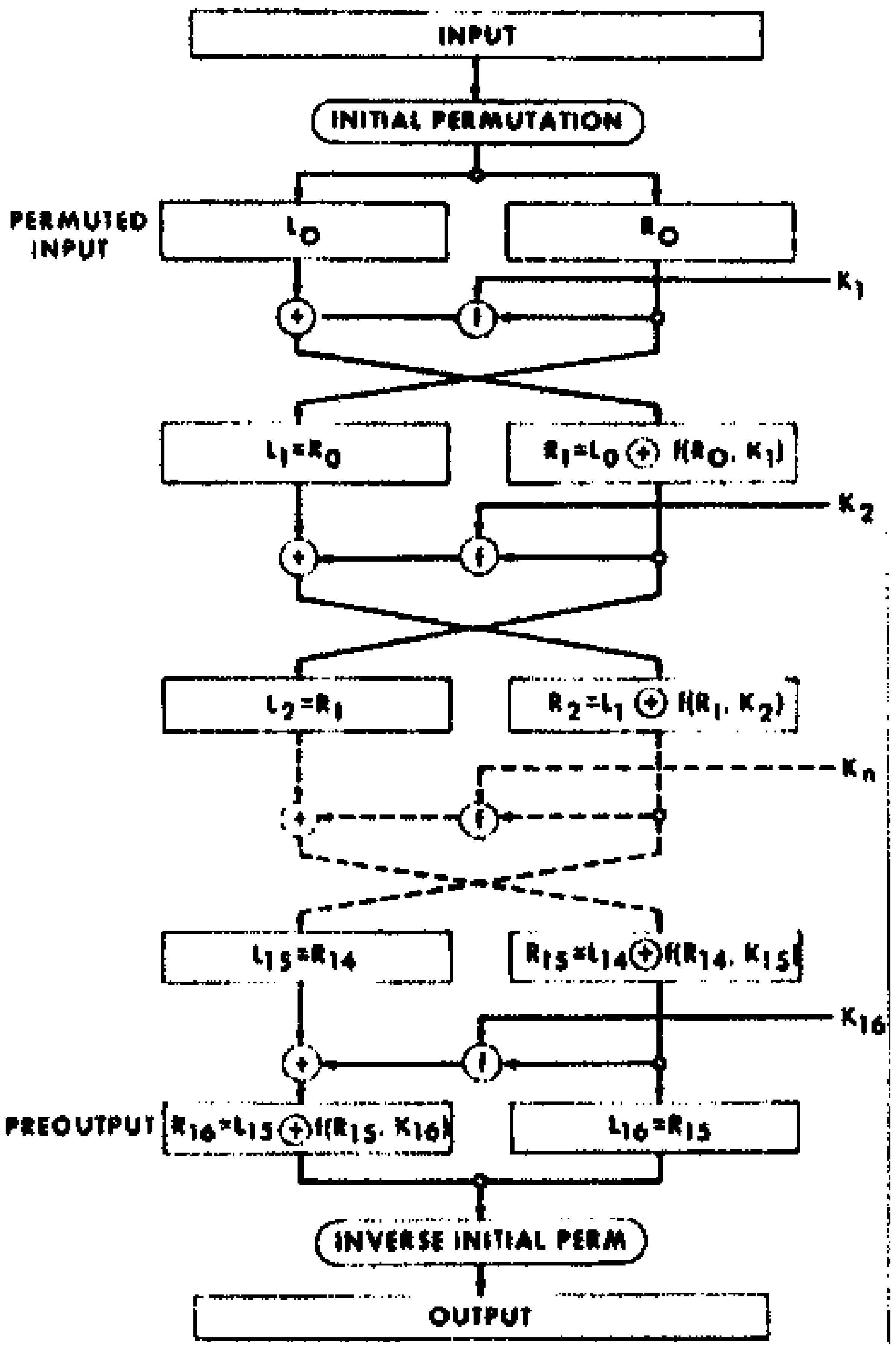

Figure 4.2: DES encryption diagram

Our Fiestel cipher is identical to DES except that we chose all 8 $S$ boxes same as 1st $S$ box.

### 4.1.3 SPN Cipher based on 8 bit boxes with six rounds and with a non linear S box function

SPN is an iterated cipher. An iterated cipher is one that encrypts a plaintext block by a process that has several rounds. In each round, the same transformation function is applied to the data using a subkey. The set of subkeys are usually derived from the user provided secret key by a key schedule.

In this case, we used substitution function (popularly known as S box). We used 64 bit block size with 6 rounds of encryption. Non linear $S$ box was defined as follows.

$$f(X_i^1, X_i^2, ..X_i^8) = \left\{ \begin{array}{c} X_i^3 + X_i^5.X_i^7 \\ X_i^1.X_i^3 + X_i^7 \\ X_i^1.X_i^3 + X_i^5.X_i^7 \\ \bar{X}_i^1.X_i^3 + X_i^3.\bar{X}_i^7 \\ \bar{X}_i^1.X_i^5 + X_i^1.X_i^7 + \bar{X}_i^7 \\ \bar{X}_i^1.X_i^3 + \bar{X}_i^5.\bar{X}_i^7 + X_i^7 \\ X_i^1.X_i^3 + \bar{X}_i^5 + X_i^1.\bar{X}_i^7 \\ \bar{X}_i^1.\bar{X}_i^5 + \bar{X}_i^5.X_i^7 + \bar{X}_i^3.\bar{X}_i^7 \end{array} \right\}$$

An SPN network with 3 rounds and 4 $S$ boxes is shown in [figure 4.3].

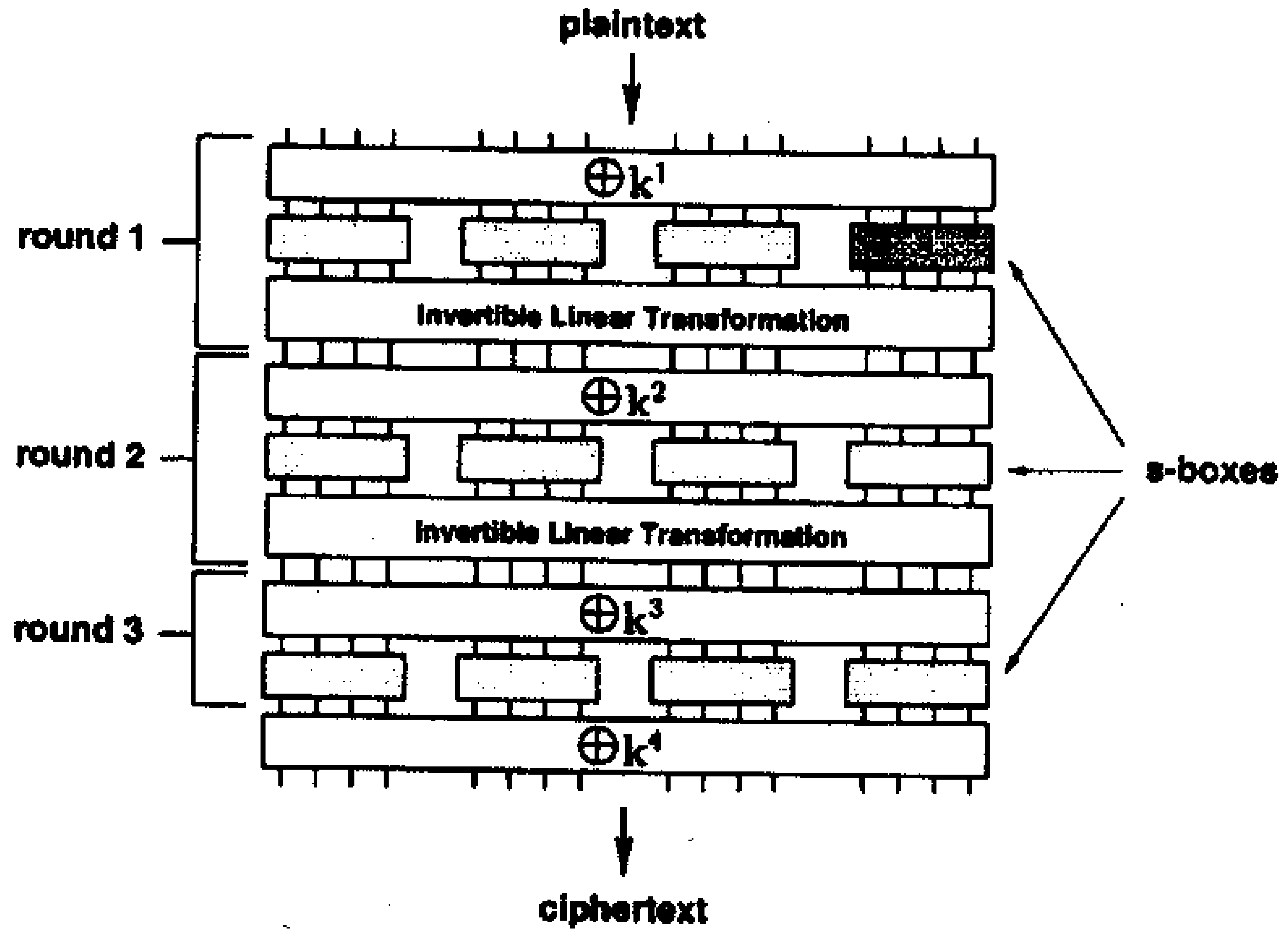
plaintext
round 1
round 2
round 3
⊕k¹
Invertible Linear Transformation
⊕k²
Invertible Linear Transformation
⊕k³
⊕k⁴
s-boxes
ciphertext

Figure 4.3: SPN Network with 3 rounds and 4 S boxes

## 4.2 Stream Ciphers

### 4.2.1 LFSR based linear Stream Cipher on GF(2)

Binary stream ciphers are often constructed using linear feedback shift registers (LFSR)s because they can be easily implemented in hardware and can be readily analyzed.

An LFSR is constructed using primitive polynomial. Note that a primitive polynomial of degree $m$ over GF(2) is used to generate extension field $GF(2^m)$. $m$ degree primitive polynomial is used to create $m$ bit LFSR. The values in $m$ bit LFSR determines its state. Note that $m$ bit LFSR can have $2^m - 1$ different states (excluding all $0's$). A primitive polynomial of degree $m$ guarantees that period of corresponding LFSR is $2^m - 1$ if we start with non-zero initial state. Note that next output bit of LFSR is determined by current state of the LFSR. In other words, it depends on previous $m$ output bits. Note that, LFSR is used to generate pseudo-random bit sequence (refer [4]).

Encryption is done by simply XORing plaintext bits with corresponding keystream bit. Decryption is done in the way identical to Encryption process. [Figure 4.4] describes the LFSR key stream generator.

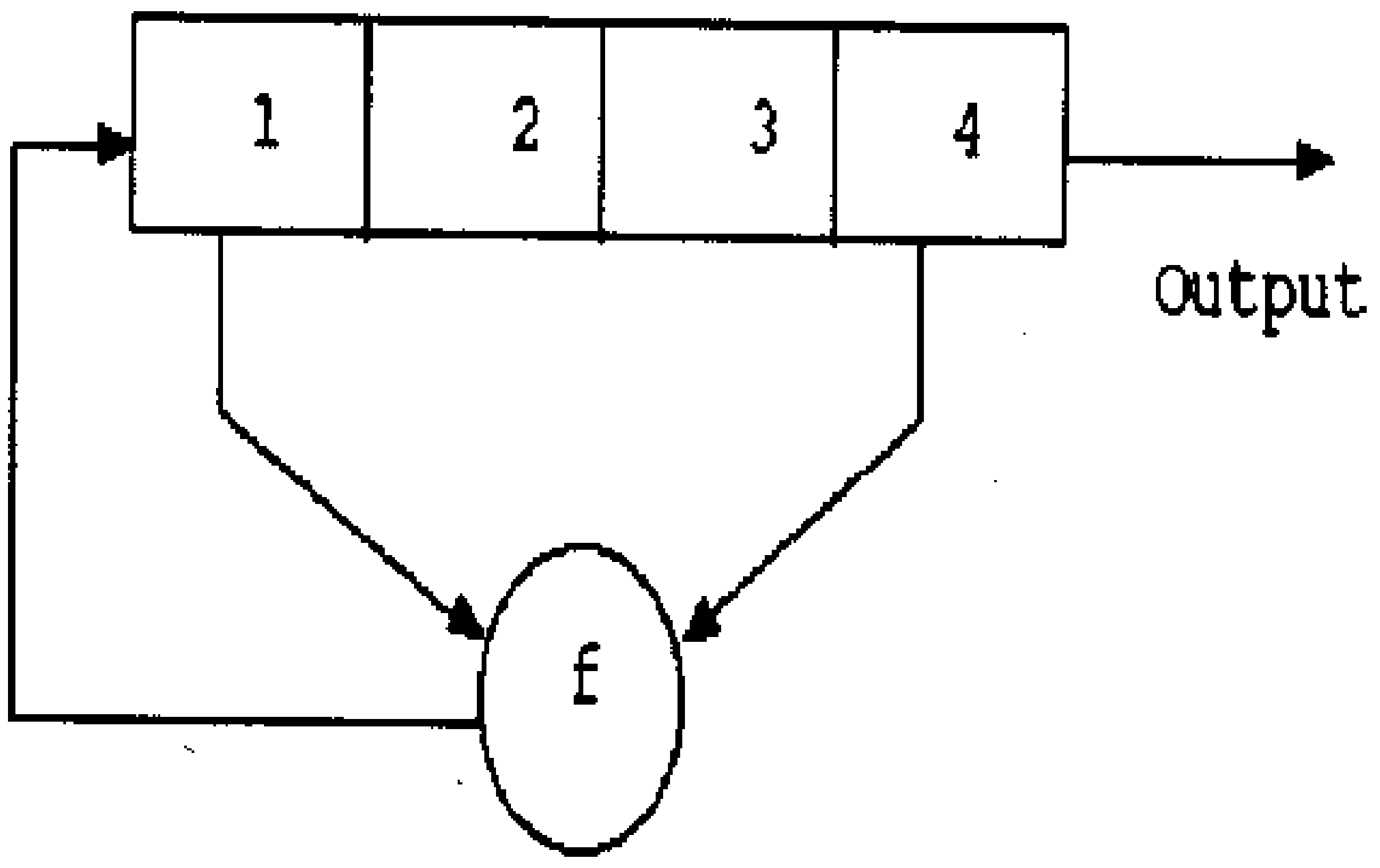

Figure 4.4: LFSR key stream generator

### 4.2.2 non LFSR based Stream Cipher

We know that Stream Cipher needs key stream generator. For this purpose, a recurrence relation is used. We used following recurrence relation.

$$X_m = (aX_{m-1} + b)\%N \text{ where } gcd(a, N) = 1 \text{ and } X_0 \; is \; given$$

For example, let
$N = 256$
$a = 23$
$b = 34$

In this case, this recurrence will output a byte at a time. This byte is converted to bits. Encryption is done by simply XORing plaintext bits with corresponding keystream bit. Decryption is done in the way identical to Encryption process.

### 4.2.3 Nonlinear Filter Generator of length 64 on GF(4) based on LFSR

We used the process based on LFSR algorithms on GF(4), but the output key stream was generated by a non linear filter generator $g$ based on the register sequence $\{s_i\}$. Thus given the current sequence $\{s_{63}, s_{62}, ..s_0\}$, the output at that stage was a non linear function of $\{s_i\}s$, rather than the output stream of the LFSR system. Note that, the bits have possible values of $0, 1, \alpha \; and \; (1 + \alpha)$.

We chose non linear filter generator function as follows.

$$g = s_0.\bar{s_7} \bigoplus s_7.s_{13} \bigoplus s_{37}.s_{59}$$

Its 64 degree primitive polynomial is as follows.

$$\begin{aligned} f^{pp4,64}(x) &= x^0 + \alpha x^1 + (1+\alpha)x^2 + \alpha x^4 + x^5 + \alpha x^6 + \alpha x^7 + x^8 + \alpha x^9 \\ &+ x^{11} + (1+\alpha)x^{12} + \alpha x^{13} + (1+\alpha)x^{14} + \alpha x^{15} + \alpha x^{16} + \alpha x^{17} \end{aligned}$$

$$
\begin{aligned}
&+\; x^{19} + x^{20} + \alpha x^{21} + x^{22} + \alpha x^{23} + (1+\alpha)x^{26} + x^{27} + (1+\alpha)x^{28} \\
&+\; x^{30} + (1+\alpha)x^{31} + (1+\alpha)x^{32} + \alpha x^{33} + (1+\alpha)x^{34} + x^{35} \\
&+\; \alpha x^{36} + \alpha x^{37} + x^{40} + x^{41} + \alpha x^{43} + x^{44} + x^{45} + (1+\alpha)x^{46} \\
&+\; x^{47} + \alpha x^{48} + (1+\alpha)x^{50} + \alpha x^{51} + x^{52} + \alpha x^{53} + (1+\alpha)x^{54} \\
&+\; x^{55} + x^{56} + \alpha x^{57} + +x^{60} + x^{62} + \alpha x^{63} + x^{64}
\end{aligned}
\tag{4.1}
$$

### 4.2.4 VMPC cipher

VMPC applies key scheduling algorithm to transforms a cryptographic key and (optionally) an initialization vector into a 256 - element permutation $P$ and initializes variable $s$. The VMPC Key Scheduling Algorithm(KSA) transforms a cryptographic key. The main algorithm generates a stream of 8-bit values. This cipher is very new. Its main algorithm is as follows (refer [6]).

Let,  
P: 256-byte table(array) storing a permutation initialized by VMPC KSA  
s: 8-bit variable initalized by the VMPC KSA  
n: 8-bit variable  
L: desired length of the keystream in bytes

**VMPC Stream Cipher**

```
1. $n = 0$
2. $Repeatsteps\ 3 - 6\ L\ times:$
   3. $s = P[(s + P[n])\ modulo\ 256]$
   4. $OutputP[(P[P[s]] + 1)\ modulo\ 256]$
   5. $Temp = P[n]$
      $P[n] = P[s]$
      $P[s] = Temp$
6. $n = (n + 1) modulo\ 256$
```

# Chapter 5

# Experimental Results

## 5.1 Problem 1

Following are the classes.

**Class 1** $1^{st}$ class is Vigenere cryptosystem. It is a block cipher

**Class 2** $2^{nd}$ is an stream cipher whose key generating recurrence relation is as follows.

$$X_m = (aX_{m-1} + b)\%N \quad \text{where} \quad gcd(a, N) = 1 \quad \text{and} \quad X_0 \quad \text{is} \quad \text{given}$$

### 5.1.1 Frequency based randomness test

Size of Ciphertext chunk = 1000 characters.

Here, we experimented using one character frequency test. Sample Chi square values are shown in [table 5.1]. Note that theoretically, the probability of the occurrence of a 5 bit character is 1/32.

## 5.2 Problem 2

Following are the classes.

| Vigenere | Stream |
|---|---|
| 213.504 | 34.688 |
| 221.120 | 36.8 |
| 180.744 | 58.56 |
| 1011.328 | 106.432 |
| 1121.472 | 80.960 |
| 2393.856 | 175.744 |
| 1880.640 | 177.792 |
| 2601.024 | 193.344 |
| 2507.52 | 173.44 |
| 2173.632 | 174.528 |

Table 5.1: Frequency based randomness test

**Class 1** $1^{st}$ class is a stream cipher whose key generating recurrence relation is as follows.

$$X_m = (aX_{m-1} + b)\%N \quad \text{where} \quad gcd(a, N) = 1 \quad \text{and} \quad X_0 \quad \text{is} \quad \text{given}$$

**Class 2** $2^{nd}$ class is a stream cipher using LFSR (Linear Feedback Shift Register) as random key generator. Here we have used $GF(2)$ as base field.

### 5.2.1 Frequency based randomness test

Size of Ciphertext chunk = 70000 characters.

Here, we experimented using one character frequency test. Cryptosystems had following Chi - Square values.

**frequency based randomness test** We applied test against theoretical random frequency pattern. Note that, theoretically, the probability of a 5 bit character is 1/32. Following were Chi Square distances.

Class 1: 8461.339

Class 2 with primitive polynomial of degree 8 : 8023.427

Class 2 with primitive polynomial of degree 16 : 70.11

Class 2 with primitive polynomial of degree 24 : 21.89

**Relative distance between classes** We applied Chi Square distance function. Its tables are as follows [table 5.2] [table 5.3].

| Chi-Square | Class 1 |
|---|---|
| class 2 with 16 degree polynomial | 14719.55 |
| class 2 with 16 degree polynomial | 7721.60 |
| class 2 with 24 degree polynomial | 7627.60 |

Table 5.2: Chi square distance: row represents class 1 and column represents class 2

| Chi-Square | 8 degree polynomial | 16 degree polynomial | 24 degree polynomial |
|---|---|---|---|
| class 1 | 17166.41 | 8506.70 | 8421.76 |

Table 5.3: Chi square distance: row represents class 1 and column represents class 2

### 5.2.2 Test data using two keys of LFSR

In the next step, we experimented to analyze behavior of key in the classification process. For this purpose, we slightly modified our above experiment to include one more class. In this $3rd$ class, we used different initial key but same LFSR having same primitive polynomial.

Size of Ciphertext chunk = 70000 characters.

Here, we experimented using one character frequency test. Note that theoretically, the probability of the occurrence of a 5 bit character is 1/32.

Cryptosystems had following Chi - Square values.

**frequency based randomness test** We applied test against theoretical random frequency pattern. Not that, theoretically, the probability of a 5 bit character is 1/32. Following were Chi Square values.

Class 1: 8461.339

Class 2 with primitive polynomial of degree 8 : 8023.427

Class 2 with primitive polynomial of degree 16 : 70.11

Class 2 with primitive polynomial of degree 24 : 21.89

Class 3 with primitive polynomial of degree 8 : 14147.93

Class 3 with primitive polynomial of degree 16 : 71.73

Class 3 with primitive polynomial of degree 24 : 21.33

**Relative distance between classes** We applied Chi Square distance function. Its tables are as follows [table 5.4] [table 5.5] [table 5.6].

| Chi-Square | class2 | class3 |
|---|---|---|
| class 1 for 8 degree polynomial case | 17166.42 | 33424.21 |
| class 1 for 16 degree polynomial case | 8506.70 | 7782.93 |
| class 1 for 24 degree polynomial case | 8421.76 | 8516.36 |

Table 5.4: Chi square distance: row represents $1st$ class and column represents $2nd$ class

| Chi-Square | class1 | class3 |
|---|---|---|
| class 2 with 8 degree polynomial | 14719.55 | 33935.69 |
| class 2 with 16 degree polynomial | 7721.60 | 139.96 |
| class 2 with 24 degree polynomial | 7627.60 | 42.68 |

Table 5.5: Chi square distance: row represents $1st$ class and column represents $2nd$ class

| Chi-Square | class1 | class2 |
|---|---|---|
| class 3 with 8 degree polynomial | 24552.94 | 34029.33 |
| class 3 with 16 degree polynomial | 7131.80 | 140.11 |
| class 3 with 24 degree polynomial | 7651.67 | 42.88 |

Table 5.6: Chi square distance: row represents $1st$ class and column represents $2nd$ class

## 5.3 Problem 3

Following are the classes.

**Class-1** $1^{st}$ class is VMPC cryptosystem of level 1. It is a stream cipher.

**Class-2** $2^{nd}$ class is Nonlinear Filter Generator of length 64 on $GF(4)$ based on LFSR. It is a stream cipher. It used 64 degree primitive polynomial (section 4.2.3)

**Class-3** $3^{rd}$ class is SPN (Substituation Permutation Network) based on 6 rounds with 8 blocks each of 8 bits and with a non linear S-box function. It is a block cipher (section 4.1.3).

**Class-4** $4^{th}$ class is Fiestel Cipher (32 bit + 32 bit; two blocks) with a simple Fiestel function simpler than DES (section 4.1.2).

### 5.3.1 Step 1 of Learning Phase

Size of training ciphertext chunk = 1792000 characters

Representative frequency patterns had following Chi - Square values.

**frequency based randomness test** We applied test against theoretical random frequency pattern. Note that, theoretically, the probability of a 5 bit character is 1/32. Following were Chi Square values.

| | |
|---|---|
| Class 1: | 74.4823 |
| Class 2: | 36.7688 |
| Class 3: | 69.421 |
| Class 4: | 2743.3 |

**Relative distance between classes** We applied Chi Square distance function. Its table is as follows [table 5.7].

| Chi-Square | class1 | class 2 | class 3 | class 4 |
|---|---|---|---|---|
| class 1 | N.A. | 109.867 | 140.347 | 2925.93 |
| class 2 | 109.587 | N.A. | 97.4145 | 2551.53 |
| class 3 | 140.998 | 97.4289 | N.A. | 2524.96 |
| class 4 | 2984.89 | 2584.22 | 2573.07 | N.A. |

Table 5.7: Chi square distance: rows represents 1*st* class and column represents 2*nd* class

### 5.3.2 Step 2 of Learning Phase

In this step, we tested using different sizes of ciphertext chunks to predict probability of correct classification. We also analyzed the relation between correctly classified and misclassified data. Corresponding tables are [ table 5.8], [table 5.9] and [table 5.10].

Here Each Row represents percentage of sample from class indicated by row number in class indicated by column number. Last row represents the correct classification percentage.

**Experiment 1** Size of ciphertext chunk = 5000 characters
Please see [table 5.8].

| row:actual class col:predicted class | class 1 | class 2 | class 3 | class 4 | correct classification |
|---|---|---|---|---|---|
| class 1 | 35.56 | 26.66 | 26.66 | 11.12 | 35.56 |
| class 2 | 34 | 33 | 26 | 7 | 33 |
| class 3 | 29 | 30 | 29 | 12 | 29 |
| class 4 | 32 | 19 | 15 | 34 | 34 |

Table 5.8: Distribution of class prediction (in percentage) by classifier

**Experiment 2** Size of ciphertext chunk = 2000 characters
Please see [table 5.9].

**Experiment 3** Size of ciphertext chunk = 1000 characters
Please see [table 5.10].

| row:actual class col:predicted class | class 1 | class 2 | class 3 | class 4 | correct classification |
|---|---|---|---|---|---|
| class 1 | 40 | 18.5 | 22.5 | 19 | 40 |
| class 2 | 29 | 29.5 | 27 | 14.5 | 29.5 |
| class 3 | 29 | 22.5 | 28 | 20.5 | 28 |
| class 4 | 27 | 25 | 10.5 | 37.5 | 37.5 |

Table 5.9: Distribution of class prediction (in percentage) by classifier

| row:actual class col:predicted class | class 1 | class 2 | class 3 | class 4 | correct classification |
|---|---|---|---|---|---|
| class 1 | 29 | 23.75 | 24.75 | 22.5 | 29 |
| class 2 | 30.25 | 21.75 | 25 | 23 | 21.75 |
| class 3 | 27.5 | 20 | 22.75 | 39.75 | 22.75 |
| class 4 | 35 | 16.75 | 13.25 | 35 | 35 |

Table 5.10: Distribution of class prediction (in percentage) by classifier

## 5.4 Problem 4

Following are the classes.

**Class-1** 1st class is VMPC cryptosystem of level 1. It is a stream cipher.

**Class-2** 2nd class is Nonlinear Filter Generator of length 64 on $GF(4)$ based on LFSR. It is a stream cipher. It used 64 degree primitive polynomial (section 4.2.3)

**Class-3** 3rd class is SPN(Substituation Permutation Network) based on 6 rounds with 8 blocks each of 8 bits and with a non linear S-box function. It is a block cipher.

### 5.4.1 Step 1 of Learning Phase

Size of training ciphertext chunk = 6, 144, 000 characters

Representative frequency patterns have following Chi - Square values.

**frequency based randomness test** We applied test against theoratical random frequency pattern. Not that, theoratically, the probability of occurance a two 5 bit character is $1/(32 \times 32)$.

Class 1: 1279.24

Class 2: 1671.04

Class 3: 1279.24

**Relative distance between classes** We applied Chi Square distance function. Its table is as follows [table 5.11].

| Chi-Square | class1 | class 2 | class 3 |
|---|---|---|---|
| class 1 | N.A. | 4129.24 | 6105.28 |
| class 2 | 4197.58 | N.A. | 3783.62 |
| class 3 | 6113.72 | 3795.41 | N.A. |

Table 5.11: Chi square distance: rows represents $1st$ class and column represents $2nd$ class

### 5.4.2 Step 2 of Learning Phase

In this step, we tested using ciphertext chunks to predict probability of correct classification [table 5.12]. We also analyzed the relation between correctly classified and misclassified data.

Size of ciphertext chunk = 5000 characters
Please see [table 5.12].

| row:actual class col:predicted class | class 1 | class 2 | class 3 | correct classification |
|---|---|---|---|---|
| class 1 | 54.44 | 16.67 | 28.89 | 54.44 |
| class 2 | 25 | 53 | 22 | 53 |
| class 3 | 21 | 24 | 55 | 55 |

Table 5.12: Distribution of class prediction (in percentage) by classifier

Here Each Row represents percentage of sample from class indicated by row number in class indicated by column number. Last row represents the correct classification percentage.

# Chapter 6

# Final Remarks

In this report, we presented our currently practised approach to handle previously described problem. We successfully modelled it as a classification problem. We also identified and fixed different parameters. We obtained encouraging results. Since, there is no existing known solution in literature, it may act as a guideline for any future attempt.

Although, it may be enough to provide solution according to given set of classes (Cryptosystems), we can test the generalization capability of our frequency based learning scheme. This can be tested with different set of classes and varying number of classes.

In this report, we tackled the problem to distinguish originating cryptosystem. Our next problem can be defined as follows.

"Given a ciphertext with unknown origin, identify whether the algorithm, which created it, is block cipher or stream cipher"

In this report, we used Pearson's Chi Square distance function as our similarity measure. In future, other distance measures may be used. We used prototype method in our classification approach. In literature related to pattern classification, there are many other good classification strategies, for example, K-NN classification strategy. Similarily, we can fix other parameters accordingly.

In this report, we assumed 32 different characters in the plaintext chunk. In the actual practice, some other symbols also occur in the English plaintext chunk. We need to confirm the performance of our classification scheme under this relaxation.

# Bibliography

[1] Trivedi, Kishor S. : *Probability & Statistics With Reliability, Queuing, And Computer Science Applications*, Prentice Hall of India, New Delhi, 1998

[2] Mendenhall, W., Wackerly, D. D., Scheaffer, R. L. : *Mathematical Statistics With Applications*, *4th* Edition, PWS-Kent Publishing Company, Boston

[3] MacWilliams, F. J., Sloane, N. J. A., :*Theory of Error Correecting Codes*, North Holland Mathematical library, 2003

[4] Stinson, Douglas R. : *Cryptography, Theory and Practice*, CRC Press, Boca Raton, 1995

[5] Scheneir B S : *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd Edition, John Wiley and Sons, New York, 1995

[6] Zoltak, B. (February 2004): "VMPC One-way Function and Stream Cipher", 11th International Workshop, FSE 2004 Delhi, India, Febraury 2004. Published in *Lecture notes in Computer Science with title "Fast Software Encryption* " by Springer. pp 200-225.

[7] Duda, R. O., Hart, D. G., Strok, D. G.: *Pattern Classification* , 2nd Edition, Wiley, New York, 2000

[8] Tou, J. T., Gonzalez, R. C.: *Pattern Recognition Principles* , Addison-Wesley, London, 1974