A

TH-43042

Some Studies on Information Representation and Transmission

Thesis
submitted by

Dhruba Kumar Bhattacharyya

under the guidance of

Prof. S Nandi and Prof. M C Bora

for the award of the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science
Tezpur University
Tezpur - 784 001, INDIA
August, 1998

Certificate

This is to certify that the thesis entitled " **Some Studies on Information Representation and Transmission**", submitted by **Dhruba Kumar Bhattacharyya**, a research scholar in the *Department of Computer Science, Tezpur University, Tezpur*, for the award of the degree of **Doctor of Philosophy**, is a record of an original research work carried out by him under our supervision and guidance. The thesis has fulfilled all requirements and in our opinion has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

[illegible] 14.8.98

Madhab Chandra Bora, Professor,
Business Administration Department,
Tezpur University,
Tezpur - 784 001, Assam, India

Dated : 14 August, 1998

[illegible]

Sukumar Nandi, Associate Professor,
Computer Sc. & Engg. Department,
Indian Institute of Technology,
Guwahati - 781 001, Assam, India

Dated : 14 August, 1998

Acknowledgement

It is a pleasure to express my deep gratitude towards both the supervisors Prof. M. C. Bora and Prof. S. Nandi, who were the driving force behind this work. I am highly indebted to Prof. S Nandi, who first introduced me to these problems and motivated to work, on which I am grateful to him. His unfaltering enthusiasm and utmost attention to my doubts was a constant source of inspiration. Prof. Bora's constant encouragement in every step of my work induced research abilities in me. I consider myself fortunate to have been associated with both of them.

I am specially thankful to Prof. K. M. Pathak, the Hon'bl Vice Chancellor of this University for his inspiration and valuable suggestions throughout the work. I am thankful to Prof. Indraneel Chakravorty, Department of Electronics, I.I.T., Guwahati, who carefully reviewed the thesis with his deep insights in the preliminary stage. I am also grateful to Prof. A K. Borkakoti, Department of Mathematical Sc., for his guidance in initiating research.

Prof. S N N Pandit, Visiting Fellow, Department of Computer Science, was more than a guide to me and was always ready to help me in all problems. He spent his valuable time with me in finding the cryptographic solutions. Among the colleagues, Prof D. K Saikia, Dr Rajib Das and Smriti Sinha, with whom I regularly shared my thoughts and ideas, and I received critical comments from them. I specially acknowledge Shyamanta, Sarat and Bubul for their contribution in various stages of my research work. I am grateful to Dr B. Dubey, Department of Math. Sc., for his encouragement and cooperation.

I would also like to thank Prof. L. N. Bhuyan of Texas A & M University, USA, Dr. Prathima Agarwalla of Bell Lucent Laboratory, USA, Prof. A. K. Somani, of Iowa University, USA and many other reviewers, for their critical comments and helpful suggestions.

Finally, I must appreciate the sacrifice of my wife Dr. Chayanika, for her support and encouragement, which made the work possible. I am also thankful to my parents and other family members for cheerfully bearing with my whims and for letting me make my own decision.

[illegible]

(D. K. Bhattacharyya)
16.8.98

Abstract

The present thesis embodies a study of 3-neighbourhood additive Cellular Automata (CA) behaviour and its applications in *authentication* and *authenticated key exchange*. Earlier works in the field of CA have been reviewed. Subsequently, characterization of Extended Neighbourhood CA (ENCA) (not explored by previous researchers) and its application in data encryption are presented. Various error control coding techniques for t-Error Correcting (EC)/ d-Error Detecting (ED) and All Unidirectional Error Detecting (AUED) codes as well as Single error correcting (SEC)/Double error detecting (DED)/AUED codes have been reviewed. Two novel techniques, one for t-EC/d-ED/AUED codes and the other for SEC/DED/AUED codes respectively are presented. For efficient handling of multimedia data as well as for conceptual representation of any complex hypermedia domain, an existing data model, referred as Relationship Management Data Model (RMDM) has been analysed and subsequently modified. The modified RMDM has been implemented and the results are embodied in the present thesis. Further, it also introduces a user-friendly, three-layered image database system (IDS) architecture for handling complex queries over large repository of images. To assist in better understanding of the complex queries, the thesis also presents a visual semantic data model.

Keywords : Cellular Automata(CA), Extended Neighbourhood CA (ENCA), Authentication, Data encryption, Password-only Authenticated Key Exchange (PAKE), All Unidirectional Error Detection (AUED), Hypermedia, Relationship Management Data Model (RMDM), Image Database System (IDS), Visual Semantic Model (VSM).

Contents

List of Figures

List of Tables

Abbreviations Used

CA	Cellular Automata
MACA	Multiple Attractor CA
NBCA	Null Boundary CA
PBCA	Periodic Boundary CA
ENCA	Extended Neighbourhood CA
PCA	Programmable CA
CHA	Cellular based Hash Function
XNOR	Exclusive NOR
XOR	Exclusive OR
CPAKE	Cellular based Password-only Authenticated Key Exchange
SEC	Single Error Correcting
DED	Double Error Correcting
AUED	All Unidirectional Error Detecting
ROM	Random Only Memory
VLSI	Very Large Scale Integrated Circuit
RMDM	Relationship Management Data Model
IDS	Image Database System
VSM	Visual Semantic Data Model

Chapter 1

Introduction

In the last five decades of its existence, the modern computer has come a long way from doing just the basic number crunching. In terms of computing power, small personal computers easily out-perform large main-frame machines of yester-years. In terms of the spread of functionality, we notice an ever increasing use of computer systems in data management. Advances in communication technologies have further fueled the use of information technology in distributed environment, whether it is for managing on-line transaction across a large distributed network, or in maintaining large databases often containing unstructured information including multi-media data. With this ever-increasing growth of computer and communication technologies, the need for efficient and secure representation of data and its reliable transmission have become the basic necessities.

Secure representation of data for preventing the un-authorised interception is a legitimate need of not only the government institutions including the defense services in particular, but also of the business sector and private individuals. Authentication, data security, digital signature and file security are the various elements that need to be examined for protecting data in computer and distributed database systems from unauthorized disclosure and modification To solve the security problems in any distributed application, basically four kinds of mechanisms are used, such as *privacy*, *authentication*, *integrity* and *scalability*. Privacy includes the desire to keep documents and communication secret, as well as to hide the very existence of certain kinds of information and to protect the identities of the parties communicating with each other. Authentication and integrity refer to the need to confirm the identity of user, the authenticity of messages and the integrity of messages or connections. Scalability mechanisms such as distribution centers and digital certificates, add another crucial aspect to the success of any such applications, because they help in creating systems that involve millions of users, transactions and documents. Among these four measures, privacy and authentication problems are serious, and special care is needed to design appropriate schemes for data security and for user as well as message authentication. Significant research work has been carried out over the past decade to investigate these two problems and several enciphering and authentication schemes [Denni82, Reupl86, Welsh88, Sebry89, Schnr96, Palch97] have been evolved. However, a study of these schemes has revealed that - most of the existing enciphering schemes are impractical, mainly due to, either the use of a large key or the need of a ciphertext much larger than the plain-text. It appears that the time and resource complexities of the existing authentication schemes are also significantly large [Schnr96], which make the hardware implementation more complicated. From the analytical study of additive Cellular Automata (CA) it is observed that - the simple, regular structure-based features of CA can be utilized for designing secure and hardware efficient schemes for authentication and data encryption.

With the advancement in the convergence of computer and communication technologies, the demand for efficient and reliable digital data transmission and storage systems has also been increasing. This demand has been accelerated by the emergence of large-scale, high-speed data networks for exchange, processing and storage of digital information in the military, government and private spheres. In this background, a major concern of a researcher is to control the errors so that reliable reproduction of data can be obtained. Basically, the computer systems encounter three classes of errors: *symmetric*, *asymmetric* and *unidirectional*. Among the various types of causes of errors, such as : transient, intermittent and permanent, it is seen that the transient errors are responsible for mostly limited number of symmetric or multiple unidirectional errors. The intermittent faults cause limited number of errors, but the permanent faults may cause both symmetric as well as the unidirectional errors. In this context several error control codes have been proposed [Bose82, Lin83, Wickr96, Bose85, Nikol86, Lin88, Tao88, Blaum89, Rao89, Boinc90, Kundu90, Montg90, Nikol91, Bruck92, Basam94]. However, still there is a growing demand for construction of some simplified, storage-efficient, easily implementable symmetric error(s) correcting codes, which can also simultaneously detect all Unidirectional errors.

Now-a-days, every new step towards the advancement in technology centers around new ways to store, retrieve and transmit various types of information. In the past five years there has been growing interest in hypermedia design approaches [Garzo93a, Isakw95, Lange94]. There are many different problems the hypermedia designer has to deal with, since the combination of navigation through an intricate information space with the unstructured and dynamic nature of multimedia data poses new and complex problems that must be solved in a systematic and modular way. Each design activity in a design methodology should address difficult concerns at the proper stage and at the proper level of abstraction. To model the complex navigation patterns of any hypermedia application domain in a conceptual manner, the modelling tool should be rich enough from semantical point of view. From the current review, it is observed that RMDM [Isakw95] fulfills most of the criteria necessary for any complex, regular structure-based hypermedia applications.

Visual information is known to be a persuasive, intuitive and most expressive communication medium for humans. Its usefulness and usability have been increasing widely with the advent of systems handling multimedia information in a variety of application areas including natural sciences, military, industry and medicine. With the emerging technologies in high-speed communication networks and the presentation techniques, in particular, there has been a great demand for high-quality information processing that requires a suitable Image Database Systems (IDS) architecture with efficient image indexing and retrieval mechanisms Because of the large volume and difficulties in analyzing the contents of images, regardless of whether they are still images or video, however, it remains to be an investigating problem to effectively store and retrieve them based on their contents

In the background of the above scenario, the present thesis has utilized the simple and regular structure of CAs for the design of one-way hash function, data encryption and password-based authenticated key exchange schemes. The thesis also includes construction of two simple and lesser hardware-based error correction and detection schemes Further, it also discusses two key issues and their solutions relating to multimedia data modelling and image database processing The next section briefly reports the major contributions of the previous researchers in these fields Subsequently, it presents the schemes designed and developed to provide better solutions to some of the well-known information representation and transmission bottlenecks

1.1 Information Representation and Transmission

A brief review of the earlier studies on various issues relating to information representation and transmission is reported in this subsection. Some of the key issues relevant to CA based applications, error correction/detection, hypermedia application design, image database processing and retrieval are highlighted.

1.1.1 CA and Its Applications

J. von Neumann [Neuma66] first initiated the study of Cellular Automata (CA) and he framed CA as a cellular space with self-reproducing configurations involving 5-neighborhood cells, each having 29 states. Subsequently, theory and application of CA have been developed in diverse areas such as pattern recognition, modelling, biological and physical systems, parallel computation etc. Study of regular structure of CA has drawn considerable attention in recent years [Alady76, Toffo77, Wolfr83, Wolfr84, Wolfr85a, Wolfr85b]. However, it is not possible to characterize null boundary and hybrid CAs with polynomial algebraic tools. Recently, a more versatile tool based on matrix algebra [Das90b, Barde90, Das91] has been proposed to characterize 3-neighborhood additive CAs. Different applications of additive CAs have been investigated [Palch97, Nandi94a, Misra92, Pries86, Serra90, Das93] Major applications of 3-neighborhood additive CA proposed so far can be categorized as : (i) pseudo-random, pseudo-exhaustive and deterministic pattern generation [Palch97, Nandi94a, Das93, Chowd92, Barde90], (ii) synthesis of easily testable Finite State Machine (FSM) design [Palch97, Nandi94a, Chowd92, Misra92], (iii) Data encryption [Palch97, Nandi94a], (iv) Error Correcting Codes [Palch97, Chowd92], (v) Hashing [Palch97, Chowd92] and (vi) Signature analysis [Palch97, Misra92, Serra90, Das90a, Das90b]. The analytical study of the simple and regular structure of CA has motivated us to further investigate the CA behaviour and development of its applications in the various fields of cryptography. Various developments in three major areas of cryptography proposed so far are as follows

1. Message Authentication using one-way hash function [Denni82, Welsh88, Schry89, Schni96, Merkl90, Rivst92, NIST93, Rhee94, Simon92, Menez97, Zheng93, Vande95, Dobbr96, Dingrd90, Daemu93a, Daemu93b, Schno91, Schno93].

2. Data Encryption [Denni82, Welsh88, Sebry89, Schnr96, Palch97, Simon92, Menez97].

3. Authenticated key exchange [Sebry89, Schnr96, Belvn92, Diffi79, Diffi92, Belvn92, Gong93, Jabln96, Oorsc96, Simon92, Menez97, Merkl79]

1.1.2 Symmetric Error Correction and All Unidirectional Error detection

Current survey reveals that many faults in VLSI circuits (such as the faults that affect address decoders, word-lines, power supplies and stuck-fault in a serial bus, faults in ROMs and RAMs etc) cause mostly, unidirectional errors. The number of symmetric errors is usually limited while the number of unidirectional errors may be very large. In this background, several error control codes have been studied [Bose82, Lin83, Bose85, Nikol86, Lin88, Tao88, Blaum89, Rao89, Boinc90, Kundu90, Montg90, Nikol91, Bruck92, Basam94, Katti96a, Katti96b, Laih96]. It is felt that, construction of a hardware efficient, new class of symmetric error(s)correcting codes is very essential, which would be able to detect all unidirectional errors simultaneously.

1.1.3 Hypermedia data Modelling

Data modelling is an essential part of any systematic and modularized hypermedia application development process. The modelling requirements for a hypermedia data model are different from the traditional data models, particularly from navigational point of view. A hypermedia data model should be capable enough in expressing the semantics of the multimedia object-relationships including the navigational aspects Several hypertext and hypermedia data models have been studied from [Isakw95, Hofst93, Hughe91, Teory86, Halas94, Leget94, Schns93a, Schns93b, Stots89, Garzo93a, Garzo93b, Karmc96, Bomel91] RMDM (Relationship Management Data Model) [Isakw95] is one of those models, which was proposed for any complex, regular structure-based hypermedia applications It is observed that in the context of several moderately complex hypermedia domains, the aforesaid model fails to express some of the situations in a conceptual manner. This has motivated us to further investigate RMDM to find the solutions necessary to overcome the limitations detected so far

1.1.4 Image Database Processing and Retrieval

Answering a user-query in an image database system (IDS) typically requires correlation of digital data stored in different formats such as PCX, TIFF, GIFF, TARGA, BMP etc. Correlating these various pieces of information at the physical level by pre-analysis is not an attractive option. For example, there could be thousands of image-objects that could be recognized in a scene, but linking all of these objects to relevant textual or structured data would be very time consuming and un-rewarding process [Kashp95, Subra96]. It is believed that to support the correlation of all this information, it is necessary to represent the relevant digital data in *semantic level*. In general, humans are more capable than machines for abstracting information efficiently from images displayed on the computer This enables them to correlate information at a higher *semantic level* with other forms of representations such as the symbolic representation of data in structured databases This correlation of information at a semantic level in an efficient manner, across different representations is lacking in the existing IDS architectures and has been characterized as a semantic bottle-neck [Kashp94, Kashp95, Subra96, Jain94]. Researchers have looked into this problem of designing a suitable metadata-based architecture, and some preliminary works have been reported [Jain94, Gupta91, Chu94, Subra96, Klaus94]. It seems necessary to carry out further investigations on this problem of designing a suitable IDS architecture

1.2 Aim of Work

The main objective of the thesis is to explore CA, find some new characterizations and its applications in cryptography The thesis has also aimed to study the expressiveness of the existing hypermedia data models from conceptualization point of view. Further, it is also aimed to develop a suitable architecture for image database applications, irrespective of its nature, type or scale-factors

The thesis reports characterization of some of the group and non-group Extended Neighborhood CAs (ENCAs) based on a matrix algebraic tool. It has been shown that there exist group ENCAs which satisfy all the properties of maximum-length and non-maximum length 3-neighborhood additive group CAs Such maximum-length group CAs are useful for generation of high quality pseudo-random patterns [Nandi94a]. It has been shown that there exists a class of non-group ENCAs, which satisfy all the properties of TPMA CAs [Nandi94a]. Such non-group CAs are utilized for generating powerful hashing function.

Based on the above characterization, CA based applications have been developed in the following directions :

1. **CA based one-way hash function :** properties of non-group, non-linear complemented MACAs (Multiple Attractor CA) are utilized in defining a one-way hash function. To improve the intruders' complexity, CA based block ciphering functions [Nandi94a] are used as pre-processing steps to the MACA-based hashing module. The whole hashing process is operated in a feedback mode for several cycles.

2. **CA based data encryption :** a block encryption scheme has been developed based on the ultimate periodicity properties of group ENCAs. To improve the intruders' complexity, it utilizes a CA-based permutation scheme [Nandi94b] as a pre-processing step to the ENCA based block cipher. Due to the many-to-many mapping among the plaintext and ciphertext blocks, and due to the exponential order of key-space size, the scheme is guarded against all possible types of intruders' attacks.

3. **CA based password-only authenticated key exchange :** using the CA based schemes (for authentication and data encryption) developed so far, a new authenticated key exchange scheme based on a small password is developed. Due to the regular, modular and cascadable logic structure of CAs, the scheme has been found to be significant in comparison to the existing schemes The scheme was examined in light of the various known as well as unknown attacks and it has been found to be guarded and secure.

Besides, we have also extended our work in the following directions :

4. **Error detection and correction :** a new scheme to construct a t-EC/d-ED ($t < d$) and AUED systematic codes with faster and lesser hardware is discussed and established For larger values of t, in majority of cases, the proposed codes need lesser number of check-bits A new class of Single Error Correction (SEC) / Double Error Detection (DED) and All Unidirectional Error Detection (AUED) codes also have been developed based on *odd-weight column matrix* for *even-parity check* In

case of both the schemes, the ROM based implementation show significant results by reducing the ROM word-length.

5. **Study and analysis of hypermedia data models :** an analysis of a structured hypermedia application data model, referred as RMDM (Relationship management data model) [Isakw95] is discussed. Some drawbacks of RMDM are detected and subsequently, possible remedies also have been given, which ultimately extend the present RMDM to enable the modellers to represent any scale of application domains in a conceptual manner. Some implementation results based on the extended RMDM also have been reported.

6. **New architecture for image database system :** a modular, three-layered and user-friendly architecture for image database processing and retrieval, based on content-based and text-based features is presented. For better understanding of the complex queries, a visual semantic data model also has been defined. Further, for the sake of testing and justification, a prototype implementation of the proposed architecture also has been discussed.

The main contributions of the research work can be summerised as follows :

- Characterization of extended neighborhood additive CA behaviours (not explored by previous researchers).
- CA based one-way hash function design.
- Design of CA based data encryption scheme.
- Design of CA based authenticated key exchange based on a small password.
- Design of
  - SEC/DED/AUED systematic Codes,
  - t-EC/d-ED/AUED ($t < d$) systematic codes.
- Analysis of RMDM (a hypermedia data model), detection and reporting of the drawbacks of RMDM, remedies to overcome the drawbacks as well as to extend the RMDM and finally implementation of the extended RMDM.
- Design of a new image database system architecture and to assist in the query solving mechanism, development of a semantic data model.

1.3 Thesis Organization

To fulfill the aforesaid objectives, the research work reported in this thesis has been organized as follows :

- Prior to describing the application of cellular automata for design of one-way hash function, data encryption and authenticated key exchange schemes, the homogeneous structure of CA and its characterization based on matrix algebraic tool are briefly reviewed in *Chapter 2*. Some new characterization of the ENCA behaviours using matrix algebra as a tool, has also been reported.

- Based on the properties of non-group, non-linear complemented MACAs, a one-way hashing scheme is presented in *Chapter 3*. Application of such a CA based hash function in Electronic Economy has also been elaborated

- Based on the state transition features of group ENCAs, a block cipher system is presented in *Chapter 4*. Exploitation of programmable structures of additive group CAs in obtaining better randomness among the plaintext and ciphertext blocks, in the proposed cipher system, has also been reported.

- A new scheme on password-based authenticated key exchange, designed using the CA based schemes discussed so far in the previous chapters(*Chapter 3 & 4*) has been presented in *Chapter 5*. Various applications of such CA based schemes also has been established.

- Design of two efficient error control coding schemes has been presented in *Chapter 6*.

- Analysis of hypermedia data model in the context of a real-life case has been discussed *Chapter 7*; some drawbacks are detected and for remedies, solutions are suggested towards the extension of the data model. For justification of the extensions, implementation results are included

- New architecture for an image database system and its implementation issues in the context a real-life case have been reported in *Chapter 8*

- Finally, *Chapter 9* summerised the overall contributions of the thesis and identifies future research directions in the relevant fields

Chapter 2

Properties of Cellular Automata and its extension

2.1 Introduction

In early 50s, J. von Neumann first introduced Cellular Automata (as a 'cellular space') to study the complex structures capable of self-reproduction and self-repair [Neuma66]. Following him, several researchers have taken interest in the study of CA for modelling the behaviour of complex systems. Theory of CA received a consolidated attention by Burks [Burks70] and considerable simplifications were introduced by Codd, Banks, and Smith [Codd68, Palch97]. CA has been investigated as a class of Mathematical tool in early 75's for modelling a wide variety of biological systems such as self-reproduction, birth, growth, death etc. The development of structures and patterns in the growth of organisms, as pointed out in [Steve74], often appear to be governed by very simple local rules and can be modelled with a CA. Aladyev [Alady76] has surveyed various computational and theoretical aspects of CA, particularly the decidability of reachability problem for configurations. CA has also been used as a tool to study number theory and associated applications [Mille70, Apsim70]

However, it can be observed that in the early phase of the development, the theory has evolved in a bit unsystematic and fragmented way. Several researchers' persistent efforts [Yamad71, Amoro72, Richa72, Harao78, Toffo77, Wills75] resulted in the consolidation of the theoretical framework; they started analyzing CA by means of well tested mathematical tools such as abstract algebra, topology, measure theory, etc. Current intensive interest in this field can be attributed to the significant development of efficient schemes such as those for data encryption, pseodorandom pattern generation, error correction/detection etc. with the cost effective realization of the CA structure. In this chapter, development of CA and its applications in diverse fields have been briefly reviewed. It also introduces some new characterizations of the *Extended Neighbourhood* CA and its applications. In the following section, present development of CA based on the Wolfram's extensive investigations has been discussed briefly.

2.2 Present development of CA

Wolframs' investigations [Wolfr83, Wolfr84, Wolfr85a, Wolfr85b] on CA confirmed its usability as a mathematical model for self-organizing statistical systems. Several new characteristic features of self-organization in uniform 3-neighbourhood (left, right and self vide *Figure 2.1*) infinite CAs with 2-state per cell, were identified by him based on polynomial algebra.

The structure of a CA investigated by *Wolfram* can be viewed as a discrete lattice of sites (cells) where each cell can assume either the value 0 or 1 [Palch97]. At discrete time step (clock cycle), the evolution of a site value depends on some rule (a combinational function), which is a function of the present state of 'k' of its neighbours for a k-neighbourhood CA. For a 2-state 3-neighbourhood CA, the evolution of the i^{th} cell can be represented as a function of the present states of $(i-1)^{th}$, $(i)^{th}$ and $(i+1)^{th}$ cells as : $q_i(t+1) = f\{q_{i-1}(t), q_i(t), q_{i+1}(t)\}$, where f represents a combinational function. In effect, each cell as shown in *Figure 2.1*, consists of a storage element (D flip-flop) and combinational logic implementing the next state function.

In this Cellular Automata, there are 2^3 distinct neighbourhood configurations and 2^{2^3} distinct mappings from all these neighbourhood configurations to the next state; each mapping represents a CA rule

Neighbourhood State :	111	110	101	100	011	010	001	000	
Next State :	0	1	0	1	1	0	1	0	(*Rule*90)
Next State :	1	0	0	1	0	1	1	0	(*Rule*150)

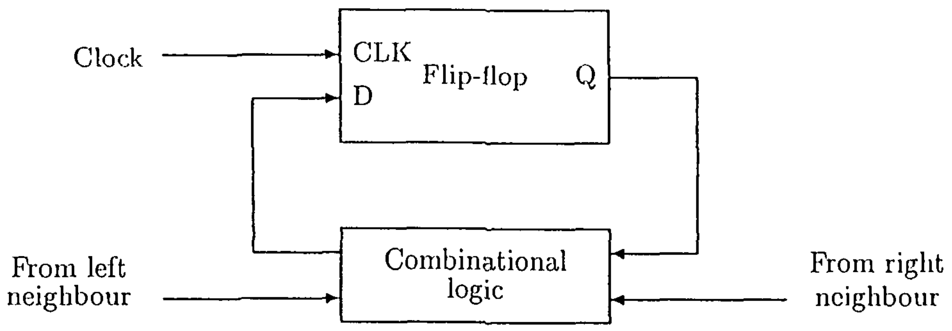


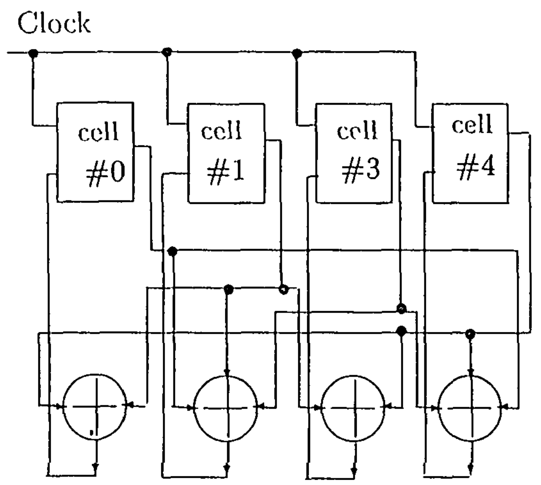
Figure 2.1: A typical CA cell

The top row gives all the 8 possible states of the 3 neighbouring cells at the time instant t, while the second and third rows give the corresponding states of the i^{th} cell at time instant $t+1$ for two consecutive CA rules. A detailed study of the truth tables for these rules can be found in [Palch97]. If in a CA the same rule applies to all of the cells, then the CA is referred to as a *uniform* CA; otherwise the CA is called *hybrid* CA. If the rule of a CA cell involves only XOR logic, then it is called a *linear* rule. Rules involving XNOR logic are referred to as *complemented* rules. A CA with all the cells having linear rules is called a **linear** CA, whereas a CA having a combination of XOR or XNOR rules is called an **additive** CA. The CA with non-linear rules is called **non-linear** CA.

A CA is said to be a **Periodic Boundary CA (PBCA)** if the extreme cells are adjacent to each other (*Figure 2.2*). A CA is said to be a**Null Boundary CA (NBCA)** if left(right) neighbour of leftmost(rightmost) terminal cell is connected to logic-0 state.

However, the polynomial algebra-based analytical tool has some difficulties in characterizing the hybrid CA employing non-uniform rules Also, a detailed polynomial algebraic analysis on non-group CA, had been lacking for long time. To overcome this, a new analytical tool based on **matrix algebra** [Das90b, Barde90, Das91] has been developed. In the next section, for the sake of completeness, some properties and results of 1-D 3-neighbourhood additive CA based on this matrix algebraic tool have been reported from [Palch97].

Definition 2.1 *An n-bit* **complement vector** *associated with an n-cell complemented CA is an n-bit binary vector in which a 1 in the i^{th} position indicates the rule at the i^{th} cell to be a complemented one*



$$T = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} x & 1 & 0 & 1 \\ 1 & 1+x & 1 & 0 \\ 0 & 1 & x & 1 \\ 1 & 0 & 1 & 1+x \end{bmatrix}$$

Characteristic polynomial $= x^4 + x^2$

Figure 2.2: Four cell periodic boundary CA with rule vector $< 90, 150, 90, 150 >$

Corresponding to each complemented CA, there exists a non-complemented CA whose state transition behaviour can be utilized as follows to derive the behaviour of the complemented one. For a complemented CA with the complement vector S_c, the next state $S(t+1)$ of a present state $S(t)$ is represented as

$$S(t+1) = [T]S(t) \oplus S_c$$

where $[T]$ is the CA matrix of the corresponding non-complemented CA. The state transition diagrams of cellular automata have been characterized using the $[T]$ matrix and its characteristic polynomial. Depending on the nature of their state transition behaviour, CA can be broadly classified into two classes - (i) **group CA** and (ii) **non-group CA**. In the state transition diagram of a group CA all states are cyclic; each state has a unique successor and a unique predecessor state. The state transition diagram of two group CAs are shown in *Figure 2.3 & 2.4*.

For the sake of completeness, some of the basic definitions and few important results related to *group* and *non-group* CAs have been reproduced from [Das90b, Pries86, Barde90, Serra90, Nandi94a] below.

Group CA :A group CA has been identified by the condition that the $[T]$ matrix is nonsingular. In other words, there must exist some positive integer p such that $[T]^p = I$ (*the identity matrix*), and $S(t+p) = [T]^p S(t) = S(t)$

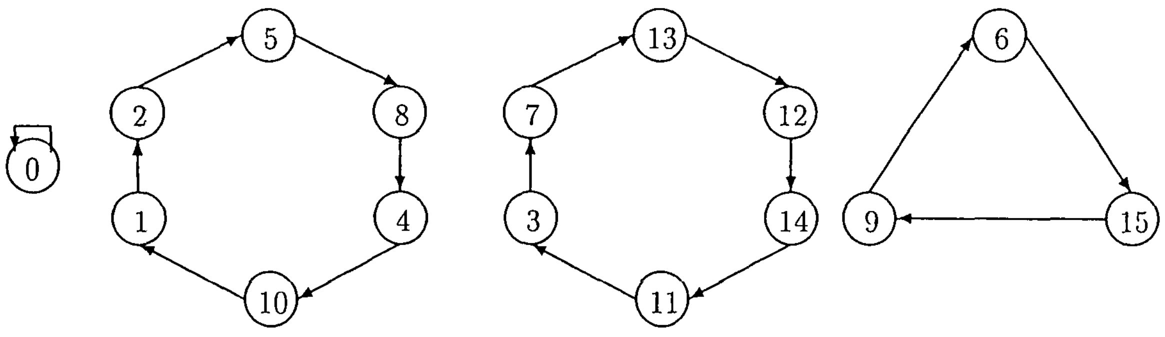


Figure 2.3: The state transition diagram of a nonmaximum length group CA

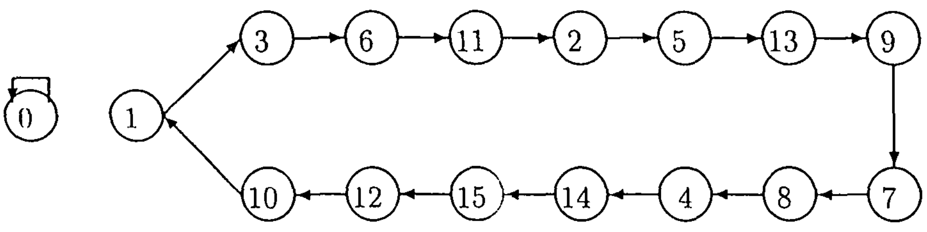


Figure 2.4: The state transition diagram of a maximum length group CA

Theorem 2.1 *A CA is a group CA if and only if the determinant det[T]=1, where [T] is the characteristic matrix of the CA.*

Theorem 2.2 *A group CA has a cycle of length p or factors of p with a nonzero starting state if and only if* $det[[T]^p \oplus [I]] = 0$.

Theorem 2.3 *Let* $\overline{T}^p$ *denote application of the complemented rule* $\overline{T}$ *for p successive cycles, then*

$$\overline{T}^p S(t) \;=\; [I + T + T^2 + \cdots + T^{p-1}][S_c] \;+\; [T^p][S(t)]$$

where T is the characteristic matrix of the corresponding non-complemented rule vector and $[S_t]$ *is an n-dimensional vector (n = number of cells) responsible for inversion after XORing.* S_c *has '1' entries (i.e. nonzero entries) for CA cell positions where XNOR function is employed.*

Theorem 2.4 *The complement of a group CA is also a group CA.*

Theorem 2.5 *CA rules 60, 102 and 204 form groups for all lengths(l) with a group* order $O(G) = p = 2^a$; $a = 0, 1, 2, \cdots$; $m \geq l > m/2$.

Group CAs can further be classified into *non-maximum* (*Figure 2.3*) and *maximum* (*Figure 2.4*) length CAs. An n-cell *maximum* length CA is characterized by the presence of a cycle of length $(2^n - 1)$ in the corresponding state transition diagram and a primitive characteristic polynomial.

All states of a CA with complemented counter part (with rule 195, 153 and 51) form cycles of equal length and cycle length is 2^j, where $j = 1, 2, 3, \cdots$ and $j \neq 0$ [Nandi94a].

Lemma 2.1 : *Complemented CA rules 195, 153 and 51 form groups for all lengths with group order* $O(G) = m = 2^a$; $(a = 1, 2, 3, \cdots)$.

Theorem 2.6 : *If a null boundary uniform or hybrid CA configured with rules 51, 153 and 195 is a group CA, then its state transition diagram consists of equal cycles of even length.*

The null boundary CA with rule configuration $< 153, 153, 153, 153 >$ generates cycles as shown in *Figure 2.5*.

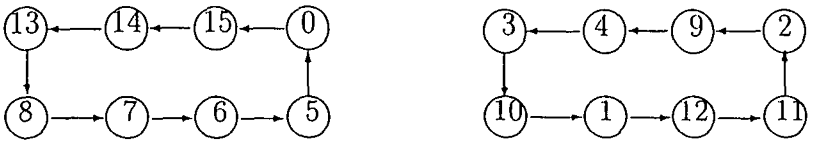


Figure 2.5: The state transition diagram of a nonmaximum length group CA

In periodic boundary conditions, the only rule available forming equal cycles of even length is rule 51.

Non-Group CA : In a non-group CA (*Figure 2.6*), all the states are not cyclic, moreover, a state can have 0 or 2^p $(p \geq 1)$ number of predecessors (*Figure 2.7*). In the state transition diagram of a CA, a state is said to be **reachable** if it has one or more predecessors. A state without any predecessor is referred to as a **non-reachable** state. Reachable states which form cycles are called **cyclic states**.

Definition 2.2 *A state with a self-loop is referred to as a* **graveyard state (or attractor)**. *In other words, an attractor is a unit length cyclic state.*

In the state transition graph, **level** of a state S_i is defined as the minimum number of time steps required to reach a cyclic state starting from S_i. Level of a cyclic state is counted to be zero. In *Figure 2.7*, level of state '15' is 2. Thus node 15 can be marked as 2-level predecessor of the attractor 3.

Definition 2.3 Depth *of a CA can be formally defined as the minimum number of time steps required to reach a cyclic state from a state in the highest level of the state transition diagram of the CA.*

Depth of the CA in *Figure 2.7* is 3. All the i-level $(0 \leq i \leq d)$ predecessors of an attractor form an inverted tree rooted at the attractor, where d is the depth of the tree.

Such an inverted tree is simply mentioned as a *tree* in the rest of the thesis.

Definition 2.4 *All the i-level $(0 \leq i \leq d)$ predecessors of an attractor constitute the corresponding* **attractor basin**, *where d is the depth of the tree rooted at the attractor. The basin with a state α as attractor is referred to as α-basin (*Figure 2.7*).*

If l is the smallest integer for which $T^{d+l} = T^d$, for $d > 0$, then the CA is a non-group CA with depth d and cycle length factors of l.

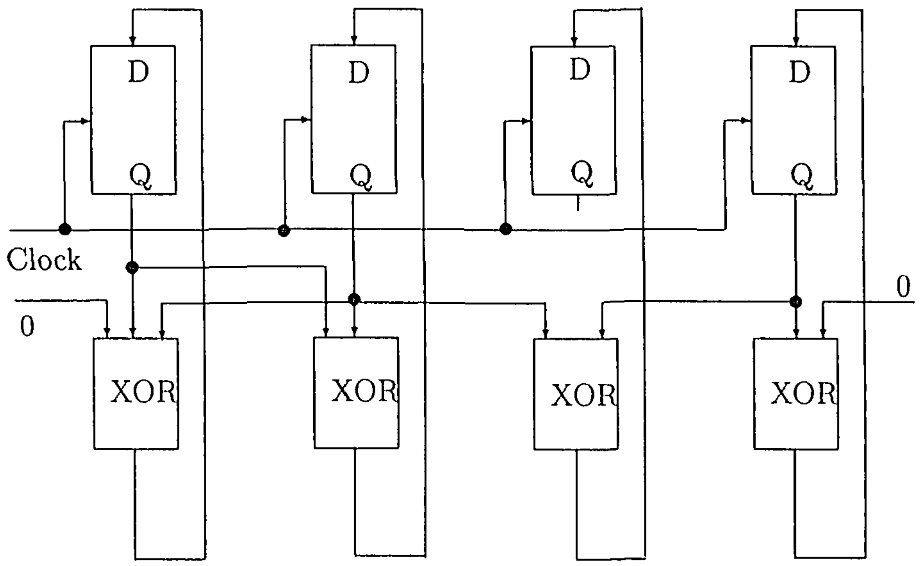


Figure 2.6: A 4 cell null boundary CA with rule $< 102, 60, 90, 204 >$

Example 2.1 The state transition diagram of a non-group CA (*Figure 2.6*) having rule vector $< 102, 60, 90, 204 >$ with periodic boundary is shown in *Figure 2.7*. The

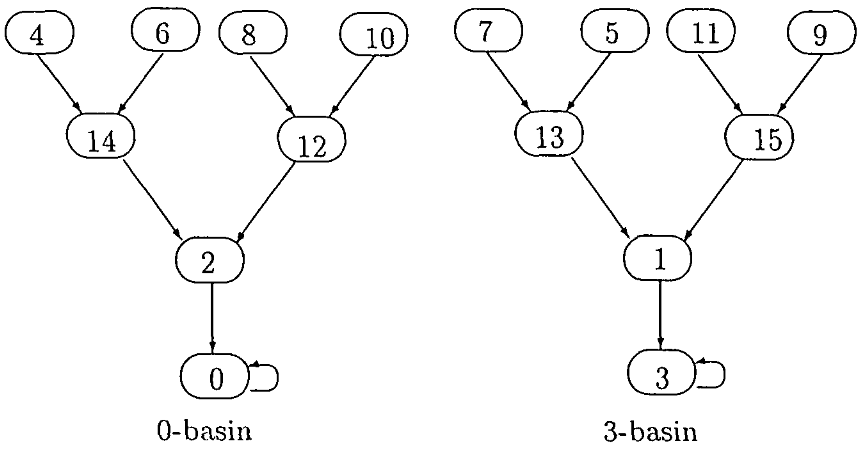


Figure 2.7: State transition diagram of non-group CA with rule $< 102, 60, 90, 204 >$

characteristic matrix T, characteristic polynomial $p(x)$, depth, graveyard state (attractor) and attractor basins are illustrated below.

$$T = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Characteristic poly.	:	$p(x) = det(T + Ix) = x^3(x+1)$
Depth	:	3
Attractors	:	0 and 3
0-basin	:	$\{$ 0, 2, 4, 6, 8, 10, 12, 14$\}$
3-basin	:	$\{$ 1, 3, 5, 7, 9, 11, 13, 15$\}$

□

Some of the results related to non-group CA's, are restated from [Das90b] below without any proof.

Theorem 2.7 *If for a largest value d, x^d divides the minimal polynomial $f(x)$ of the T matrix of a CA, then the depth of its state transition graph is d.*

Theorem 2.8 *If a state is reachable from p predecessors for a CA, then only $1/p$ of the total states are reachable.*

Theorem 2.9 *The number of predecessors of a state in the state transition graph is 2^{n-r}, where r is the rank of the T matrix of the n cell CA.*

Theorem 2.10 *If for a CA of depth $d > 0$, l is the smallest integer for which $T^{d+l} = T^d$, then the CA state transition graph has cycles of lengths which are factors of l.*

Theorem 2.11 *In an n-cell non-complemented CA with characteristic matrix T, num-*

ber of graveyard states is 2^{n-r} where r is the rank of the $[T+I]$ matrix Any graveyard state can be represented as a linear combination of $n-r$ linearly independent state vectors.

In the next subsection, a special class of non-group CAs, referred to as Multiple Attractor CA (MACA), where all the cycles are of unit-length, has been discussed. Some important properties of this class of CAs are also cited. This class of CAs has been used to realize efficient hashing function.

2.2.1 Properties of Multiple Attractor CA (MACA)

The state transition diagram of a MACA is shown in *Figure 2.7*. It is a non-group CA with the following properties:

- each of its reachable states has two predecessors; and
- it has a set of cyclic states, each cycle of unit length;

that is, each cyclic state is a graveyard state (also referred to as an **attractor**).

The structure of the state transition diagram of a MACA consists of a subset of states leading to an attractor, in an inverted tree-like fashion. That is, if the CA is loaded with any state in the tree, after a few cycles (equal to the depth of the tree) the CA will evolve to the attractor state. Each such set of states along with its attractor is called a *basin* (*Definition 2.4*). Thus, the complete set of CA states are divided into a number of basins, and each basin is built around a specific attractor. One of the attractors is always 0-state (all 0's binary bit) and its corresponding basin is known as the *0-basin*. Any other basin with non zero attractor (say with decimal value i) is referred to as a *i-basin*. In *Figure 2.7*, states (in decimal notation) 0, 2, 4, 6, 8, 10, 12 and 14 belong to the 0-basin, while the states in the 3-basin are 1, 3, 5, 7, 9, 11, 13, 15. The states are distributed at different levels of a basin. If the CA is loaded with a j^{th} level ($j > 0$) state on the i^{th} basin, the CA will evolve to state i after j number of cycles The 0^{th} level state is the attractor itself with self-loop and so it needs one cycle to reach to itself. The results reported below from [Nandi94a, Palch97] characterize some important properties of MACAs.

Lemma 2.2 *The sum of two parent states of any state is the non zero predecessor of the 0-state.*

Theorem 2.12 *The sum of two states lying at different levels p and q $(p > q)$ of the i basin is a state at level p of the 0-basin.*

Lemma 2.3 *There exists a one to one correspondence between the states of the 0-basin and i-basin of a MACA.*

Lemma 2.4 *In a d depth MACA, if the level wise transitions from a non-reachable state to its attractor are known, the states in the 0-basin can be enumerated level-wise.*

Lemma 2.5 *In a d depth n bit MACA, if* (i) *the level wise transitions of a non-reachable state to its attractor, and* (ii) $(n - d)$ *linearly independent attractors are known, then states belonging to all the basins can be enumerated level-wise.*

Lemma 2.6 *In a d depth $(d > 1)$ MACA, the sum of all states in any basin is zero.*

Lemma 2.7 *No of attractors in the complemented MACA is same as that in the original linear one.*

Lemma 2.8 *The complemented CA derived from multiple attractor linear CA can not have cycles of length greater than unity.*

An important advantage of 3-neighbourhood CA is its *programmability features*, which have been found to be useful in many applications such as cryptography, pseudo-random pattern generation, test pattern generation etc. One of the major advantages of CA is that it can be programmed with minimum logic structure. The main objective of a Programmable CA (PCA) is to configure the CA with different rules at different instants of time. An application of a variant of such PCA (based on *rule 90* & *rule 150*) can be found in [Nandi94a]. As from the positional representation of *rule 90* and *rule 150*, it is evident that they are neighbourhood dependent, differ by only one position, namely viz. on the cell itself. Therefore, by allowing a single control line per cell (*Figure 2.8*), one can apply both *rule 90* and *rule 150* on the same cell at different time steps. Thereby, an n cell CA structure can be used for implementing 2^n CA configurations. Realizing different CA configurations on the same structure can be achieved by using a control logic to control the appropriate switches; a control program, stored in a ROM, can be employed to activate the switches. The 1(0) state of the i^{th} bit of a ROM word closes (opens) the switch that controls the i^{th} cell. *Figure 2.8* shows a PCA with simple control structure- allowing one control input per cell that configures the CA cell, either to rule 90 or rule 150. For example, the control word $< 0110 >$ for a four cell PCA leads to the fact that the 1st and 4th cells are configured with rule 90, while the 2nd and 3rd cells are configured with rule 150.

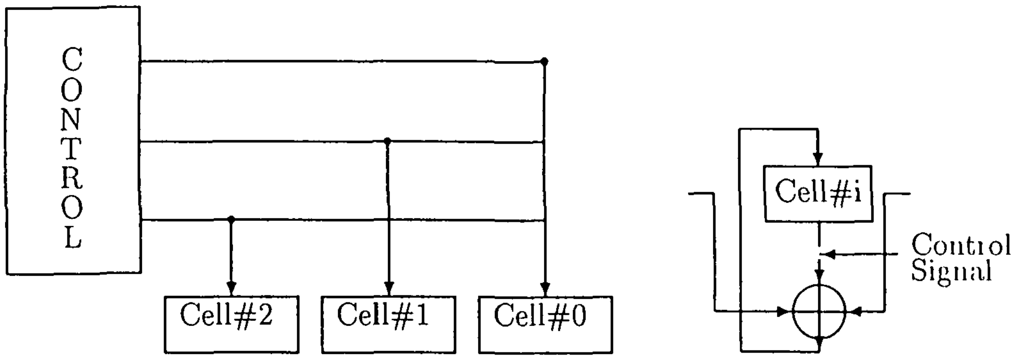


Figure 2.8: A 3-cell Programmable CA structure and a PCA cell

Using such a structure for all the cells, all possible additive non-complemented rule combinations can be achieved to realize any hybrid additive CA [Nandi94a]. Such an n cell PCA can implement 2^{3^n} number of different CA configurations. A 3-neighbourhood PCA cell with provision of both complemented and non-complemented rules is shown in *Figure 2.9*. This is the structure of the most general PCA cell with 4 control lines per cell.

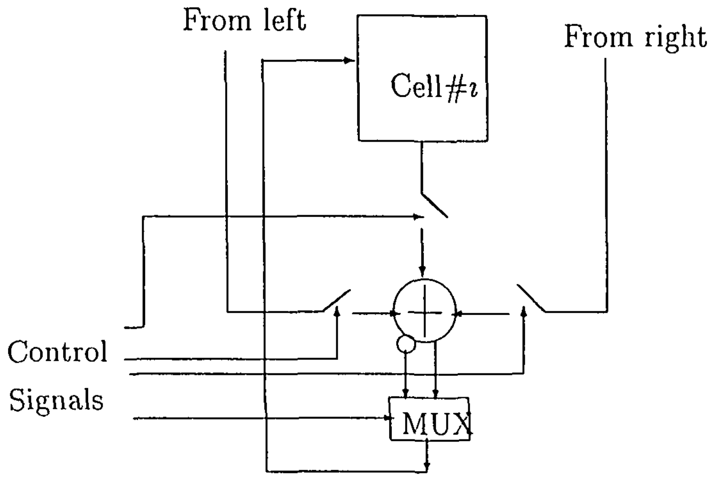


Figure 2.9: A PCA cell with all possible additive rules

2.3 Characterization of Extended Neighbourhood CA (ENCA)

In this section, a general class of CA (based on n-neighbourhood interconnections) is investigated. The characterizations of this class of CAs are basically extensions of the group and non-group 3-neighbourhood additive CAs, discussed in the previous sections. If the ENCA is *non-singular* and with a repeated application of the CA in a finite set of elements, if the entire set of states ultimately converge to a single or multiple cycles, it is referred to as a *group* ENCA; otherwise, it is called *non-group* ENCA. Both these categories of ENCAs have some important properties, which can be successfully utilised in the field of cryptography, pseudo-random pattern generation, hashing etc. In the next section, characterization of these CAs, in general, has been given.

2.3.1 General Characterization

Consider any finite set of elements, $V = \{\alpha_0, \alpha_1, \alpha_2, \cdots, \alpha_w\}$ which is closed with respect to an associated operator T (i.e.an ENCA). In a crude analogy with construction of a 'power' sub-group of a multiplicative group, one can generate a sequence, say, $\alpha_0 = a_0, \alpha_1 = \alpha_0 + a_0, \alpha_2 = \alpha_1 + a_0, \cdots, \alpha_{r+1} = \alpha_r + a_0, \cdots$. Since the set V is finite and as the sequence is gradually built, at some stage, i.e. say, for some $\gamma \leq w$, the ordered subset $V_1 : \{\alpha_0, \alpha_1, \cdots, \alpha_{\gamma-1}\}$ has all elements distinct but $\alpha_\gamma = \alpha_{\gamma-1} + a_0 \in V_1$ and from then onwards, the sequence $\alpha_0, \alpha_1, \cdots, \alpha_{\gamma-1}, \alpha_\gamma$ will start repeating itself, though not necessarily from the beginning; suppose, $\alpha_\gamma = \alpha_{al+1}$, where $al < w$ and hence, the sequence will be $\alpha_0, \alpha_1, \cdots, \alpha_{al}, \alpha_{al+1}, \cdots, \alpha_{al+p}, \alpha_{al+p+1} = \alpha_{al}, \cdots$, with an *aperiodicity* 'al' and (ultimate) *periodicity* 'p', i.e. $al + p = w$.

However, if $\gamma < w$, the set V has elements not included in the subset V_1. Starting with another element say, $a_2 \notin V_1$, one can build another sequence $\alpha_{21}, \alpha_{22}, \cdots \notin V_1$ until one of these elements repeats itself with an ultimate periodicity say, 'p_2' and aperiodicity 'a_2' or, some $\alpha_\beta \in V_1$, in which case the sequence or stream 'merges' into the earlier stream V_1. In *Figure 2.10*, an illustration is given

Thus, in general, an ENCA generates *trajectories*, starting from some element, (one can call these elements as *non-reachable* or *starting* states, which have no *predecessors*

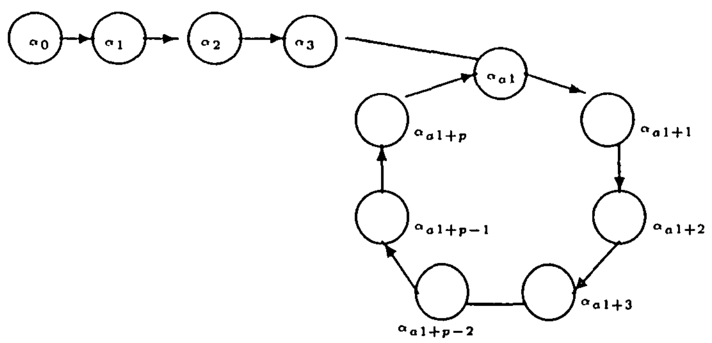


Figure 2.10: Basin structure showing 'aperiodic' behaviour

but, are different *starting points* and ending in possibly more than one cycles (lengths of cycle may even be *one* only, i.e. *attractor*)). In other words, an *Extended-Neighbourhood* CA (ENCA) partitions the elements of the 'operand set' into partially ordered sets (POSET), referred as *basins*, where the structure of each basin is characterized by the type of CA chosen. These basin-structures exhibit some peculiar characteristics which can be of use in many different contexts. Some of the results found from the analysis of the various types of ENCA based basin structures, have been reported next.

Example 2.2 Consider the characteristic matrix

$$T = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

It is seen that $T^3 = I$ (i e. identity matrix) and the characteristic equation is $x^5 = x^4 + x^3 + x^2 + x + 1$; $x^6 = 1$, i.e. $T^6 = I$; Hence, $T^2 \cup T^3 \cup T^6 = I$, i.e. lengths of cycles either may be 2 or 3 or 6; It is verified that $T^2 \neq I$ but $T^3 = I$. It follows that there can be cycles of length 3 only. Also, since T is non-singular, every state has only one *parent* (possibly itself). Hence, in the basin structure induced by this CA- there can be either 'attractors' or cycles of length *three* only; every state is either its own parent or it is a member of a 3-cycle. To obtain all the attractors of T, one has only to find the solutions of $(T + I)\Delta = 0$, (where Δ is a boolean variable). The only possible solutions are $(\Delta 0 \Delta \Delta \Delta)'$. Thus, there are only 2 attractors, viz 0 itself and 23 The other 30 states in the space are partitioned into 10($= 30 \div 3$) cycles as follows ·

$$\{1, 5, 14\}, \{2, 18, 7\}, \{3, 29, 9\}, \{4, 31, 27\} \cdots$$

□

For illustration, see *Figure 2.11*.

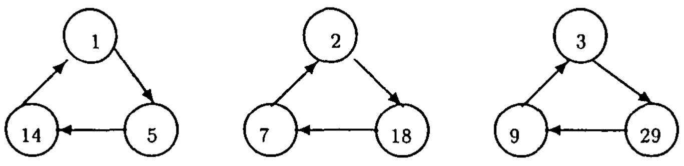


Figure 2.11: State transition diagram of non-maximum length group ENCA

If the characteristic polynomial of group ENCA is primitive, then it exhibits properties analogous to the 3-neighbourhood additive, maximum-length CAs [Nandi94a]. Such an ENCA of size n generates all the 2^{n-1} states in successive cycles, excluding the all-zero state. One such CA with the maximum-length cycle is illustrated in *Example 2.3*.

Example 2.3 Consider the characteristic matrix

$$T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

From the matrix T, it can be found that it is of non-singular type and the characteristic equation is : $x^5 + x^4 + x^2 + x + 1 = 0$; This has $x = 1$, not as a root and it does not possess any attractor other than 0. Also, we have $x^{31} = I$. So, this CA gives rise to a fully periodic basin (maximum-length cycle) of all 31 non-zero state, as shown in *Figure 2.12*.

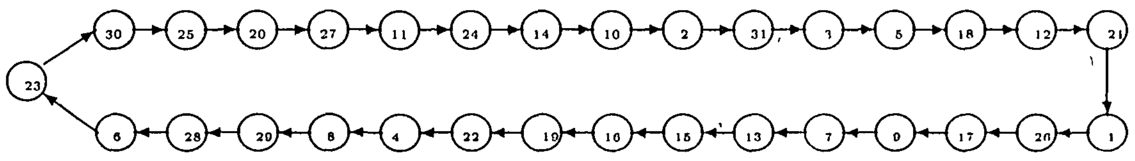


Figure 2.12: State transition diagram of maximum-length group ENCA

□

2.3.2 Properties of Group ENCA

Some of the properties of 3-neighbourhood additive CAs from [Palch97, Nandi94a]) derived with matrix algebraic tool were tried with ENCAs and similar results were obtained. The findings of such investigations are reported next. However, the proofs of *lemmas* and *theorems* are not reported here, because they are analogous to those reported in the *Section 2.2* of the present chapter.

Theorem 2.13 *An ENCA is group CA, if and only if the determinant* $\det T = 1$, *where* T *is the characteristic matrix for the CA.*

Theorem 2.14 *A group ENCA has cycle-lengths of* m *or factors of* m *with a non-zero starting state iff* $\det\, [T^m \oplus I] = 0$.

Lemma 2.9 *If the order* (O_P) *of a group ENCA characterized by* T *is a non-prime number, then the cycle-lengths are equal to its factors only.*

Theorem 2.15 *If the characteristic polynomial of a group ENCA is primitive it generates a maximum-length cycle.*

Non-group ENCAs form a special class of CAs, in which some of the states are not reachable from any state. In contrast to the *group* ENCA, (i) in the state transition graph of such CA, the reachable states can have multiple *predecessors*, and (ii) the characteristic matrix is *singular* in nature. The study of non-group CA behviour has not received sufficient attention for long time. Next section presents some of the interesting properties of this class of CAs. In *Examples 2.4 & 2.5*, an analysis of the basin structure due to a 5-cell non-group ENCA, has been given. From state transition graph, it can be found that the cyclic states are lying on one or more cycles. Other states form inverted trees rooted at one of the cyclic states. Such inverted trees are referred here as simply *trees*. The 'cycles' in the state transition diagram of a non-group ENCA are referred as *attractor*. Thus an *attractor* is a cycle of length $l (l \geq 1)$. The tree rooted at a 'cyclic-state' α, is denoted as 'α-basin'. The depth of such a basin is defined to be the minimum number of 'clock-cycles' [Palch97] required to reach the nearest cyclic state from any non-reachable state in the state-transition graph.

2.3.3 Properties of Non-Group ENCAs

This subsection presents some fundamental results characterizing a special class of non-group ENCA (analogous with the TPMA 3-neighbourhood CA of [Palch97]). For the sake of clearness about the behaviour of such ENCAs, the following examples have been cited.

Example 2.4 Consider the CA

$$T = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Thus, we have the characteristic equation as : $x^5 + x^4 + x^3 = 0$ and here, $x^6 = x^3$. Hence, $T^6 = T^3$. Thus, for any state S, one has $T^6.S = T^3.S$ and any sequence $\{S_n | S_n = T.S_{n-1}\}$, will have at most only 5 distinct states, the sixth state S_6 in the sequence being identical with S_3. The sequence is ultimately periodic with periodicity either 3 or a factor of 3. Of course, since 3 is prime, the ultimate periodicity is either 1 or 3 only. Again, since, $x^5 + x^4 + x^3 = 0$, the only eigen value admissible is $x = 0$. Solving the equation $T.S = 0$, we have $T[00101]^t = T.5 = 0$ as the only solution, other than 0, of $T.S = 0$. Also, $T.S = b$ has no solution if $b_1 + b_2 + b_4 = 1$, i.e. there are $2^2.(\binom{3}{1} + \binom{3}{3}) = 16$ states got by choosing components b_1, b_2 and b_4 such that only one of them or all these are '1', the elements b_3 and b_4 being arbitrarily '0' or '1', which can not be reached from any other state. These only can be the *non-reachable* states of a trajectory. Since, $T.5 = 0$, we have $T.(S + 5) = T.S$, only for all 'S'. Hence, if S has $T.S = b$, then $T.(S + 5) = b$ also, i.e. for any state in the trajectory other than a starting state, there will be two *parents*, which differ by the state **5**. Thus, for instance, since $T.$**28** $=$ **1** we have $T.($**28** $+$ **5** $=$ **25**$) =$ **1** also.

Also, since all the columns of T have now been used as a parent, with its *progeny* obtained, one can compute $T^2, T^3, \cdots$ by simple table look-up. The structure and contents of the table is as shown in *Table 2.1*. Again, from the characteristic equation i.e. $x^5 + x^4 + x^3 = 0$, as it can be observed that $x = 1$ is not a root, there is no 'S' such that $T.S = S$. Also, we have $T^6 = T^3$, thus the periodicity is $(6 - 3) = 3$. So, cycles of length 3 can exist as the terminal points of the trajectories.

Table 2.1: CA and Its Power Sequences

Operators	Corresponding Sequences
T	$28, 18, 15, 25, 15$
T^2	$1, 5, 11, 1, 11$
T^3	$15, 0, 4, 15, 4$
T^4	$11, 0, 15, 11, 15$
T^5	$4, 0, 11, 4, 11$
T^6	$15, 0, 4, 15, 4$

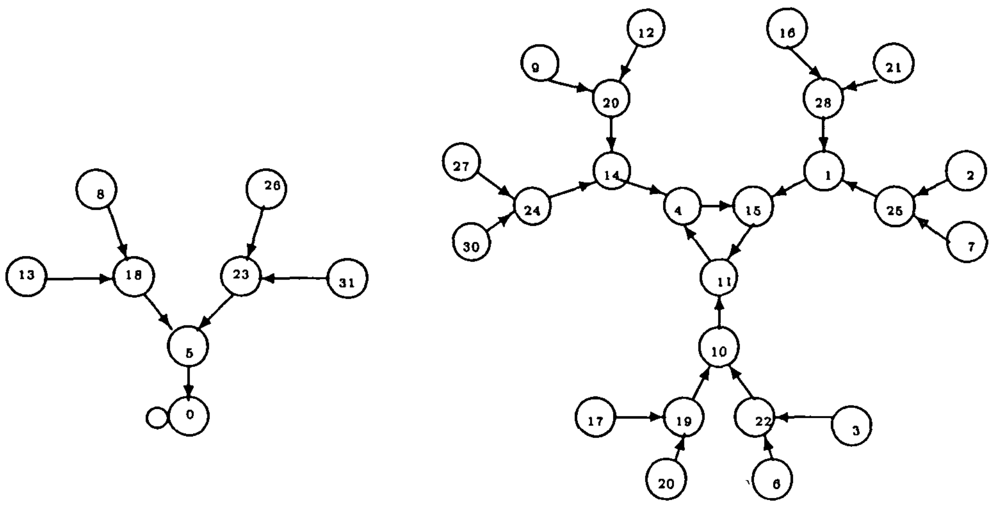


Figure 2.13: Basin structure generated by non-group ENCA

Since, each *offspring* shall have only two parents, i.e. $T.S = b$, if at all solvable, has only two solutions, and since each element in a cycle necessarily has one *parent* in the cycle, there should be another *parent* to it, which is not in the cycle. If the *parent* is itself not a *starting state*, it should have exactly two parents, etc. Further, from $T^3 = (15, 0, 4, 15, 4)$, we find, for $(T^3 + I)$ that- there are exactly four '*eigen values*' of T^3, including the trivial 0 state. Hence, there is only one cycle of length 3 (for illustration, see *Figure 2.13*). □

Example 2.5 Consider the CA

$$T = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

Thus, we have the characteristic equation as : $x^6 - x^5 - x^2 = 0$; from this characteristic equation it is obvious that the CA is singular and $x = 1$ is not a root.Hence, there is no non-trivial *eigen vectors* with $x = 1$. Also, we have -
$x^6 = x^5 + x^2$;
$x^7 = x^5 + x^3 + x^2$;
$x^8 = x^5 + x^4 + x^3 + x^2$;
$x^9 = x^4 + x^3 + x^2$;
$x^{11} = x^4 + x^2$;
$x^{13} = x^5 + x^4 + x^2$;
$x^{14} = x^3 + x^2$; $x^{17} = x^2$;

Since, $x^{17} = x^2$, we have, *aperiodicity* $a = 2$ and *periodicity* $p = 15$. Hence, one can have cycles only of lengths $(1 \cup 3 \cup 5 \cup 15)$. From the derived power sequences, we have : $T^3 = (14, 45, 14, 22, 0, 51)$; $T^5 = (51, 30, 51, 35, 0, 22)$; and $T.S = 0$ has only one solution : $S = \mathbf{42}$. It provides the only cycle of length 1. Also, for the starting state, $\beta_1 + \beta_4^{\cdot} + \beta_5 = 1$. Hence, there are 32 non-reachable states. Thus, in this basin of 64 states, 32 are non-reachable states; since, each offspring has exactly two parents (one of them possibly, itself) which differ by the value **42**, the 32 starting states can be paired off with 16 pairs, each pair giving rise to one reachable state. There is a cycle of length 15 and each state on the cycle is having one predecessor in the cycle and hence must have one parent not in the cycle; these have to be from among the 16 *offsprings* from the starting states. Hence, the 15 elements in the cycle and their non-cycle parents, again 15 in number, and their two parents each of which is a starting state, account for 60 of the states in the space. This leaves a set of 4 values : 2 starting states, their one *offspring* and its *offspring* which should also be its own parent (viz. the *zero* state) which constitute a tree like structure. Since **42** is the only non-trivial solutions of $T.S = 0$, each *offspring* has exactly two parents, the state difference between them being **42**. Also, since $T^3 + I$ and $T^5 + I$ are non-singular (in fact they are permutation matrices), they can not be cycles of length 3 or 5. Hence, $T^{15} = (8, 16, 8, 6, 0, 43)$ and $T^{15} + I$ is of *rank* 2. Hence, there are $2^4 = 16$ states which are such that $T^{15}.S = S$; this set, of course includes the zero state. The other 15 states constitute a cycle, of length 15. □

The following *lemmas* and *theorems* have been found to be valid for this special class of non-group ENCAs :

Theorem 2.16 *The number of predecessors of a reachable state and those of the state 0 in a non-group ENCA are equal.*

Lemma 2.10 *If d is the dimension of the null space of the characteristic matrix of a non-group ENCA, then the total number of predecessors of the all zero states (i.e. 0-state) is 2^d.*

Lemma 2.11 *If a state is reachable from 'k' predecessors for an ENCA, then only $1/k$ of the total states are reachable.*

Lemma 2.12 *If an n-cell non-group ENCA with characteristic matrix T, the number of attractors is 2^{n-r}, where 'r' is the rank of the $(T \oplus I)$ matrix.*

Lemma 2.13 *The number of l predecessors ($l > 0$) of any cyclic state is the same as that 0-state.*

Lemma 2.14 *The number of states at level 'l' ($l > 0$) of a tree rooted at a cyclic state is the same as the number of states at level 'l' of the tree rooted at 0-state.*

Lemma 2.15 *The 'depth' of a tree rooted at any cyclic state cannot be more than that rooted at the 0-state.*

Theorem 2.17 *Let 'd' be the largest integer ≥ 0 such that x^d divides $m(x)$ (the minimal polynomial of the ENCA). Then, each of the cyclic states in the state transition graph is the root of a tree of depth 'd' and the structure of each tree is isomorphic to that of the tree rooted at the 0-state.*

Lemma 2.16 *The sum of two parent states of any state is a non-zero predecessor of the 0-state.*

Theorem 2.18 *The sum of two states lying at different levels 'p' and 'q' (where, $p > q$) of the i^{th} tree is a state at level 'p' of the 0-tree.*

Lemma 2.17 *For an n-cell d-depth non-group ENCA with characteristic polynomial $x^d(x+1)^{n-d}$ and minimal polynomial $x^d(x+1)$ -*

- *the number of attractors is 2^{n-d},*
- *the number of states in each attractor-tree is same and equal to 2^d.*

Theorem 2.19 *In an n-cell d-depth non-group ENCA with 2^m attractors, there exist m-bit positions at which the attractors give pseudo-exhaustive 2^n bit-patterns.*

2.4 Conclusion

The theory of 3-neighbourhood additive CAs based on matrix algebra has been reported in this chapter. We have further extended characterization of n-neighbourhood or extended-neighbourhood CA i.e. ENCA based on matrix algebraic tools. The various group and non-group properties of ENCAs have been reported. In the subsequent chapters (*chapters 3, 4 & 5*), applications are developed by utilizing properties of these 3-neighbourhood and ENCAs.

Chapter 3

One-Way Hash Function Design Using Cellular Automata

3.1 Introduction

In the previous chapter, the theory of group and non-group 3-neighbourhood Cellular Automata(CA) has been addressed The regular, modular and cascadable structure of those additive CAs has been successfully exploited as a tool in several application areas, such as cryptography, error correction/detection, pseudo-random pattern generation, test pattern generation etc. Application of CA in other areas has also been explored. One such area is the one-way hash function design.

One-way hash functions play a fundamental role in modern cryptography. For the purpose of authentication of transmission of confidential data in insecure network, one-way hash function has an important role. They are used as building blocks in many protocols. Most of the one-way hash functions $H(M)$s operate on an arbitrary-length message, M and generates a fixed-length information, h, known as 'hash value'. Two common properties in these hash-functions are : *firstly*, it is irreversible in nature and *secondly*, it is hard to find another message M' , such that $H(M) = H(M')$. One-way hash function has several applications, such as: public key cryptography, distributed database security, network security etc. It has been found that most of the one-way hashing schemes operate with $2r$-bit input and r-bit output, where, the input to the function is the block of text and the hash of the previous block of text. That is, hash of a block M_i is, $h_i = (M_i, h_{i-1})$, where h_{i-1} is the hash of the previous block used on feedback mode. The hash of the last block becomes the *hash* of the entire message.

In this chapter, a new one-way hashing scheme for message authentication purpose has been designed by utilizing mainly, the features of *non-group, non-linear, Multiple Attractor*-based 3-neighbourhood CA. The scheme can be found to be significant in comparison to the other existing schemes [Merkl90, Rivst92, NIST93, Schnr96], in view of the following facts :

1: A CA whose rule vector can be modified dynamically is referred to a PCA (Programmable Cellular Automata) (see *Chapter 2, Section 2*). Such a PCA based hardwired implementation of the hash function provides better flexibility of configuring different CAs from the same structure.

2: Due to the simple, regular, modular and cascadable structure of CA, the implementation of this scheme becomes attractive and it can be operated at higher speed due to the local inter-connection.

3: It provides better security against different types of attacks because, besides using a general one-way hashing strategy analogous to the other existing ones, it also incorporates features of non-group, non-linear CAs.

There are several applications of one-way hash functions, such as distributed database security, network security, public key cryptography, authenticated key exchange etc. With the enormous growth of information technology, another major application of one-way functions is *Ecash* or *Electronic economy*. Utilization of the proposed scheme has been shown in the context of such an *ecash* application. In the next subsection, a brief review of the existing one-way hashing schemes has been reported.

3.1.1 Reviewing the existing schemes

A definitive reference for the cryptographic hash function and its detail survey can be found in [Precn94]. Davis and Price [Davis83] also provides a solid treatment of message authentication and data integrity. Simmons independently developed a general theory [Simon92] of unconditionally secure message authentication schemes and the subject of authentication codes. Rabin [Rabin78] first suggested employing a one-way hash function in conjunction with a one-time signature scheme and later in a public key signature scheme. Merkle [Merkl90] further explored uses of one-way hash functions for authentication, including the idea of *tree authentication* for both one-time signatures and authentication of public files. The DES-based hash-function of Merkle [Merkl90] which employs a compression function f mapping 119-bit input to 112-bit output in 2 DES operations, allows 7-bit message blocks to be processed (at the rate of 0.055). An optimized version was also introduced, which maps 234-bits to 128-bits in 6 DES operations and it processes 106-bit message-blocks (at the rate of 0.276). MD4 and MD5 were designed by Rivest [Schnr96, Menez97]. An Australian extension of MD5, known as HAVAL has also been proposed by Zheng, et al. [Zheng93]. The first published partial attack on MD4 was by den Boer and Bosselaers [Menez97] who demonstrated that collisions could be found when Round 1 (of the three) was omitted from the compression function, and confirmed unpublished work of Merkle showing that collision could be found (for input pairs differing in only 3-bits) in under a millisecond on a personal computer, if Round 3 was omitted. Another devastating attack on the full MD4, was due to Vaudenay [Vaude95], which provided only near-collisions, but allowed sets of inputs to be found for which, of the corresponding four 32-bit words, three are constants while the remaining word takes on all possible 32-bit values. Later, in 1995, Dobbertin [Dobbr96] broke MD4 by finding collisions for meaningful messages (in one hour, requiring 20 free bytes at the start of the messages). RIPEMD [Schnr96] was designed in 1992 by den Boer and others. However, early in 1995, Dobbertin [Menez97] demonstrated that if the first or last (parallel) round of the 3-round RIPEMD compress function is omitted, collisions can be found in 2^{31} compress function computations. This result coupled with concern about inherent limitations of 128-bit hash results motivated RIPEMD-160 by Dobbertin, Bosselaer and Preneel [Dobbr96]. NIST, along with the NSA, designed the Secure Hash Algorithm (SHA) for use with the Digital Signature Standard [NIST93]. SHA produces 160-bit hash, longer than MD5. The SHA is called secure, because it is designed to be computationally infeasible to recover a message corresponding to a given message digest, or to find two different messages which produce the same digest. Besides these, several other hash functions have also evolved during the past decade, such as [Dmgrd90, Daemn93a, Daemn93b, Schno91, Schno93]. However, as found in [Schnr96], most of these hashing schemes consume huge time and resource complexities, as a result of which their hardware implementation has become costly as well as complicated.

In this background, utilizing the regular, modular, cascadable and non-linear features of 3-neighbourhood additive CAs, a novel one-way hashing scheme has been designed. The mathematical foundation of the proposed scheme is provided by the *Lemmas 2.1, 2.2, 2.3, 2.4, 2.5 & 2.6* and *Theorems 2.1, 2.2, 2.3, 2.4 & 2.6* as found in *Chapter 2*. The next section describes the design of the proposed scheme.

3.2 CA based Hashing Scheme : CHA

The proposed hashing scheme is designed based on the properties of 3-neighbourhood additive *group* and *non-group* CAs. The scheme operates block-wise and it hashes arbitrary length messages into fixed r-bit hash value. Firstly, the input message is broken into r-bit chunks, to make exact multiple of r-bit blocks, 0's are padded at the end of the message. Four r-bit variables are initialized. Then the main loop of the algorithm begins. The loop continues for as many iterations as there are r-bit blocks in the message.

The hashing process is mainly performed in two major phases : in **Phase I**, for each original source block, an intermediate cipher-block is built, through a sequence of three operations; in **Phase II**, using the intermediate ciphertext-block along with the previous hash value, MACA based hashing is performed and a corresponding hash value is generated. These two steps for each block continue until all the blocks finish. The final hash value becomes the hash of the entire message.

Phase I : Intermediate Cipher Block Building

For each r-bit block of message b_i, which is further divided into four sub- blocks, each of $r/4$-bit length - the inner loop begins for four rounds. Each round consists of three operations - bit-wise substitution of $r/4$-bit sub-block using a BS (Bit Substitution) table of size $r/4$, XORing with a set of r-bit constants (δ_i's) (dynamically changed for each message during run-time) and finally enciphering each r-bit block using a PCA based Block Cipher. In the following sub-sections, each of these operations are discussed.

Bit-wise Substitution

In this phase, each of the input $r/4$-bit sub-blocks is bit-wise substituted according to a substitution table. During substitution, the table is scanned from left to right, top to bottom and each bit position in the input sub-block is substituted using the corresponding bit values from the table. An important property which is trivial for this operation is presented next.

Lemma 3.1 : *Let α_i and β_i be two $r/4$-bit plaintext messages then,* **BS** $(\alpha_i) =$ **BS**(β_i), *only if,* $\alpha_i = \beta_i$ *and where* **BS** *is the Bit-wise substitution.*

An example **BS** table is shown in *Table 3.1* for $(r/4 = 64)$. According to this table, bit-1 of the input block is permuted to bit-43, bit-17 is permuted to bit-32. The major advantage of this permutation is that its hardwired implementation is trivial and it improves the overall complexity of the proposed scheme. The substituted sub-block is concatenated with the rest of the three sub-blocks, to obtain β'. The next step of operation is initiated with β' as the input.

Table 3.1: Bit-wise Substitution Table

43	23	55	34	26	18	10	53	60	52	44	36	28	20	12	4
32	54	46	38	30	22	14	6	64	56	48	40	62	24	16	8
57	49	41	33	25	17	9	1	59	51	58	35	27	19	11	3
61	2	45	37	29	21	13	5	63	42	47	39	31	50	15	7

XORing With Constants

A set of four r-bit constants are chosen for each plaintext message. In this phase, during each round of operation, these four initialized constants are XORed with the block, β' case-wise as follows:

```
switch(round) { /* 'round' is initialized with 1 and incremented with each cycle upto 4 */
        case 1 : $\beta'' = \beta' \oplus \delta_1$;
        case 2 : $\beta'' = \beta' \oplus \delta_2$;
        case 3 : $\beta'' = \beta' \oplus \delta_3$;
        case 4 : $\beta'' = \beta' \oplus \delta_4$;
        }
```

The following important property is trivial from the above XORing operation -

Lemma 3.2 *Let α_i' and β_i' be two r-bits plaintext message blocks then, $\alpha_i'\oplus\delta_i = \beta_i'\oplus\delta_i$, only if, $\alpha_i' = \beta_i'$, where, δ_i is a distinct r-bit constant.*

The XORed output i.e. β'' is subjected to the next phase of operation i.e. enciphering using a Block cipher.

Enciphering Using PCA based Block Cipher

Each intermediate block, β'' (r-bit) is enciphered using one of the enciphering functions, E, where, $E \in {k_2}^8$, the space of enciphering functions with the exponential order of cardinality, a considerably large number [Palch97]. The enciphered block, β''' is yielded by acting E on β'', i.e. $\beta''' = E(\beta'')$. The enciphering function consists of q fundamental transformations and each of the transformations is derived from different CA rules. By permutation of q fundamental transformations, a total of $(q!)$ enciphering functions can be generated. For example, let $E = abcd$, with $q = 4$, where

$$a = {\pi_1}^4, b = {\pi_2}^4, c = {\pi_3}^4, d = {\pi_4}^4;$$

and ${\pi_i}^8 = I; i = 1, 2, 3, 4$; implying that a, b, c, d are involutions and π's are CA representation of various permutations. An important result of this enciphering is reported next.

Lemma 3.3 *Assume α_i'' and β_i'' be two r-bit plaintext message blocks, then $E(\alpha_i'') = E(\beta_i'')$; only if $\alpha_i'' = \beta_i''$ and $E(\alpha_i'')$ represents the enciphering function acted on α_i'' and $\alpha_i'' = (BS(\alpha_i') + \delta_i)$.*

The proof of the above *lemma* can be easily established from the 'non-maximum length' group CA properties found in [Nandi94a]. By rotating left the sub-blocks of β'''- above three operations are repeated for four rounds. Finally, the resultant block, β''' is concatenated with the previous hash value, h_{i-1} (r-bit) and then input to the final phase of hashing.

Phase II : Hashing Using Complemented MACA

The structure of the MACA (as described in *Chapter 2, Section 2*) should be selected in such a way that it generates enough number of basins(trees), each rooted on an attractor. From the properties of MACA, it can be found that if the characteristic matrix of a k-cell MACA is T, and $rank(T \oplus S_c) = r$, then the number of attractor states is 2^{k-r}. Based on some properties observed in complemented MACA (*Lemmas 2.7 & 2.8*) the following property of MACA is derived.

Lemma 3.4 *In an k-cell complemented MACA with 2^m attractors, there exists m-bit positions at which the attractors give pseudo-exhaustive 2^m patterns.*

Proof: *Let T be the characteristic matrix corresponding to an k-cell MACA, CA_1 with 2^m attractors. As per* lemmas 2.7 & 2.8, *noted above- the corresponding complemented MACA, $CA_1{}'$ derived from CA_1 has same number of attractors (cycles of length unity). Since, $CA_1{}'$ has 2^m attractors, we have*

$$\text{rank}(T \oplus I) = (k - m) \; ;$$

where, I is the $(k \times k)$ identity matrix. The attractors of $CA_1{}'$ will be governed by m-dimensional null space of $(T \oplus I)$. Hence, there exists m linearly independent basis vectors, i.e. all possible combination of these basis vectors will be present in the null space. Hence, of $CA_1{}'$, $(T \oplus I)$ produces pseudo-exhaustive bit patterns at m-bit positions.

□

In this phase, the proposed scheme uses a $2r$ bit Complemented MACA which induces 2^r number of basins, where each of them is rooted at an attractor (characterized by Pseudo-Exhaustive m-bit positions). During hashing, each block of intermediate enciphered message concatenated with the previous hash value and is loaded as a seed to the $2r$-bit non-group complemented MACA and it is allowed to run autonomously for a number of cycles equal to the depth to reach the attractor ($2r$-bit). The Pseudo-Exhaustive r-bits of the attractor are then separated out as the present hash value, h_i.

The whole process (phase I and II) repeats for the remaining set of blocks. The final hash value h_n is the hash of the entire message.

3.2.1 The Algorithm CHA

CHA uses the following variables :

n	:	total number of r-bit message blocks (after padding);
h_i	:	i^{th} hash values ;
$\beta_i, \beta_i', \beta_i'', \beta_i'''$	:	variables to hold intermediate values ;
$\eta_1, \eta_2, \eta_3, \eta_4$	:	variables to hold the $(r/4)$ bits sub-blocks ;
$\delta_1, \delta_2, \delta_3, \delta_4$	:	r-bit constants ;
M_i	:	$2r$-bit message input
ξ_i	:	i^{th} $2r$-bit attractor;
Y_1, Y_2	:	temporary variables to hold $r/2$-bit blocks;
PCA	:	r-bit null boundary hybrid CA ;
$MACA'$	:	the Complemented $2r$-Cell MACA ;
BS	:	Bit-wise Substitution table of size $r/4$-bit.

Input : Plaintext message stream divided into 'n' blocks (each of r-bit) ;
Output : r-bit hash value;

```
Step 1:  Initialise $r$-bit constants : $\delta_1, \delta_2, \delta_3$ & $\delta_4$;
Step 2:  $i \leftarrow 1; h_{i-1} \leftarrow 0$;
Step 3:  Read $r$-bit block $b_i$; decompose it into four sub-blocks $(\eta_1, \eta_2, \eta_3, \eta_4)$ of length $(r/4)$ bits ;
Step 4:  $j \leftarrow 1$ ;
Step 5:  Read right-most sub-block $\eta_1$ of $b_i$ and substitute it with the corresponding bit-pattern
         using the BS table; Concatenate the modified sub-block $\eta_1'$ with the rest other sub-block
         ($\eta_i$'s) in the same order to obtain the $r$-bit block $\beta_i'$;
Step 6:  Perform XOR operation between the pair $(\beta_i', \delta_j)$ to obtain $\beta_i''$;
Step 7:  Encipher $\beta_i''$ using a PCA-based block-cipher to obtain block, $\beta_i'''$;
Step 8:  Rotate left $\beta_i'''$ by $(r/4)$-bits;
Step 9:  $j \leftarrow j + 1$; If $j \leq 4$ goto step 5;
Step 10: Perform the following operations to prepare input for the $2r$-bit $MACA'$ :
             $Y_1 \leftarrow left(r/2)$ bits of $\beta_i'''$;
             $Y_2 \leftarrow right(r/2)$ bits of $\beta_i'''$;
             $M \leftarrow concat(Y_1, h_{i-1}, Y_2)$ ;
Step 11: Load $MACA'$ with $M$ and run upto the depth level to obtain attractor, $\xi_i$;
Step 12: Extract the PE $(r)$-bits from $\xi_i$ and store in $h_i$;
Step 13: $i \leftarrow i + 1$ ; If $i \leq n$ then goto step 3 ;
Step 14: Return $h_n$ as the final hash value.
```

The following block diagram (see *Figure 3.1*) shows the datapaths in the CHA algorithm. This diagram is basically representing the hashing process for a single block of message. The controller logic is not shown in the *Figure 3.1*.

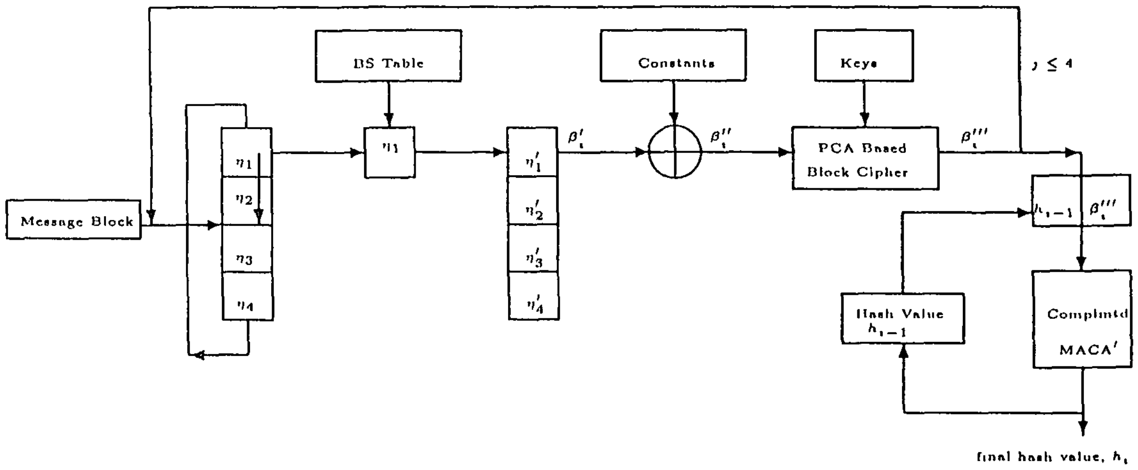


Figure 3.1: Logic diagram of the CHA (for a single block)

For an r-bit message, the proposed scheme generates a unique r-bit *message digest* or *hash value*. This uniqueness can be proved trivially based on the *lemmas 3.1, 3.2, 3.3 & 3.4* and hence, the following property can be derived.

Theorem 3.1 : *For any r-bit encrypted message block, concatenated with the previous r-bits hash value h_{i-1} (i.e.'r' PE-bits), as per* Lemmas 3.1, 3.2, 3.3 & 3.4, *the corresponding r-bit signature is unique.*

3.3 Invulnerability of the proposed scheme

The security of the scheme against the two possible types of attacks is described below :

A. *Hash-Value only Attacks :* For example, if $r = 128$, for a cryptanalyst it is essential to study 16 characters (each character made of 8 bits) frequency distribution table-which is of size $(2^8)^{16} = 2^{128}$, i.e. an enormous quantity. Hence, the scheme is guarded against cryptanalysts' *hash-value only attack.*

B. *Known & Chosen (Message, Hash-Value) Pair Attacks*: In the case of known plain-text attack, the intruder is assumed to possess a considerable amount of messages and corresponding hash values. While in case of chosen plaintext attack, the intruder is able to acquire an arbitrary number of messages and the corresponding hash value pairs (M, h) of his own choice.

In the proposed scheme, as per *Lemmas 3.1, 3.2, 3.3 & 3.4*, it can be found that each *Phase* and *Step* (within a phase) are independent to the others. Hence, the cracking complexity for the whole scheme will be the multiplication of the individual complexities offered by each of them. Next, we compute the complexities due to each of these steps, phase-wise :

- for an r-bit message, the bit-wise substitution offers complexity of the order of $O(2r/4)$;
- for XORing with r-bit constants, the complexity will be $O(2^r)$;
- In case of the PCA-based block ciphering, as it is based on the application of *Alternating Group*, which offers- in general, the complexity of the order of $(2^r!)/2$. However, the proposed block enciphering operation basically uses a subset of this alternating group (i.e. the *affine group*). So, the actual complexity will be less than it, i.e. may be of the order of $O(2^{r-1}!)$.

In Phase **I**, these three operations repeat for four times. Thus, the order of complexity due to first phase will be, approximately :

$$2^2 \times (2^{r/4}) \times (2^r) \times (2^{r-1}!) \approx (2^{2r});$$

Similarly, for the second phase,

- since, the MACA is basically a 3-neighbourhood additive CA, it uses $(6r-2)$ variables (i.e. the three major diagonals of the non-group CA). Thus, there will be basically 2^{6r-2} possible useful matrices among the whole set of 2^{r^2} matrices. From experimental study, it has been found that - for $r = 16$, and for 10100 random sample matrices of size (32×32), the number of MACAs is 2789, which is approximately, $(1/5)^{th}$ of the total sampled matrices taken.Thus, it can be assumed that- the possible number of MACAs can be more than 2^{4r} and hence, it will offer complexity approximately of the order of (2^{4r}).

- again, as the chosen MACA is complemented, due to the complement vector of say, size r, the complexity for this step will be $O(2^r)$.

So, Phase **II** will offer complexity of the order of :

$$(2^{4r}) \times (2^{r}) \approx (2^{5r}) \; ;$$

Hence, total complexity due to Phase **I** and Phase **II** will be, approximately :

$$(2^{5r}) \times (2^{2r}) \approx (2^{7r}) \; ;$$

In the preceeding section, the proposed one-way hashing scheme has been established and its security against all the possible attacks has also been reported. To justify the usefulness of the scheme, the next section discusses a potential application of the proposed scheme.

3.4 Application of the proposed scheme

In the present day world, one of the major applications of the one-way hashing scheme for message authentication is *electronic economy* or *ecash*, where a variety of cryptographic techniques are being used to minimise threats to electronic transactions. *Ecash* is an electronic payment system developed by the Digicash Co. of Amsterdam [Panur96]. It is currently being implemented by the Mark Twain Bank of Missouri in the U.S. In this payment system, to undertake transactions, both buyers and sellers must have deposits in the concerned implementor Bank's World Currency Access accounts. Access accounts are conventional checking accounts insured by an Insurance corporation, without paying interest or having a fixed maturity period.

As per the instruction of the Buyers, the Bank transfers funds from their World Currency Access accounts into their accounts' *Ecash Mint*. Funds in the mint are no longer deposits in the Bank, and they are not insured. The mint acts as a personal buffer account. At any time buyers can order their computers to remotely interface with the mint and withdraw funds from the mint into the hard drives on their personal computer. The format of the funds is digitized- and it is cryptographically secure and unique.

To solve the security problems in *Electronic Cash Transfer System*, basically, four mechanisms are used, such as privacy, authentication, integrity and scalability. In many cases, it has been observed that- more than one kind of these problems are mitigated by using a single mechanism [Baldu97]. *Privacy* includes the desire to keep documents and communication secrets, as well as to hide the very existence of certain kinds of information and to protect the identities of the parties communicating. *Authentication* and *integrity* refer to the need to confirm the identity of user, the authenticity of messages, and the integrity of messages or connections. *Scalability* mechanisms like distribution centers and digital certificates, are other crucial aspects to the success of electronic commerce systems. For, they help in creating systems that involve millions of users, transactions and documents. Among these various possible threats, authentication problems are serious, and so, special care is essential for designing the authentication scheme particularly from the message authentication point of view.

The most basic form of authentication is validating the identity of system users. One mechanism that offers substantially better user authentication is a credit-card-sized authentication device- a token or *smartcard-* that can store a secret key and perform a cryptographic challenge response. To access the system, the user must have the authentication token as well as a password. Once the users have been authenticated, the next problem is authenticating the individual messages. An electronic fund transfer system has to know that its instructions come from the expected source and have not been modified by an attacker. The core mechanism for achieving this kind of authentication is called a *one-way digest*. Cryptographic digests are one-way in the sense that it is easy to compute them, but computationally infeasible to find a message that has a given digest, that is computing the function is easier than computing its inverse. In this regard, the proposed **CHA** fulfills all these criteria. Tempering of any message will be easily detected with very high confidence in the proposed scheme. Any attacker cannot modify a message without changing the digest. Due to high exponential order of complexity, it is hardly possible for the intruder to compute the message, if the corresponding digest is given. Also, by combining one-way-digests and public-keys, one can achieve better results in authentication and integrity. By using public keys with the one-way digest, a digital signature is created, which is attached to the electronic messages or other business documents. The senders can sign a message by computing the digest and then encrypting it with their private keys. The receiver verifies that the message came from the specified receiver by decrypting the digest using the senders' public key and comparing that value with the digest he computed for the message. In *Figure 3.2*, this process is illustrated. If an intruder modifies the message, the digest will not match. If an attacker tries to modify the message and its encrypted digest, that too will fail, bacause the attacker does not know senders' private keys. In the following block diagram (see *Figure 3.2*), the role of CHA has been shown in the electronic economy.

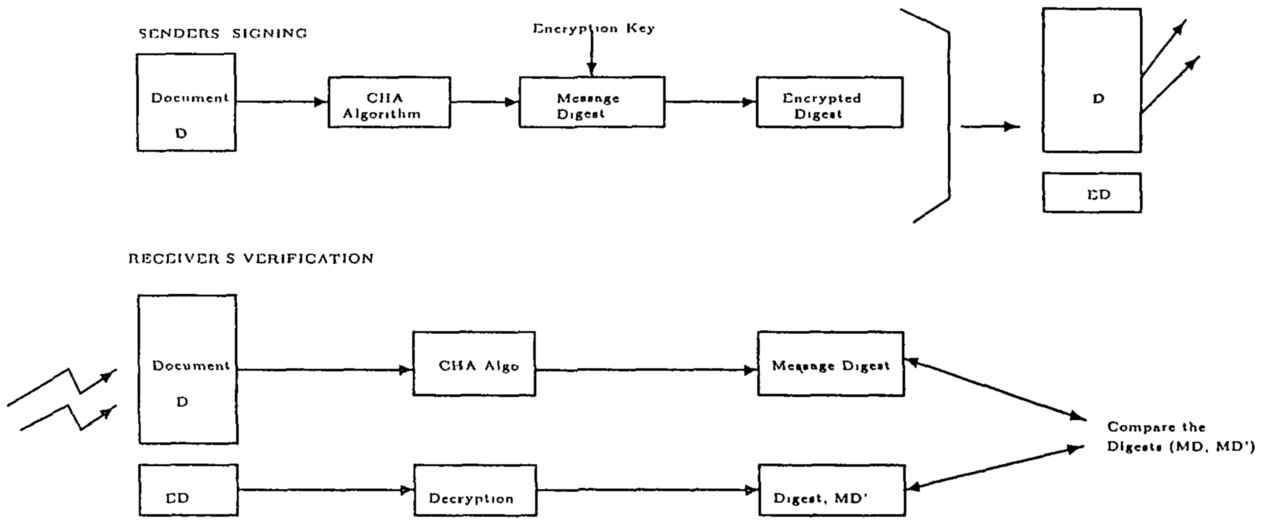


Figure 3.2: Sender's signing and receiver's verification process

3.5 Conclusions

In this chapter, a new scheme for message authentication purposes has been introduced. The scheme is based upon the properties of r-bit group and $2r$-bit *non-group, non-linear* CAs, which have been utilised to generate unique r-bit hash values. It has been established that a particular class of non-group CA (referred to as **MACA**) can serve as an efficient hashing function generator. The schemes also utilise Bit-wise permutation as well as PCA based block enciphering mechanisms- which make the schemes more significant from security point of view. The scheme offers an exponential order of cracking complexity and thus, protects it from all possible types of attacks. The complexity of the scheme can be further improved with an increase in block size.Another major advantage of this scheme is the use of simple, regular, modular and cascadable structure of CA as the basic building block that ideally suits for VLSI implementation.

In the next chapter *Chapter 4*, an application of n-neighbourhood CA in the context of an enciphering scheme design has been discussed.

Chapter 4

Design of ENCA based Cipher System

4.1 Introduction

In the previous chapter, it has been established that CA can be used as a useful tool for efficient one-way hash function design. This chapter explores the possibility of application of the extended neighbourhood CA (ENCA) in the area of cryptographic system design. Cryptography has become an essential requirement for ensuring communication privacy or concealment of data in a data bank. The process by which an unprotected message, i.e. the plaintext is transformed into ciphertext (or cryptogram) of an unintelligible form is called encryption or encipherment. Encryption may be achieved by constructing two different types of ciphers : stream and block cipher. In a *stream cipher*, the message is broken into successive bits or characters and then the string of characters is encrypted using a key stream. On the other hand, a block cipher is one in which a message is broken into successive blocks and then encrypted using single or multiple keys. Two most commonly desired characteristics of such cryptosystems are : (*i*) good measure of strength, and (*ii*) ease of implementation. However, a class of block-ciphers arising in computer privacy have been studied in [Denni82, Reupl90, Welsh88, Sebry89, Schnr96, Menez97]; it reveals that most of them have two common drawbacks: use of a large key, or need of a much larger ciphertext than the plaintext [Schnr96]. It also appears that the time and resource complexities of the existing schemes are significantly large, which make the hardware implementation more complicated. This chapter introduces the design of two simple CA-based cipher systems based upon the group properties of ENCA. The next subsection presents a brief survey on the existing block ciphering schemes.

4.1.1 Reviewing the existing schemes

In the early days, a transposition cipher was used to rearrange characters according to some specified scheme [Melln73]. A simple substitution cipher replaced each character of an ordered plaintext alphabet [Sinko66]. In those days messages were also encoded in musical symbols [Sam79]. A common method was a simple substitution of individual notes for letters. A homophonic substitution cipher maps each character of the plaintext alphabet into a set of ciphertext elements called homophones. Hammer [Palch97] has shown that it is possible to construct a high order homophonic cipher such that any intercepted ciphertext will decipher into more than one meaningful message under different keys. The development of Poly alphabetic ciphers began with Leon Battista Alberti, the father of western cryptography [Kahn67]. Most polyalphabetic ciphers are periodic substitution ciphers based on a period. A popular form of the periodic substitution cipher based on shifted alphabets is the **Vigenere Cipher** [Kahn67]. All the preceding substitution ciphers encipher a single letter of plaintext at a time. By enciphering larger blocks of letters, polygram substitution cipher makes cryptanalysis harder by destroying the significance of single-letter frequencies. The **Playfair cipher** is a diagram substitution cipher named after the English scientist Lyon Playfair; the cipher was actually invented in 1854 by Playfair's friend, Charles Wheatstone, and was used by the British during World War I [Kahn67].

A **product cipher** is the composition of a set of functions (ciphers), where each function may be a substitution or a transposition. Shannon [Shann49] proposed composing different kinds of functions to create "mixing transformation", which randomly distribute the meaningful messages uniformly over the set of all possible ciphertext messages The LUCIFER cipher, designed at IBM by Feistel [Feist73], uses a transformation that alternately applies substitutions and transposition. In 1977 the National Bureau of Standards (USA) announced a data encryption standard (DES) to be used in unclassified U S Govt applications [DES67]. The encryption algorithm was developed at IBM and was the outgrowth of LUCIFER [Palch97]

In 1978, Pohling and Hellman [Pohln78] published an encryption scheme based on computing exponentials over a finite field At about the same time, Rivest, Shamir and Adleman [Rivst76] published a similar scheme, but with a slight modification that gave the MIT group a method for realizing public-key encryption as put forth by Diffie and Hellman [Diffi76a]. In 1979, Shamir [Shamr79] studied the feasibility of constructing a Knapsack system for both secrecy and authentication. Merkle and Hellman [Merkl79] proposed a scheme whose security depends on the difficulty of solving the 0-1 Knapsack problem. They show how to convert a simple Knapsack into a trapdoor Knapsack that is hard to solve without additional information. Following them, several new enciphering schemes have been proposed in [Denni82, Reupl86, Welsh88, Sebry89, Schnr96, Menez97, Palch97]. The present block cipher systems are also particular cases of these schemes. The proposed schemes can be found to be attractive and significant in view of the following facts :

1: The security of the schemes is mainly based on the CA (i.e. the binary matrices) used. As the CAs are not easily characterizable and as the size of the matrix spaces tremendously increases with the increase in size of the CAs, the measure of strength against intrusion can be found to be at least comparable to, if not better than the existing schemes.

2 : Encipherment and decipherment proceed with the similar protocols.

3 : Due to the simplified logic structure, the scheme can be implemented with minimum hardware.

The next Section presents the design of the proposed ENCA based cipher system.

4.2 Design of the Cipher System

The proposed enciphering scheme is designed by utilizing the periodicity properties exhibited by the group ENCAs. The *Theorems 2.13, 2.14 & 2.15* and *Lemma 2.9* basically provide the mathematical foundation of the scheme.

A group ENCA T of size s and a complement vector S_c, of same size are selected, which generates say, p number of cycles, i.e. $B_1, B_2, \cdots, B_p$. In the state transition diagram of the cycle-structure, every state with a repeated multiplication of T and exclusive ORing with S_c, ultimately ends up in one of p different cycles. These cycle-structu induced by T and S_c can be utilized for encryption of any binary file having not more than p distinct r-bit blocks. During encryption, for each *distinct* r-bit block, a *unique* cycle is identified and is mapped to any element taken at random (with equal chance, for example) from this cycle. The decryption process also uses the same CAs. By a repeated application of T and S_c on the ciphertext blocks, one will get the smallest element in the cycle, which is uniquely associated with a plaintext block and hence decrypted.

Algorithm Encrypt I;

Input to the algorithm is a plaintext message stream $b_1 b_2 \cdots b_n$, where each b_i refers to a r-bit block in the plaintext language. Let there be p distinct blocks in the language, where, $p \leq 2^r$.

Define an operator matrix T of size $s \times s$ $(s > r)$ and a complement vector S_c of size s; by a repeated application of the operators (i.e. T and S_c) will generate at least p cycles of equal or various length.

Input : Plaintext message stream $b_1 b_2 \cdots b_n$, where there are 'p' distinct possible $b_i s$.
Output : Ciphertext stream $c_1 c_2 \cdots c_n$, where each c_i is an s-bit vector.

```
Step 1 :  $i = 1$;
Step 2 :  Read $r$-bit message block $b_i$;
          Select the corresponding cycle uniquely associated with $b_i$;
          Choose an element '$m$' at random from the set of elements of the
          corresponding cycle and substitute $c_i$.
          $c_i \leftarrow m$ ;
Step 3 :  $i \leftarrow i + 1$ ;
          if $i \leq n$ goto Step 1 ;
Step 4 :  Stop execution.
```

Algorithm Decrypt I

We will have a table of length 'p', where entries of the table are the least element (x) of cycle and the corresponding b_i.

Input : The ciphertext $c_1 c_2 \cdots c_n$.

Output : The deciphered text $a_1 a_2 \cdots a_n$

```
Step 1 :  $i = 1$;
          Apply $T$ and $S_c$ on $c_i$ repeatedly until it will form a loop;
          Select the least element $x$ among those generated by the above process.
Step 2 :  Assign the corresponding entry from the table to $a_i$ ;
Step 3 :  $i \leftarrow i + 1$;
          if $i \leq n$ goto Step 1;
Step 4 :  Stop execution.
```

The Encrypt I and Decrypt I procedures are illustrated in the *Example 4.1.*

Example 4.1 Consider the CA T and the Complement Vector S_c as below

$$T = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}, \qquad S_c = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

The CA and the complement vector are of size $s = 7$ and has 15 cycles as listed below: Since T is non-singular, it will generate fully periodic cycle structure, where cycles are represented by the numbers $1, 2, 3, 6, 8, 9, \cdots$ etc. which are the *smallest* in the corresponding cycles.

$B_1 = \{$ **1**, 106, 120, 65, 121, 23, 108, 63, 67, 1, ... $\}$;
$B_2 = \{$ **2**, 68, 16, 125, 40, 19, 83, 64, 47, 2, ... $\}$;
$B_3 = \{$ **3**, 18, 5, 85, 7, 45, 122, 57, 4, 3, ... $\}$;
$B_4 = \{$ **6**, 123, 111, 17, 43, 61, 59, 124, 126, 6, ... $\}$;
$B_5 = \{$ **8**, 89, 93, 98, 29, 113, 114, 92, 52, 8, ... $\}$;
$B_6 = \{$ **9**, 15, 72, 74, 50, 79, 91, 37, 31, 9, . . $\}$;
$B_7 = \{$ **10**, 33, 32, 118, 99, 75, 100, 90, 115, 10, ... $\}$;
$B_8 = \{$ **11**, 119, 53, 94, 76, 117, 77, 35, 88, 11, ... $\}$;
$B_9 = \{$ **12**, 102, 34, 14, 30, 95, 26, 96, 101, 12, ...$\}$;
$B_{10} = \{$ **13**, 48, 55, 38, 49, 97, 51, 25, 78, 13, ... $\}$;

B_{11} = { 20, 66, 87, 127, 80, 110, 71, 62, 21, 20, ... };
B_{12} = { 22, 58, 42, 107, 46, 84, 81, 56, 82, 22, ...};
B_{13} = { 27, 54, 112, 36, 73, 28, 39, 103, 116, 27, ...};
B_{14} = { 24 } ; B_{15} = { 0 } ;

The cycle B_1, for instance stands for the cycle $1 \rightarrow 106 \rightarrow 120 \rightarrow \cdots$ i.e. $T \cdot 1 \oplus S_c = 106, T \cdot 106 \oplus S_c = 120, \cdots$ where $1, 106, 120, 65, \cdots$ are integers representing the values (000001), (1101010) $\cdots$ etc. Let us consider that the plaintext language consists of 12 unique r-bit blocks (e.g. $\delta_1, \delta_2, \cdots \delta_{12}$). Now, as the cycles from B_{14} onwards have only cycles of length 1 and as according to the proposed scheme, this plaintext will require only 12 cycles, so, B_{13}, B_{14} and B_{15} are not used here. The mapping of the blocks to their corresponding cycles is as given in the *Table 4.1*. Let an example

Table 4.1: Mapping r-bit block to corresponding cycle

Block	Basin	Representative	Period
δ_1	B_8	11	9
δ_2	B_4	6	9
δ_3	B_2	2	9
δ_4	B_9	12	9
δ_5	B_1	1	9
δ_6	B_{11}	20	9
δ_7	B_{12}	22	9
δ_8	B_7	10	9
δ_9	B_{10}	13	9
δ_{10}	B_3	3	9
δ_{11}	B_5	8	9
δ_{12}	B_6	9	9

sequence of plaintext blocks be : $\delta_3\delta_4\delta_4\delta_2\delta_5 \cdots$.
The cycle representatives are :$B_2(2)B_9(12)B_9(12)B_4(6)B_1(1) \cdots$.
For instance, δ_3 is associated with cycle B_2 which is represented by the 6-bit pattern for the number 2 . Hence, encryption for δ_3 is $T^q.(2) \oplus S_c$, where q is random over $0 \cdots 8$. Thus, if $q = 4$, $\delta_3 \rightarrow 40$. Similarly, for the second and third characters, $\delta_4 \rightarrow T^{q1}.(12) \oplus S_c$ and $\delta_4 \rightarrow T^{q2}\ (12) \oplus S_c$ respectively; if $q1 = 4$ and $q2 = 2$, the same plaintext block δ_4 will be mapped to 30 and 34 respectively. Thus, the encrypted message may be - $40, 30, 34, 111, 23, \cdots$etc.

For decryption, one starts with 40, which is not a *representative* i.e. smallest in any cycle as seen from *Table 4.1*. Hence, it is repeatedly multiplied by T and exclusive-ORed with S_c giving the sequence - 40, 19, 83, 64, 47, 2, 68, 16, 125 and back to 40, the smallest element being 2, which is the representative of cycle B_2 and hence, for message block δ_3 of plaintext The next encrypted block value is 30 which is again not in the list of representatives and hence, is also operated repeatedly by T and S_c giving the sequence 30, 95, 26, 96, 101, 12, 102, 34, 14 and back to 30. The smallest number in this sequence being 12, (representing B_9) it stands for δ_4 of the plaintext. The third encrypted block is leading to the cycle $34, 14, 30, \cdots, 12, 102$ and back to 34 and hence leading to again plaintext block δ_4 and so on. Thus, for instance, the same plaintext block δ_4 is mapped sometimes to 30 and sometimes to 34, but is still recoverable as cycle B_9, and hence, block δ_4, by using the same keys i.e. T and S_c. □

With this scheme, one disadvantage is that- presently, the sets of cycle elements are totally *disjoint* or *mutually exclusive* and as a result, among the plaintext and the ciphertext blocks - only a one-to-many mapping becomes possible. But, to achieve perfect security in a cryptosystem, a necessary condition is that there must exist many-to-many relationships among the plaintext and ciphertext messages [Shann49]. In two ways, this limitation could be overcome :

- by use of a binary matrix transformation and
- by use of a bit-permutation table.

In the next, both these mechanisms have been described as additional processing steps to the aforesaid scheme, which lead to two different block enciphering schemes. Due to these additional steps, in each of the schemes, the cycle elements are overlapped in a random manner, and as a result, it becomes possible for the same block to map into the different blocks at the same time and hence, the same input block can map to different cycles also.

[A] Binary Matrix Transformation Scheme :

A permutation π of a finite set V can be defined to be an injection, $\pi : V \rightarrow V$, where, $V = \{u_1, u_2, \cdots, u_d\}$. The set of all possible permutations on V basically form a non-commutative group G_d of order $(d!)$ under the operation of permutation multiplication. One distinct feature of permutation is its cyclic structure, i.e. for any permutation π in V, one can have $\pi(z_j) = z_{j+1}$, where, $j = 1, 2, \cdots (l-1)$ for a cycle $(z_1, z_2, \cdots z_l)$ of length l. It has been established that every permutation can be uniquely expressed as a product of disjoint cycles. By a transposition, it is meant a cycle of length two. An even permutation is one that is expressible as a product of an even number of transpositions; otherwise a permutation is called an odd permutation. It is a well known result that all even permutation on V form a normal subgroup G_d, which is the alternating group, say A_d of degree d and order $(d!)/2$.

Now, considering a CA, say T_1, which through a repeated application upon the elements of a finite set, say, F over GF(2), ultimately ends in one of 'p' different cycles. Now, every permutation can be uniquely expressed as a product of these 'p' disjoint cycles. Let us now consider an 'r'-dimensional vector space V_r, which consists of all the message blocks, each of 'r'-bits long. According to this modified scheme (which operates in two steps), each 'r'-bits message block 'b_i' is subjected to a permutation operation in the first step, to generate the intermediate block '$\bar{b_i}$', which is finally substituted using the ENCA based block cipher (i.e. **Encrypt I**) to ultimately generate the final cipher-block c_i. Now, let Υ be an enciphering function and $b_i \in V_r$ be a r-bit message block to be encrypted. Here, $\Upsilon \in V_{2^r}$, the space of enciphering function with cardinality $(2^r!)$. Thus, according to this scheme, the encrypted message block will be : $\bar{b_i} \leftarrow \Upsilon(b_i)$. Basically, each of these functions i.e. Υ consists of 'g' fundamental transformations [Nandi94a] and by using permutation among 'g' transformations, one can have $(g!)$ permutation functions. For the sake of clarity, let us take an example, say $\Upsilon = k_1k_2k_3k_4$, where -

$$k_1 = \pi_1^y; k_2 = \pi_2^y; k_3 = \pi_3^y; k_4 = \pi_4^y$$

here, π_1, π_2, π_3 and π_4 are permutation representation of four disjoint cycles of say, length l, induced by a group ENCA and 'y' represents the 'cycle-length'. Now, according to the scheme, the intermediate cipher-block is obtained by applying Υ on each of the plaintext blocks, i.e.

$$\bar{b_i} = \Upsilon(b_i) = (k_1k_2k_3k_4)(b_i)$$

Similarly, during deciphering, the inverse permutation will be applied :

$$b_i = \Upsilon^{-1}(\bar{b_i}) = (k_1k_2k_3k_4)^{-1}(\bar{b_i}) = k_4{}^{-1}k_3{}^{-1}k_2{}^{-1}k_1{}^{-1}(\bar{b_i}) = (k_4k_3k_2k_1)(\bar{b_i})$$

Thus, in the deciphering process, the fundamental transformations are applied just in the reverse order on the intermediate block '$\bar{b_i}$'. Here, $y < r$ and $\pi^r{}_i = I$, $i = 1, 2, 3, 4$ and k_i are involutions. The above logic is illustrated in *Example 4.2*:

Example 4.2 For clear understanding of the permutation operation, consider the following two permutation functions, π_1 and π_2 representing two disjoint cyclic structures, generated by a CA, say T of size '4' :

$$\pi_1 : \begin{bmatrix} 0 & 5 & 6 & 7 & 8 & 13 & 14 & 15 \\ 15 & 0 & 5 & 6 & 7 & 8 & 13 & 14 \end{bmatrix}, \qquad \pi_2 : \begin{bmatrix} 1 & 2 & 3 & 4 & 9 & 11 & 12 & 10 \\ 12 & 9 & 10 & 3 & 4 & 2 & 11 & 1 \end{bmatrix}$$

Now, ${\pi_1}^3(6) = 15$; ${\pi_1}^6(14) = 0$; Similarly, ${\pi_2}^4(3) = 11$; ${\pi_2}^5(9) = 12$;
Now, consider that an example sequence of plaintext blocks be : $\delta_9\delta_6\delta_2\delta_{12}\delta_9\cdots$.
During enciphering, for instance, δ_9 is associated with table P_1 (here, each table P_j corresponds to a permutation, say π_j in implementation) and after bit-substitution it becomes $P_1(\delta_9) \rightarrow \delta_3$. Similarly, for the second and the third blocks i.e. δ_6 and δ_2, let it be $P_2(\delta_6) \rightarrow \delta_4$ and $P_3(\delta_2) \rightarrow \delta_4$ respectively. Again, for the fifth block i.e. δ_9, let the corresponding table be P_j, where $j = 0\cdots q$. After substitution, let it becomes $P_j(\delta_9) \rightarrow \delta_5$. Thus, the substituted sequence of blocks will be : $\delta_3\delta_4\delta_4\delta_2\delta_5\cdots$.

Similarly, for the inverse substitution, each substituted block is picked up and subjected for inverse substitution. For instance, δ_3 is taken and substituted with table $\overline{P_1}$, which will result - $\overline{P_1}(\delta_3) \rightarrow \delta_9$. Similarly, for the second and the third blocks i.e. δ_4 and δ_4, it will be $\overline{P_2}(\delta_4) \rightarrow \delta_6$ and $\overline{P_3}(\delta_4) \rightarrow \delta_2$ respectively. Similarly, for the remaining other blocks also, the corresponding inverse tables will be applied and finally, one can easily obtain the original plaintext block sequences. □

[B] Bit-Permutation Scheme :

This scheme uses a set of bit-wise permutation tables : $P_1, P_2, \cdots P_q$, (analogous to that found in *DES* [DES67], where 'q' is the cardinality of the set. During encryption, each 'r'-bit plaintext block b_i is associated with a Permutation table say, P_j and operated by it to obtain the substituted r-bit block $\overline{b_i}$, which is finally used as an input to the previous encryption algorithm i.e. **Encrypt I**). Similarly, for deciphering also, it uses a set of corresponding *Inverse Permutation* tables: $\overline{P_1}, \overline{P_2}, \cdots \overline{P_q}$. The first step in the deciphering is analogous to the previous decryption algorithm (i.e. **Decrypt I**); it identifies the smallest element in the cycle associated with the ciphertext block c_i and then through a simple table lookup, identifies the corresponding intermediate cipher-block, i.e. $\overline{b_i}$. In the *second* step, the corresponding inverse permutation table is applied on $\overline{b_i}$ to obtain the original plaintext block b_i.

To illustrate the above procedure, the *Example 4.3* is cited.

Example 4.3 Let us take some simple *Bit-Permutation* and its corresponding *Inverse* permutation tables, such as :

$P_1 = [\ 3\ 2\ 4\ 1\ ]\ ;\ \overline{P_1} = [\ 4\ 2\ 1\ 3\ ]\ ;$
$P_2 = [\ 4\ 2\ 1\ 3\ ]\ ;\ \overline{P_2} = [\ 3\ 2\ 3\ 1\ ]\ ;$
$P_3 = [\ 2\ 1\ 4\ 3\ ]\ ;\ \overline{P_3} = [\ 2\ 1\ 4\ 3\ ]\ ;$
....

Now, consider that an example sequence of plaintext blocks be : $\delta_9\delta_6\delta_2\delta_{12}\delta_9\cdots$. During enciphering, for instance, δ_9 is associated with table P_1 and let after bit-substitution it becomes $P_1(\delta_9) \longrightarrow \delta_3$. Similarly, for the second and the third blocks i.e. δ_6 and δ_2, let it be $P_2(\delta_6) \longrightarrow \delta_4$ and $P_3(\delta_2) \longrightarrow \delta_4$ respectively. Again, for the fifth block i.e. δ_9, let the corresponding table be P_j, where $j = 0\cdots\kappa$. After substitution, let it becomes $P_j(\delta_9) \longrightarrow \delta_5$. Thus, the intermediate sequence of input blocks may be : $\delta_3\delta_4\delta_4\delta_2\delta_5\cdots$.

Similarly, for the reverse permutation, each permuted block is picked up and subjected for inverse permutation. For instance, δ_3 is taken and substituted with table $\overline{P_1}$, which will result - $\overline{P_1}(\delta_3) \longrightarrow \delta_9$. Similarly, for the second and the third blocks i.e. two consecutive δ_4s, it will be $\overline{P_2}(\delta_4) \longrightarrow \delta_6$ and $\overline{P_3}(\delta_4) \longrightarrow \delta_2$ respectively. Similarly, for the remaining other blocks also, the corresponding inverse tables will be applied and finally, one can easily obtain the original plaintext block sequences. □

From such bit-substitution, one can easily observe that the different blocks are sometimes mapped into the same block and also the same block is mapped into different blocks. So, basically it establishes a many-to-many mapping. By increasing the size of the permutation table, one can have more varieties in mapping among the blocks.

The algorithms *Encrypt II* and *Decrypt II* presented below are enhanced versions of the previous algorithms i.e. *Encrypt I* and *Decrypt I*. For encryption, *Encrypt II* uses a set of permutation tables : $P_1, P_2, \cdots P_\kappa$, ('$\kappa$' is the cardinality of the set) induced by any of the scheme reported so far (i.e. either binary-matrix transformation or, bit-permutation), where each table is uniquely associated with any of the distinct 'r'-bit plaintext block b_i. Similarly, *Decrypt II* also uses a set of corresponding inverse tables, i.e. $\overline{P_1}, \overline{P_2}, \cdots \overline{P_\kappa}$ (due to the corresponding scheme), where each of them is uniquely associated with any of the intermediate cipher-block. *Encrypt II* operates on each of the plaintext block b_i and substitutes it with the permutation table, say P_j (where $j = 1, 2, \cdots \kappa$) to obtain intermediate cipher-block, $\overline{b_i}$. The next step of encryption is analogous to *Encrypt I*. Similarly, the first step of *Decrypt II* is analogous to the previous decryption algorithm (i.e. *Decrypt I*); it identifies the smallest element in the cycle associated with the ciphertext block c_i and then through a simple table lookup, it identifies the corresponding $\overline{b_i}$. In the *second* step, the corresponding inverse permutation table is applied on $\overline{b_i}$ to obtain the original plaintext block b_i.

Algorithm Encrypt II ;

Input to the algorithm is a stream of plaintext message blocks $b_1 b_2 \cdots b_n$, where, each b_i consists of 'r'-bits. Let there are be 'p' distinct blocks in the language, where $p \leq 2^r$.

Define a set of Permutation tables of cardinality κ : $P_1, P_2, \cdots P_\kappa$, induced by either *bit-permutation* or *matrix transformation* scheme, where each table is of size r-bits. During enciphering, each b_i will be associated with a P_j (where, $j \leq \kappa$) and substituted as r-bit $\overline{b_i}$.

Define a matrix T of Size $s \times s$ and a complement vector S_c of same size which will generate at least p number of cycles, each of which will correspond to p distinct r-bit intermediate cipher-block $\overline{b_i}$'s.

input : Plaintext message blocks $b_1 b_2 \cdots b_n$, where each b_i is of r-bits.
Output : Ciphertext stream $c_1 c_2 \cdots c_n$, where each c_i consists of s-bits.

```
Step 1 :  $i = 1$;
Step 2 :  $j = 1$;
Step 3 :  Read a $r$-bit block, $b_i$;
          Substitute $b_i$ using the corresponding $P_j$ to obtain $\overline{b_i}$, i.e.
          $\overline{b_i} \leftarrow P_j(b_i)$;
Step 4 :  Call Encrypt I with $\overline{b_i}$ as input;
```

```
          select an element 'm' randomly from the corresponding cycle of $\overline{b_i}$
          and substitute it, i.e. $c_i \leftarrow m$ ;
Step 5 :  $j \leftarrow j + 1$; $i \leftarrow i + 1$;
          if $i \leq n$ and $j \leq \kappa$ then goto step 3;
          if $i \leq n$ and $j > \kappa$ then goto step 2;
Step 6 :  Stop execution.
```

Algorithm Decrypt II

We will have a table of length p, where entries of the table are the least element (x) of cycles and corresponding $\overline{b_i}$.

Define a set of corresponding *Inverse* Permutation tables : $\overline{P_1}, \overline{P_2} \cdots \overline{P_\kappa}$; .

Input : The ciphertext $c_1 c_2 \cdots c_n$, where each c_i is of s-bits.
Output : The deciphered text $b_1 b_2 \cdots b_n$, where each b_i is of r-bits.

```
Step 1 :  $i = 1$;
Step 2 :  $j = 1$;
Step 3 :  Call Decrypt I with $c_i$ as the input;
          find the least element 'x' of the corresponding cycle associated with $c_i$
          and assign to $\overline{b_i}$;
Step 4 :  Apply the corresponding Inverse table $\overline{P_j}$ on $\overline{b_i}$ -
          $b_i \leftarrow \overline{P_j}(\overline{b_i})$;
Step 5 :  $i \leftarrow i + 1$; $j \leftarrow j + 1$;
          if $i \leq n$ and $j \leq \kappa$ goto Step 3;
          if $i \leq n$ and $j > \kappa$ goto step 2;
Step 8 :  Stop execution.
```

For better understanding of the enhanced version of the proposed scheme, *Example 4.4* is cited.

Example 4.4 Let the number of distinct plaintext blocks 'p' be 12 and the blocks be · $\delta_1, \delta_2, \cdots \delta_{12}$; and let each block consist of 4 bits. Now, let us define a set of bit-permutation tables $P_1, P_2, \cdots P_\kappa$ induced by either the *binary-matrix transformation* or the *bit-permutation* scheme, where, the cardinality of the set, $\kappa \leq p$ and each P_j is of size 4. During enciphering, each block of message (say δ_i) is subjected to a permutation table P_j (chosen according to a sequence maintained) and the corresponding $\overline{\delta_i}$ is generated. The next step of encryption with $\overline{\delta_i}$ is similar with that found in *Example 4.1.* As found in the previous example, select two key operators i.e. say T_1 and S_{c1} such that it generates at least $p = 12$ cycles. Each of the 12 $\overline{\delta_i}$s is associated with any of the 12 cycles as shown in *Table 4 1.*

Now, consider that an example sequence of plaintext blocks be : $\delta_9\delta_6\delta_2\delta_{12}\delta_9\cdots$.
During enciphering, for instance, δ_9 is associated with table P_1 and suppose after bit-substitution it becomes $P_1(\delta_9) \rightarrow \delta_3$. Similarly, for the second and the third blocks i e. δ_6 and δ_2, let it be $P_2(\delta_6) \rightarrow \delta_4$ and $P_3(\delta_2) \rightarrow \delta_4$ repectively. Again, for the fifth block i.e. δ_9, let the corresponding table be P_j, where $j = 0\cdots\kappa$. After substitution, let it becomes $P_j(\delta_9) \rightarrow \delta_5$. Thus, the intermediate sequence of input blocks become : $\delta_3\delta_4\delta_4\delta_2\delta_5\cdots$.

With this sequence of input blocks, the next phase of enciphering will be similar as found in *Example 4 1.*

During decryption, the first step is similar with the deciphering operation that found in *Example 4.1.* It unambiguously identifies the cycle B_i for each cipher-block and hence also the corresponding δ_i through a simple table look-up. Thus, through certain repetition one can recover the entire sequence of the intermediate blocks i.e.

$$\delta_3\delta_4\delta_4\delta_2\delta_5\cdots$$

Nextly, one starts with the second step of decryption. Each intermediate block is picked up and subjected to inverse substitution. For instance, δ_3 is taken and substituted with table $\overline{P_1}$, which will result - $\overline{P_1}(\delta_3) \rightarrow \delta_9$. Similarly, for the second and the third blocks i.e. δ_4 and again δ_4, it will be $\overline{P_2}(\delta_4) \rightarrow \delta_6$ and $\overline{P_3}(\delta_4) \rightarrow \delta_2$ respectively. Similarly, for the remaining other blocks also, the corresponding inverse tables will be applied and finally, one can easily obtain the original plaintext block sequences, i.e. $\delta_9\delta_6\delta_2\delta_{12}\delta_9\cdots$
$\square$

Implementation of the proposed scheme : From the *hardware implementation* point of view, it can be found that as the sets are finite only and of only moderate size, one need not have a table stored in memory for the table look-up referred above. Only the matrix T of order $s \times s$ and the complement vector S_c of size s, which are the keys, are to be stored in memory, each row of the matrix itself being stored as a binary number. For instance, the matrix T used in the illustrative example can be stored as the sequence of six numbers $\{59, 14, 71, 38, 13, 101, 92\}$ written in binary form. In fact, relevant matrix multiplication can obviously be carried out as suitably formulated operations on individual bits, a possible advantage in respect of hardware implementations. Software can be so constructed that when the task of encryption or decryption is at hand, one is only to feed the key numbers viz. the matrix rows as 'numbers' and the software constructs the tables for mapping as well as cycle identifiers along with their cycle lengths.

From the *software implementation* point of view, it has been found that- it is not necessary to use explicitly the CA multiplication and exclusive ORing- carried out on each ciphertext block. At the start of encryption or decryption, one can generate the table or inverse table for mapping encrypted blocks to plaintext as a single subscript array and carry out decryption by table lookup.

4.3 Invulnerability of the Schemes

Security of the proposed schemes against possible attacks are discussed below :
(a) Ciphertext only attack : According to the schemes, as among the plaintext and ciphertext blocks, a *many-to-many* mapping exists, the schemes may be considered to be guarded against crypt-analyst's ciphertext only attack.
(b) Known and Chosen Plaintext attack: In case of known plaintext attack, the intruder is assumed to possess a considerable length of plaintext and the corresponding ciphertext. While in case of chosen plaintext attack, the intruder is able to acquire an arbitrary pairs of message and corresponding ciphertexts i.e. (M, C) of his own choosing.

In case of both the schemes, the encryption is basically performed in two major steps and each step is independent of the other. So, the intruders' complexity will be the product of the individual complexities offered by each step. One can compute the cracking complexity in the following manner :

- in step I, for an r-bit message block, due to the permutation operation, the possible order of complexity is : $O(2^r!)$;

- in step II, the complexity offered is due to the key-space generated by the CA used;

if the size of the CA is 's', in general, the size of the key-space is $O(2^{s^2})$. However, the scheme uses only those CAs which exhibit *ultimate periodicity* properties and generate certain minimum number of cycles. Thus, the actual size of the key-space should be less than the aforesaid size. From our exhaustive experimentation, based on sample data generated by random sampling it has been observed that for ($s = 6$), the proportion of matrices with at least 8 cycles is around 20 % and for ($s = 8$) the proportion appears to be 21 percent; similarly, for ($s = 7$), ($s = 9$) and ($s = 10$), it is reported in *Table 4.2*. Thus the complexity may be higher than $O(2^{s^2/2})$; Lastly, due to the complement operation using the complement vector S_c, of size 's', the possible complexity is of order $O(2^s)$;

Hence, the total approximated complexity due to step **I** & step **II** may be of the order of :

$$O(2^r!) \times O(2^{s^2/2}) \times O(2^s)$$

- an extremely large number!

Table 4.2: Finding 'm' for different CAs of size 's'

No. of Sample Matrices M	Size of the Operator T				
	$s = 6$	$s = 7$	$s = 8$	$s = 9$	$s = 10$
10	2	2	2	3	2
100	20	27	21	31	25
1000	197	214	219	259	267
10000	2108	2179	2203	2574	2682

4.4 Conclusion

Two simple, but invulnerable block ciphering schemes have been presented in this chapter. The schemes are based on the periodicity properties of a finite set of elements over the field of GF(2). The encipherment and decipherment procedures of the schemes follow the similar protocols. Due to the non-linear operations and many-to-many mapping, the measure of strength of the proposed schemes against intrusion can be found to be comparable, if not better, than the existing schemes. There are further scopes to improve the complexity by increasing the size of the message block as well as the CAs. The simplified logic structure makes the schemes more efficient from the implementation point of view.

In the modern cryptography, *authenticated key exchange* scheme has an important role in providing security over an insecure channel. In such scheme, one-way hash functions and cipher systems are used as the basic building blocks for authentication as well as data encryption purposes. A new *password-only authenticated key-exchange scheme*, has been developed based on the schemes designed and presented so far (i.e. in *Chapter 3* and in the *present*), which will be discussed in the next Chapter.

Chapter 5

CA based Password Authenticated Key Exchange

5.1 Introduction

The theory of CA and its applications in the area of data security and message authentication have been discussed in detail in the previous chapters (namely, *Chapters 2, 3 & 4*). It has been observed that it is possible to apply CAs and the CA based schemes reported so far, in the area of authenticated key exchange applications. In this chapter, an attempt has been made to explore such possibilities in password-only authenticated key exchange systems. Password authenticated key exchange schemes (PAKE) are very important for strong authentication over an insecure channel. Designing an efficient password scheme over an insecure network has been a challenging problem, particularly in the light of dictionary attack. There is an increasing volume of work focussed on the password problem in the last few years and several novel solutions have been produced [Schnr96, Menez97, Jabln96, Belvn92, Diffi92]. From crypt-analysis point of view, large passwords are always preferable, whereas for ordinary people, it is difficult to remember them. So, construction of a strong remote user authentication scheme using only a small password is always encouraging. For user authentication purpose, though *smart-cards* are more convenient, there is always added advantages with a password-only authentication scheme, because *firstly*, it is less expensive and *secondly*, it is more resistant to theft. This chapter presents a new password-only authenticated key exchange scheme, designed by utilizing the features of group and non-group Cellular Automata (CA) [Palch97]. It uses the results reported in the earlier chapters (namely *Chapters 2, 3 & 4*). Due to the simple logic structure and the high order of intruders' complexity, the proposed CA-based scheme could be found to be a potential alternative to the existing one. The scheme is shown to be guarded from various previously-known and new attacks.

Such authenticated key exchange schemes can have several applications, which may be broadly classified into two: *user-to-host* and *user-to-user* authentications. This chapter highlights some of the potential applications of the proposed CA based scheme. Next section discusses some of the existing related works and some general classifications derived from the literature survey.

5.2 Existing Related Works

The landmark 1976 paper of Whitefield Diffie and Martin Hellman [Diffi76b] is the pioneer work for both the seminal idea of public key cryptography and the fundamental technique of exponential key agreement. An early work of Diffie and Hellman [Diffi76a] presented the concept of public key agreement and the use of public-key techniques for identification and digital signature. In the fall of 1974, Merkle independently conceived a particular method for key agreement [Menez97], known as 'Merkle's puzzle system', which may be presented as follows. Alice constructs 'm' puzzles. each of which is a cryptogram and which Bob can solve in 'n' steps (exhaustively trying 'n' keys until a recognizable plaintext is found). Alice sends all 'm' puzzles to Bob over an insecure channel. Bob picks one of these, solves it (cost : 'n' steps) and treats the plaintext therein as the agreed key, which he then uses to encrypt and send to Alice a known message. The encrypted message, now a puzzle which Alice must solve, takes 'n' steps (by exhaustively trying 'n' keys). For $m \approx n$, both Alice and Bob require $O(n)$ steps for key agreement while an opponent require $O(n^2)$ steps to deduce the key. Reuppel [Reupl90] explores the use of function composition to generalise Diffie-Hellman key agreement. Shmuely [Menez97] and McCurley [Mcrly88] considered composite Diffie-Hellman, i.e. Diffie-Hellman key agreement with a composite modulus. McCurley presents a variation thereof, with an RSA like modulus 'm' of specific form and particular base 'r' of high order in Z_m^*, which is probably as secure (under passive attack) as the more difficult of factoring 'm' and solving the discrete logarithm problem with modulo the factors of 'm'. Regarding Diffie-Hellman key agreement, Van Oorschot and Wiener [Oorsc91c] note that the use of "short" private exponents in conjunction with a random prime modulo

'p' (e.g. 256-bit exponent with 1024-bit 'p') makes computation of discrete logarithm easy. They also document the attacks, which is related to issues explored by Simmon [Simon95] concerning a party's ability to control the resulting Diffie-Hellman key, and more general issues of unfairness in protocols. Waldvogel and Massey [Wldvg93] carefully examine the probability distribution and entropy of Diffie-Hellman keys under various assumptions. When private exponents are chosen independently and uniformly at random from $\{0, 1 \cdots p-2\}$ (as is customary in practice); in the best case (when p is a safe prime, $p = 2q+1$, where q is prime) the most probable Diffie-Hellman key is only six-times more likely than the least probable, and the key entropy is less than 2-bits. While in the worst case governed by a particular factorization pattern of $p-1$ the distribution is still sufficiently good to preclude significant cryptanalytic advantage, for 'p' of industrial size or larger. The One-pass ElGamal key agreement protocol [Menez97] a variant of Diffie-Hellman, provides a one-pass protocol with unilateral key authentication, provided the public key of the recipient is known to the originator a priori. The MTI/AO protocol (as found in [Menez97]) is closely related to a scheme later presented by Goss. Matsumoto et al. equate the computational complexity of Passive attacks (excluding the known key attack) on selected key agreement protocols to that of one or two Diffie-Hellman problems. Active attacks (source substitution) on MTI/AO are considered by Diffie, van Oorschot and Wiener [Diffi92] and Menezes, Qu and Vanstone [Menez97]. Yacobi and Shmuely [Yacob90] note two time-variant versions of Diffie-Hellman key agreement which are insecure against known-key attacks. A similar protocol which falls prey to known-key attack was discussed by Yacobi [Yacob91], subsequently rediscovered by Alexandris et al. [Alexn93] and re-examined by Nyberg and Rueppel [Nybrg93]. Yacobi [Yacob91] proves that the MTI/AO protocol with composite modulo is probably secure under known-key attacks by a passive adversary; Desmedt and Burnmester [Desmd93], however, note the security is only heuristic under known-key attack by an active adversary. A formal logic security comparison of the protocols of Goss, Gunther, and STS is given by van Oorschot [Oorsc92]. Burnmester [Burms94] identifies known-key triangular attacks, which may be mounted on the former two and related protocols, which provide only implicit key authentication (including MTI protocol). Variations of STS and an informal model for authentication and authenticated key establishment are discussed by Diffie, Oorschot and Wiener [Diffi92]. Bellovin and Merrit [Belvn92, Belvn93] propose another hybrid protocol (Encrypted Key Exchange - EKE) involving exponential key agreement with authentication based on a small shared password, designed specifically to protect against password-guessing attacks by precluding easy verification of guessed passwords; Steiner, Tsudik and Waidner [Stein95] provide further analysis and extensions. A hybrid protocol with similar goals is given by Gong et al. [Gong93], including discussion of its relationships to EKE and expanding the earlier work of Lomas et al. [Lomas89]. Recently, Jablon [Jabln96] also proposed a simplified password exponential key exchange method, which is basically a variant of DH-EKE [Belvn92]. From the survey, it reveals that a strong authenticated key exchange scheme using only a small password, is always desirable for any insecure network. For the sake of understanding, the following subsections provide general classifications and some basic concepts, relevant to the proposed scheme, from [Menez97].

5.2.1 General Classifications

Key establishment is a process or protocol whereby a shared secret becomes available to two or more parties, for subsequent cryptographic use. Basically, one can have two broad subclasses of key establishments, such as : *key transport* and *key agreement* as defined below.

Definition 5.1 *A key transport protocol or mechanism is a key establishment technique where one party creates or otherwise obtains a secret value, and securely transfers it to the other(s).*

Definition 5.2 *A key agreement protocol or mechanism is a key-establishment technique in which shared secret is derived by two (or more) parties as a function of information contributed by or associated with, each of these, (ideally) such that no party can pre-determine the resulting value.*

Key establishment protocols involving authentication typically require a setup phase whereby authentic and possibly secret initial keying material is distributed. Most protocols have as an objective the creation of distinct keys on each protocol execution. In some cases the initial keying material predefines a fixed key which will result every time the protocol is executed by a given pair or group of users. Usually systems involving such static keys are insecure under known key attacks.

Definition 5.3 *'Key pre-distribution' schemes are key establishment protocols whereby the resulting established keys are completely determined a priori by initial keying mate-*

rial. In contrast, dynamic key establishment schemes are those whereby the key established by a fixed pair of users varies on subsequent executions.

Dynamic key establishment is also referred to as *session key establishment.* In this case, session keys are dynamic, and it is usually intended that the protocols are immune to known key attacks.

5.2.2 Basic Concepts

It is a desired property in a key establishment protocol that each participating party be able to determine the true identity of the other(s) which could possibly gain access to the resulting key, implying preclusion of any unauthorized additional parties from deducing the same key. In this case the technique is said to (informally) provide *secure key establishment.* This requires both secrecy of the key and identification of those parties with access to it. Furthermore, the identification requirement differs subtly, but in a very important manner, from that of entity authentication- here the requirement is knowledge of the identity of parties which may gain access to the key, rather than corroboration that actual communication has been established with such parties. In the following, various such related concepts are highlighted by the definitions. Entity authentication is defined as the process whereby one party is assured of the identity of a second party involved in a protocol, that the second has actually participated (i.e. active at, or immediately prior to, the time the evidence is acquired). Entity authentication presents protocols providing entity authentication alone.

Definition 5.4 *Key authentication is the property whereby one party is assured that no other party aside from a specifically identified second party may gain access to a particular secret key.*

Key authentication is independent of the actual possession of such key by the second party, it need not involve any action whatsoever by the second party. For this reason, it is sometimes, also referred to as (implicit) *key authentication.*

Definition 5.5 Key confirmation *is the property whereby one party is assured that a second (possibly unidentified) party actually has possession of a particular secret key.*

Definition 5.6 Explicit key authentication *is the property obtained when both (implicit) key authentication and key confirmation hold.*

In the case of explicit key authentication, an identified party is known to actually possess a specified key, a conclusion which can not otherwise be drawn. Encryption applications utilizing key establishment protocols which offer only implicit key authentication often begin encryption with an initial known data unit serving as an integrity check-word, thus moving the burden of key confirmation from the establishment mechanism to the application.

The focus in key authentication is the identity of the second party rather than the value of the key, whereas in key confirmation, the opposite is true. Key confirmation typically involves one party receiving a message from a second containing evidence demonstrating the possession of the key by the latter. In practice, possession of a key may be demonstrated by various means, including producing a one-way-hash encryption of a known quantity using the key.

Entity authentication is not a requirement in all protocols. Some key establishment protocols (such as unauthenticated Diffie-Hellman key agreement) provide none of entity authentication, key authentication and key confirmation. Unilateral key confirmation may always be added e.g. by including a one-way-hash of the derived key in a final message.

Definition 5.7 *An authentic key establishment protocol is a process or protocol whereby a shared secret becomes available to two or more parties and which provides assurance to the communicating parties that no other parties (except them) may gain access to a particular secret key.*

The security of a protocol is examined based on the assumption that the underlying cryptographic mechanisms used, such as encryption algorithm and digital signature schemes are secure. A *passive attack* involves an adversary who attempts to defeat a cryptographic technique by simply recording data and thereafter by analyzing (e.g. in key establishment, to determine the session key). An *active attack* involves an adversary who modifies or injects messages.

It is simply assumed that protocol messages are transmitted over unprotected network, modeled by an adversary able to completely control the data therein, with the ability to record, alter, delete, insert, redirect, reorder and reuse past or current messages, and inject new messages. An adversary in key establishment protocol may pursue many strategies, including attempting to :

- deduce session key using information gained by eavesdropping,
- participate covertly in a protocol initiated by one party with another, and influence it, e.g. by altering messages so as to be able to deduce the key;
- initiate one or more protocol executions (possibly simultaneously), and combine (interleave) messages from one with another, so as to masquerade as some party or carry out one of the above attacks.
- Without being able to deduce the session key itself, deceive a legitimate party regarding the identity of the party with which it shares a key.

In analyzing key establishment protocols, the potential impact of compromise of various types of keying material should be considered, even if such compromise is not normally expected. In particular, the effect of the following is often considered :

1. Compromise of long term secret (symmetric and asymmetric) keys, if any, and
2. Compromise of past session key.

A protocol said to have perfect forward secrecy if compromise of long term keys does not compromise past session key. The idea of perfect forward secrecy (sometimes called break forward protection) is that previous traffic is locked securely in the past. A protocol is said to be vulnerable to a *known key attack*, if compromise of past session keys allows either a passive adversary to compromise future session keys, or impersonation by an active adversary in the future.

Next section presents a CA based authenticated key exchange scheme, which fulfills the requirements, as determined in the previous discussion. The scheme is designed based on the properties of *group* & *non-group* CAs as discussed in *Chapter 2* as well as the schemes developed and reported so far, in the *Chapter 3 & 4*.

5.3 CA based Password-only Authenticated Key Exchange : CPAKE

The objective is to design an authenticated key exchange scheme based on some small shared password, which not to be vulnerable to dictionary attack. The scheme should bridge the growing gap between the smallest safe key and the size of the largest easily remembered password.

The proposed CPAKE scheme operates in two major phases; the *first* phase, establishes the secret session key K, through exchange of intermediary computed values Q; and in the *second* phase, both confirm each other's knowledge of the key i.e. K before proceeding to use it as a session key. Each of these phases operates in a sequence of stages. Key establishment *phase* is operated in two major stages and each stage follows multiple steps of processing. In the first stage, both A and B uses a small (r-bits) shared password Y_p as input to a function $f_e(Y_p)$, which expands it into an $2r$-bits Y_p'. Next Y_p', is further operated with a Pseudo-random sequence-based permutation scheme π to generate Y_p''. A strong one-way hashing scheme H_1 is applied on Y_p'' to finally generate a $2r$-bits digest Q_A, which is communicated to B. At the other end also, B will follow the similar protocol to generate Q_B and will be communicated to A. The second stage then computes the secret session key based on the values of Q_A and Q_B and a small (r-bits), shared, secret key m. It performs in two steps : in the first step, both A and B uses m with each of the Q_A and Q_B and after mixing up the bit-patterns using the function ((after necessary padding of 0's), they are subjected to the hash function H_1 to obtain the corresponding digests $Q_A{}'$ and $Q_B{}'$ ($2r$-bits each) respectively. These bit-patterns (i.e. $Q_A{}'$ and $Q_B{}'$) are then again operated with η and the output is hashed using another strong one-way function H_{11} to generate finally the $2r$-bits shared session key K.

It is presumed that John and Diana (i.e. A and B) are two well-behaved legitimate parties. In user-to-host situation, John is the user. The various notations along with their meanings, used in this scheme are described in *Figure 5.1.* The logic for the *Phase-I* operations to establish the session key is presented in *Figure 5.2.* In the second phase, both A and B confirms each other's knowledge of K before proceeding to use it as a session key. For confirmation purpose, either one can use the traditional key-confirmation method, as found in [Jabln96], or, any of the CA based encryption schemes found in [Schnr96, Palch97, Menez97] or, the ENCA based cipher system reported in the previous chapter (i.e. *Chapter 4* of the present thesis). To use the scheme proposed in *Chapter 4*, the user will select an operator T, based on the value of K. Here, T can be either Group or a non-group CA. For confirmation, each communicates to the other a randomly chosen number say c_A or c_B , which will be used as a basin element by both the end and by using T repeatedly, either the attractor (in case of non-group CA) or the least element in the corresponding cycle (in case of group CA) i.e. say, γ will be computed. Next, each of them will verify γ_A or γ_B with respect to T. The steps of processing for each phase are noted below :

Y_p	:	a small (r-bits) shared password for John and Diana.
$f_e(Y_p)$	:	a function to expand Y_p into a $2r$-bits $Y_p{}'$.
R	:	a maximal-length Group CA-based Pseudo-random Sequence Generator.
π_v^y	:	a permutation scheme π based on non-maximal-length group CA where
	:	'v' represents the particular CA and 'y' represents the cycle-length.
H_1, H_{11}	:	two group and non-group CA based strong One-Way hash functions.
m	:	a small (r-bits) shared secret for John and Diana.
$A \rightarrow B : \gamma_A$	:	John sends the value 'γ_A' to Diana.
$\eta(w, z)$	:	a function to mix $2r$-bits patterns w and z to generate a $2r$-bits patterns.
K	:	a $2r$-bits session key.

Figure 5.1: Various Notations used and their meanings

Phase I : Computation of the Secret Session Key

S1.	John computes :	$Q_A \leftarrow H_1(\pi(f_e(Y_p)))$	$A \rightarrow B : Q_A$
S2.	Diana computes :	$Q_B \leftarrow H_1(\pi(f_e(Y_p)))$	$B \rightarrow A : Q_B$
S3.	John Computes :	$K \leftarrow H_{11}(\eta(H_1(m, Q_A), H_1(m, Q_B)))$	
S4.	Diana Computes :	$K \leftarrow H_{11}(\eta(H_1(m, Q_A), H_1(m, Q_B)))$	

Phase II : Confirming the knowledge of the Key by each other

S5.	John chooses random c_A and computes : $\gamma_A \leftarrow T^{r1}.(c_A)$	$A \rightarrow B : \gamma_A$
S6.	Diana chooses random c_B and computes : $\gamma_B \leftarrow T^{r2}.(c_B)$	$B \rightarrow A : \gamma_B$
S7.	John verifies that γ_B is attractor or least-element corresponding to T	
S8.	Diana verifies that γ_A is attractor or least-element corresponding to T	

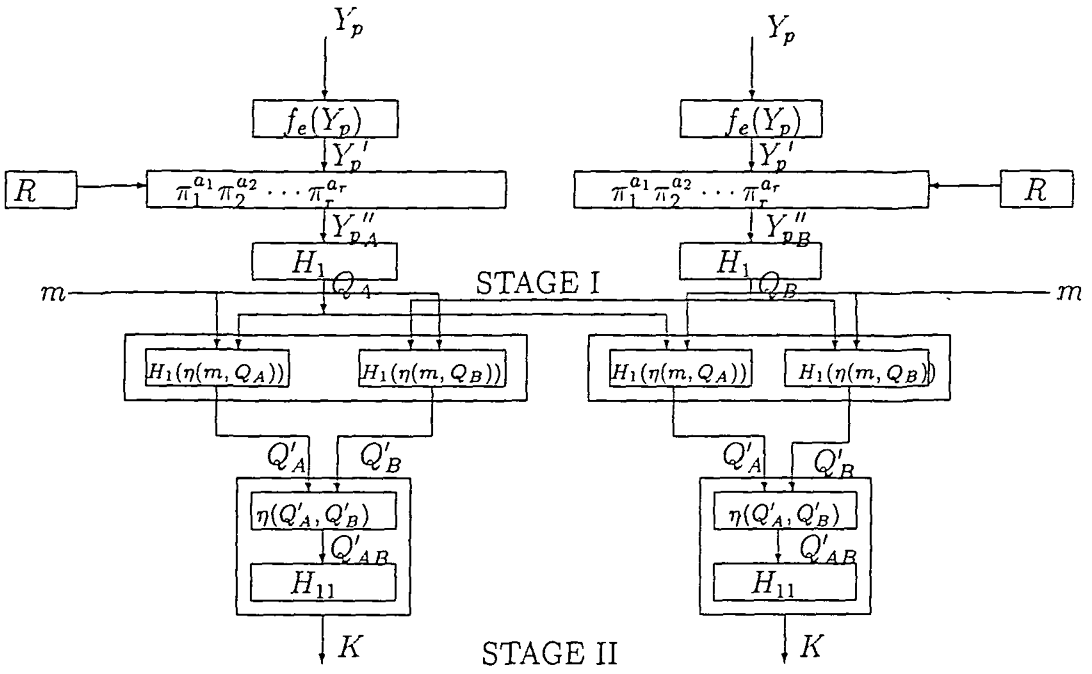


Figure 5.2: Block diagram of Session Key management

For more concrete confirmation, one can use additionally a secret-key symmetric enciphering scheme on γ_A or γ_B before communication. In the following sub-section, the various modules and functions used, are described in brief :

Expanding Function : $f_e(Y_p)$

This operation is similar with the E-Box operation as found in [DES67]. The r-bits Y_p is expanded to $2r$-bits Y_p' by an expansion permutation. This operation has mainly two purposes : it makes the result the same size (i.e. $2r$) for the next step of operation, and it provides a longer result that can be compressed during hashing operation. Because of this expansion, the dependency of the output bits on the input bits spreads faster.

CA Based Permutation Module : π

Several permutation schemes can be found in [Schnr96, Menez97]. A CA based potential alternative to the existing permutation schemes can be found in [Palch97]. The major advantages of the CA based permutation schemes are : (a) they are easily implementable and (b) there are scopes to improve the intruders' complexity to a great extent by - (i) configuring the rule-vectors dynamically (to have combinations of multiple CAs, i.e. $\pi_1, \pi_2, \pi_3 \cdots$), and (ii) using a sequence of cycle-lengths (i.e. y's) for the permutations, which are pseudo-random in nature. The cycle-length sequences are obtained from a CA based pseudo-random sequence generator, R.

The maximum-length group CAs can be used as an efficient pseudo-random sequence generators [Barde90, Nandi94a]. It is well known fact that fixed-key pseudo-random sequence generator is vulnerable to the intruders' *correlation attack*. Thus, it is desirable to have a running key generator consists of several pseudo-random sequence generator with a non-linear combining functions. The scheme proposed in the present chapter [Nandi94a], or a variant of it, based on *Programmable CA* (as found in *Section 2 from Chapter 2*) can be used here.

CA Based One-Way Hashing Function : H

The proposed CPAKE uses a CA based hashing scheme, designed based upon the properties of group, non-group and non-linear CAs, with a similar concept as discussed and reported in *Chapter 3*. The scheme hashes $2r$-bits Y_p' into a fixed-length r-bits hash value in feedback mode. Hashing is performed mainly in two steps : in the first step, r-bits block is taken, divided into 'k' sub-blocks; these sub-blocks are then subjected to three operations : Bit-wise substitution, XORing with Constants and Enciphering using a PCA based Block Cipher (a variant of which is reported in *Chapter 4*). With a shift-rotate operation among these sub-blocks, the above these operations are repeated for 'k' cycles. The final r-bits (concatenated) output is then concatenated with the previous r-bits hash value (in case of the first block, it will be zeros) and then input to a $2r$-bit non-group multiple attractor CA (MACA). CA runs for few cycles to reach the attractor [Palch97]. The Pseudo-Exhaustive r-bits of the attractor are then separated out as the present hash value, h_i. Similar process will repeat for the second block also. Thus, after repeating the processes for several cycles, the final hash value of the $2r$-bits input block, i.e. Y_p' is generated. In the following section, an analysis of the scheme in light of the various known and unknown intruders' attacks is presented.

5.4 Invulnerability of CPAKE

The proposed scheme is designed with an integrated key exchange supported by mutual authentication provision. It sufficiently proves to each of the two parties that the other knows the password. The scheme aims to generate a session key for securing a subsequent authenticated session between the parties, which does not leave any separation information regarding its individual steps. The scheme also does not generate any persistent data, which have to be distributed and securely stored. It shares only a small password Y_p and a small key m (i.e. for $r = 64$, a total of 128-bits, reasonably small in size for a secret key). Thus, the scheme is advantageous in this context. Security of the scheme against the various possible attacks are now discussed :

Dictionary Attack

All passwords are vulnerable to dictionary attack - if any opportunity is given. So, the primary job of the designer is to remove the opportunities. Dictionary attacks can be *online* as well as *offline*. In case of the proposed scheme, the online dictionary attack can be easily detected by counting the access failures and thus thwarted. However, offline dictionary attacks are of complex type and it needs to be handled with care. One can make this attack by posing as a legitimate party to gather information, or by one who monitors the messages between two parties during a legitimate valid exchange. A very little information 'leakage' during an exchange can be exploited. If the expansion, permutation or hashing functions are not carefully built, the exchanged messages Q_A or Q_B may reveal discernible structure, and can "leak" information about Y_p. In the following - it has been attempted to establish the efficiency of each function or module of the CPAKE, in this regards.

- In case of the *expansion permutation* f_e, by allowing each bit of the secret password i.e. Y_p to undergo this expansion permutation, the *avalanche* effect can be increased, because- dependency of the output bits on the input bits increased substantially.

- For the *CA-based randomizer* R, it has been established in [Nandi94a] that- the sequence generated by R fulfills all the Knuth's criteria and hence ensures the desired immunity against the conventional *correlation attack*. The huge exponential order of intruders' complexity to crack-key stream also guards the scheme from *known plaintext attack.*

- In case of the *permutation scheme* π, as the same ciphertext may be generated from different plaintext, as well as any ciphertext may give rise to different plaintext under different CA rule configurations, the scheme is guarded from *ciphertext only attack.* The exponential order of intruders' complexity guards the scheme from other known as well as chosen plaintext attacks.
- Due to the non-group, non-linear features of complemented MACA in the *one-way hashing scheme* H, as well as the due to the PCA-based block ciphering strategy used for intermediate substitution, the scheme has been found to be immune from *hash-value only attack.* In case of the known as well as chosen (*message, hash-value*) attack, as the scheme offers an exponential order of complexity, approximately of the order of $O(2^{8\times r})$ (as reported in *Chapter 3*), the scheme is guarded from other possible attacks.

Stolen Session Key Attack

In this type of attack, a stolen session key K is used to mount a dictionary attack on the password Y_p [Jabln96, Stein95]. CPAKE appears to be guarded in this attack, because with only K, probably it will not be easy to compute Y_p - it requires the appropriate assessment of R, π, f_e and also the hash functions H_1 and H_{11} to find Y_p, whereas each of them already has been established to be immune from ciphertext only attack.

Verification Stage Attack

The verification stage of CPAKE is where both parties prove to each other- knowledge of the shared key K. As K is cryptographically large, the second stage is presumed to be immune to brute-force attack.

5.5 Applications of CPAKE

Password-only schemes are broadly useful for any applications where the prolonged key storage is risky or impractical, and where the communication channel may be insecure.

Some of its common applications are : user-to-user applications, diskless workstations, bootstrapping new system installation, cellular phones or other key-pad systems, multi-factor password + key systems etc. From the economic point of view as well as to encounter the stolen key problems, these authentication schemes are always preferable than the *smart-cards*. Applications of password-only methods basically can be classified into two : *user-to-host authentication* and *user-to-user authentication*; some of the potential applications of the proposed CPAKE are described here.

5.5.1 User-to-Host Authentication

In case of the systems, where only a numeric key-pad is available, e.g. *cellular telephone authentication*, such a CA based password-only scheme is especially convenient, as it is based on a small numeric password. *TV remote-controlled set-top boxes* might be another area of new applications for the proposed scheme. It is advantageous in terms of these applications, because- it avoids the necessity for long-term storage of persistent keys for such environments.

Diskless workstations are another class of device where it is inconvenient to have locally stored keys. The proposed password-only method can be ideal for establishing an initial connection to a trusted host, and to obtain the user's safely stored credentials. The concept of *bootstrapping* a new secure system is broad, and is best illustrated with a common case. Unless some additional site-specific keys are manually installed on the system, a strong password-only method is needed to allow the new station to make a secure channel to the rest of the network. Once an authenticated channel is established, the station can automatically obtain any further credentials or keys. Similar bootstrapping situation arise in almost all secure systems.

5.5.2 User-to-User Authentication

So far, it has been established the usage of CPAKE on *user-to-host* authentication; this method is equally useful in direct *user-to-user* authentication also. [Ellis96] describes the use of an interactive questionnaire session to authenticate the identity of a user across a network. The main idea behind this authentication is that they are sharing some common facts, which has to be proved to each other, without revealing those facts.

Such general authentication paradox can easily be solved by the proposed CA based authentication method. For example, in a bank application, the banker may want to know that his client knows his *secret social security number*, and at the same time, the client may also like to know that the bank knows his *secret account number*, but neither wants to reveal the information directly to the other. In solving such a problem, a CA based password-only method can be successfully applied.

5.6 Conclusion

A new password-only authentication protocol based upon the properties of group and non-group CAs is introduced in this chapter, which appears to be at least as strong as the existing SPEKE [Jabln96] and DH-EKE [Diffi92] methods. Using particularly a small password, the proposed method provides authentication over an insecure channel, and are immune to offline dictionary attack. Due to its exponential order of intruders' complexity, the scheme is guarded against the other possible types of attacks. The proposed scheme utilizes non-group CA (MACA) as an efficient hashing function generator and the PCA based block ciphering mechanism, in generating the intermediate values. The major advantage of the scheme is the use of simple, regular, modular and cascadable structure of CA as the basic building block that is ideally suitable for VLSI implementation. The possible usage of the proposed scheme has been justified in many application domains, such as: user-to-host authentication, user-to-user authentication etc. In all such applications, one of the major requirements is the receiving of error-free information by both the participating parties. As such, it is essential that the errors if any, caused during transmission of information be detected and (if possible) corrected at the receiving end. The next chapter presents two schemes which take into account such communication errors.

Chapter 6

Theory and Design of Unidirectional Error Correcting Codes

6.1 Introduction

In the earlier chapters theory and applications of binary matrices for designing authorization and data security schemes have been addressed. With the advancement in the convergence of computer and communication technologies, there has been an increasing demand for efficient and reliable digital data transmission and storage systems. This demand has been accelerated by the emergence of large-scale, high-speed data networks for exchange, processing and storage of digital information in the military, governmental and private spheres. In this background, a major concern of a researcher is to control the errors so that reliable reproduction of data can be obtained.

It is observed that many faults in VLSI memories, PLA, shift registers are due to three classes of errors : (a) Symmetric, where both $1 \to 0$ and $0 \to 1$ errors are equally likely in a codeword, (b) Asymmetric, where only one of the errors, $1 \to 0$ or $0 \to 1$, can occur in a codeword, and (c) Unidirectional, where both $1 \to 0$ and $0 \to 1$ errors can occur but not in the same codeword. The various possible causes of such errors are : transient, intermittent and permanent, among which the transient errors are responsible for mostly limited number of symmetric or multiple unidirectional errors. The intermittent faults cause a limited number of errors and, but the permanent faults may cause both symmetric as well as the unidirectional errors. Among these errors, many faults in VLSI circuits cause mostly unidirectional errors [Rao89]. The number of symmetric errors is usually limited while the number of unidirectional errors can be very large. Thus, construction of an efficient class of symmetric error(s) correcting codes which can also simultaneously detect all unidirectional errors, is essential. In the next subsection, the background of such error correcting codes with review of some of the existing related works have been discussed.

6.1.1 A Review of Error Correcting Codes

In a generalized error correcting code, the source encoder transforms the information sequence I_s into another sequence C, called the *codeword*. In most instances C is a binary sequence. After transfer through the coding channel, let the codeword be transformed from C to C' due to some error. The decoder receives the codeword C' that is the same as C in error free condition. Using some error correction logic, the errors, if there are any, are detected or corrected by the decoder on processing the codeword C'. The correct information sequence I_s can thus be forwarded to the destination. There are two different types of codes in common use : (a) *Block Codes* and (b) *Convolutional Codes*. This chapter is restricted to the *linear* block codes only. The basic principles of linear block codes are briefly given as follows. A detailed discussion on block codes and convolution codes may be found in [Rao89].

Linear Block Codes : A *block code* of length n and 2^k codewords is called a *linear* (n, k) code if and only if its 2^k codewords form a k-dimensional subspace of the vector space of all the n-tuples over the field GF(2). A *binary block code* is linear iff the module-2 sum of two codewords is also a codeword; that is, if u and v are two codewords than $u + v$ is also a codeword, where $+$ refers to bit-wise module-2 sum. Let, the vector space V_n of all n-tuples over GF(q) be

$$\mathrm{V_n} = (a_0, a_1, a_2, \cdots, a_{n-1}) \mid a_i \in GF(q);$$

A subset W of V_n is a linear code, if and only if it is a subspace. Since W is a subspace, all the codewords of W can be conveniently represented as the row space of a $(k \times n)$ matrix G, called the *generator matrix*, which is given by,

$$G = \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & \cdots & g_{1,n-1} \\ \cdots & \cdots & \cdots & \cdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix} = \begin{bmatrix} g_0 \\ g_1 \\ \cdots \\ g_{k-1} \end{bmatrix}$$

If the row vectors of G are linearly independent, then W has dimension k and has exactly q^k codewords in it. Such a code W is called an (n, k) code. Alternatively, W can be defined as the 'null space' of the parity matrix H, which may be written as,

$$H = \begin{bmatrix} h_0 \\ h_1 \\ \cdots \\ h_{r-1} \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & \cdots & h_{0,n-1} \\ h_{10} & h_{11} & \cdots & h_{1,n-1} \\ \cdots & \cdots & \cdots & \cdots \\ h_{r-1,0} & h_{r-1,1} & \cdots & h_{r-1,n-1} \end{bmatrix}$$

Again the r vectors of H are linearly independent, and thus H has rank r and its null space has dimension $(n - r)$. Therefore, the H matrix of an (n, k) code satisfies the relation $r = n - k$. The G matrix can be represented in its systematic form [Rao89] as $G = [I_k, P]$, where, I_k is a $k \times k$ identity matrix, and P is the 'parity generator matrix', which operates on the information block to generate the check-bits. The H matrix is represented as $H = [-P^T I_{n-k}]$, so that $g_i h_j = 0$, for all i, j; that is, G is orthogonal to H. P^T refers to the transpose of P and the "-" sign implies negation of P^T; i.e. an element p_{ij} of P becomes $-p_{ij}$ in $-P$, where $-p_{ij}$ is the additive inverse of p_{ij} (in mod-2 operation, $-p_{ij} = p_{ij}$).

Let us now consider a data (information) sequence I_s of k symbols $(i_0, i_1, \cdots i_{k-1})$. If W is an (n, k) linear code over GF(q), the codeword will be an n-tuple $C = (c_0, c_1, \cdots, c_{n-1})$. If the symbols $(i_0, i_1, \cdots, i_{k-1})$ of information I_s appear in its codeword C unchanged, then the code is said to be a *Systematic Code* [Lin83]. So, if the parity matrix H is of systematic form $H = [P^T I_{n-k}]$, then the codeword $C = [H.I_s]$ generates a systematic code. It is often very important in computer systems that the codewords are in systematic form for speed, cost, and convenience in decoding and processing.

The decoding of linear codes is usually done in parallel, based on the H matrix of the code. For a received word $C' = (c_0', c_1', \cdots, c_{n-1}')$, its syndrome $s(C') = (s_0, s_1, \cdots, s_{n-k-1})$ is defined as $s(C') = C' \cdot H^T$. If $s = 0$, then there are no errors in the received codeword; whereas for $s \neq 0$, some errors are detected. Based on the properties of the code, the detection and correction steps are implemented in the decoder. Let us assume that the correct codeword transmitted is $C = (c_0, c_1, \cdots, c_{n-1})$, and the corresponding received word is $C' = (c_0', c_1', \cdots, c_{n-1}')$. Then the error word is $E = (e_0, e_1, \cdots, e_{n-1})$ such that -

$$C' = C + E \text{ and, } E = C' - C = (e_0, e_1, \cdots, e_{n-1});$$
$$\text{i.e., } e_i = c_i' - c_i \text{ for } i = 0, 1, \cdots n - 1.;$$

Then, $s(C') = C' \cdot H^T = C \cdot H^T + E \cdot H^T = 0 + E \cdot H^T = E \cdot H^T = s(E)$. A few encoding/decoding schemes of linear block codes are reviewed next.

Hamming Codes : Hamming codes are the first major class of linear block codes designed for error correcting purpose [Rao89]. For a Hamming code of length $(2^m - 1)$ (where m is a single integer)- one has to construct a matrix where columns consist of all non-zero binary m-tuples. For example, a (n, k) Hamming code is defined by the parity-check-matrix H, where the ordering of the columns is arbitrary and the matrix is designed specially to describe a systematic code. In the corresponding generator matrix G, the k-information bits always occupy the last k co-ordinates of each codeword. In general, the minimum distance of (n, k) Hamming code, i.e. the smallest number of distinct non-zero binary m-tuples that can sum to zero is *three*, and it can be used either as : (i) Single Error Correcting (SEC), or (ii) Double Error detecting (DED) code.

However, one can modify a (n, k) Hamming code by adding one parity bit $(n + 1, k)$ SEC-DED code, where the syndrome-bit is the XOR sum of all $(n + 1)$ inputs; which results in higher hardware cost and circuit delay. To circumvent this problem, the Hamming SEC-DED code has been modified and optimized. The resultıng code is called modified Hamming SEC-DED code or *Hsıao Code* [Palch97], which is described next.

Hsiao Code : Here, the minimum distance of a SEC-DED code is at least 4. Since, an n-tuple of weight 3 or less is not a codeword, any set of 3 columns of the H matrix should be linearly independent. The sum of two odd-weight r-tuples is an even-weight r-tuple (i.e. odd + odd = even; even + odd = odd; even + even = even). Based on this property, a SEC-DED code with r check-bits can be constructed by performing some row operations on Hamming SEC-DED H matrix. The modified H matrix consists of distinct non-zero r-tuples of column vectors having odd-weight. This modified Hamming code is called *odd-weıght-column SEC-DED code* or *Hsıao code* because every H matrix column vector is odd-weight.

t-Error Correcting Codes : Let C_1 and C_2 be two codewords in C, such that the minimum distance D (say) between them can be defined as : $d(C_1, C_2) = D$. Let E_1 and E_2 be two error patterns that satisfy the conditions : (a) $E_1 + E_2 = C_1 + C_2$; and (b) E_1 and E_2 do not have non-zero components in common place. Thus we have :

$$W(E_1) + W(E_2) = W(C_1 + C_2) = d(C_1, C_2) = D \tag{6.1}$$

Now, suppose that C_1 is transmitted and is corrupted by the error pattern E_1. Then, the received codeword is : $R = C_1 + E_1$; The Hamming distance between C_1 and R is thus,

$$d(C_1, R) \quad = W(C_1 + R) \quad = W(E_1) \tag{6.2}$$

Similarly, the Hamming distance between C_1 and R will be :

$$d(C_2, R) = \quad W(C_2 + R) = \quad W(C_2 + C_1 + R) = W(E_1) \tag{6.3}$$

Now, suppose that the error patterns E_1 contains more than t-errors (i.e. $W(E_1) > t$). Since, $2(t + 1) \leq D \leq (2t + 2)$, based on *equation 6.1*, we have : $W(E_2) \leq (t + 1)$. Combining *equations 6.2 & 6.3* and using the fact that $W(E_1) > t$ and $W(E_2) \leq (t+1)$, we obtain the following inequality :

$$d(C_1, R) \quad \geq \quad d(C_2, R) \tag{6.4}$$

This inequality says that there exists an error pattern of $l(l > t)$ errors which results in a received codeword that is closer to an incorrect codeword than to the transmitted codeword. Summarizing the above results, a block code with minimum distance D guarantees correcting all the error patterns of $t = \lfloor (D - 1)/2 \rfloor$ or fewer errors, where $\lfloor (D - 1)/2 \rfloor$ denotes the largest integer no greater than $(D - 1)/2$. The parameter $t = \lfloor (D - 1)/2 \rfloor$ is called the 'random error correcting capability' of the code. The code is referred to as a *t-error correcting* code.

Fujiwara [Fujiw78] proposed a generalized odd-weight-column code, where the H matrix of the subspace W over GF(2^q) satisfies the following condition for every column in the H matrix : $\sum_{i=0}{}^{r-1} = I_q$, for columns, $j = 0, 1, \cdots, n - 1$; where, $h_{i,j}$ is the i^{th} element in the j^{th} column vector $\in$ GF(2^q); I_q is the identity element of GF(2^q), and $\sum$ denotes summation in GF(2^q). The major advantage of this code is that it has a minimum equal-weight code, which makes the hardware implementation of the encoding/decoding circuit optimal.

However, the current survey reveals that the failures in the cells of semiconductor based large scale integrated (LSI) non-volatile memories are most likely caused by leakage of the charge, since a charge cannot be created except by a rewrite process. These memory cells are apt to exhibit *unidirectional errors*. Although the rest of the memory system the address decoding, the cell selection, the gating logic and so on is subject to *symmetric failures*, for the overall memory system, the probability of *unidirectional errors* is significantly large. Some of the theoretical basics of *unidirectional errors* are described next.

Unidirectional Error Codes : Let, $N(X,Y)$ represents the number of $1 \rightarrow 0$ crossovers from X to Y. That is, $N(X,Y)$ is the number of positions i for which $x_i = 1$ and $y_i = 0$. For example, when $X = (10101)$ and $Y = (10010)$, $N(X,Y) = 2$ and $N(Y,X) = 1$. The asymmetric distance is,

$$d_a(Y,X) = max[N(X,Y), N(Y,X)] = 2;$$

Definition 6.1 *For vectors $X = (x1, x2, \cdots x_n)$ and $Y = (y1, y2, \cdots y_n)$, X is said to cover Y if for all i, $y_i = 1$ implies $x_i = 1$.*

We can write this as $Y \leq X$. If $X \leq Y$ and $Y \leq X$, then X and Y are said to be *unordered*. If $X \leq Y$ or $Y \leq X$, then they are *ordered* pair. For $X_1 = (1011)$ and $Y_1 = (1001)$, then $Y_1 \leq X_1$. Also $X_2 = (1010)$ and $Y_2 = (0110)$ are unordered. Also, if X and Y are an ordered pair, then their $d_a(X,Y) < d(X,Y)$. The unidirectional errors in a code are detected based on the following theorem.

Theorem 6.1 *A code C is capable of detecting all unidirectional errors (i.e. it is a AUED code) if and only if every pair $X, Y \in C$ are unordered.*

Two important classes of AUED codes are known as *constant-weight codes* and *Berger codes*. The *constant-weight* or *m-out-of-n* codes are of non-systematic as well as non-linear type. But, for any given n, the number of codewords (of weight m) is given by $\begin{pmatrix} n \\ m \end{pmatrix} = n!/((n-m)!m!)$. For the code $m = \lfloor n/2 \rfloor$ or $\lceil n/2 \rceil$ this number is maximized. This class of *constant-weight* code has the higher possible information rate, because of which- it is referred to as optimal class of AUED codes [Rao89]. However, the *Berger codes* are systematic type; for the codeword $[a_0, a_1, \cdots a_{k-1}, a_k, \cdots, a_{n-1}$, the information part is :$(a_0, a_1, \cdots, a_{k-1} = I_s$ and $(a_k, \cdots, a_{n-1}) = \beta(I_s)$ is the *Berger check bits*. The *Berger check* $\beta(I_s)$, is the binary number representing the number of 0's in I_s. The number of check-bits r required is given by $r = n - k = [\log_2 k + 1]$, e.g., for a 6-bit ($k = 6$) information, $r = [\log_2 6 + 1] = 3$. *Berger check(s)* are important and form the basis for the construction of systematic classes of $t - EC/AUED$ codes, which is described next.

t-EC/AUED Codes : This class of codes have several applications such as semiconductor memories, PLA, shift registers etc [Pradh80]. Bose and Rao [Bose82] had shown that a code is a t-EC/AUED code if and only if, for all distinct $X, Y \in C$, $N(X,Y) \geq t+1$ and $N(Y,X) \geq t+1$, where $N(X,Y)$ is the number of $1 \rightarrow 0$ crossovers from X to Y. In practice, however the codes are desired to be systematic. Most of the designers construct it by adding a tail to t-EC code such that the resulting code becomes t-EC/AUED code. In [Blaum89], Blaum and van Tilborg proposed their t-EC/AUED code by constructing $(n', k, 2t+1)t$-EC codes and appending a tail of length r such that the code can satisfy the conditions for t-EC/AUED code, where k is the number of information bits. The overall code length of t-EC/AUED codes in [Blaum89] is $n = n' + r$, and the number of redundant bits is $n - k$. Generally, the *tail* is a function of the number of weight of the codewords in t-EC code. Thus, by reducing the number of weight of the codewords, one can get better t-EC/AUED codes due to the small length r. Construction of [Bruck92] can be viewed as a modification of the construction in [Blaum89]. Several other significant systematic t-EC/AUED codes and more general t-EC/d-ED/AUED codes have been reported in [Bose85, Lin88, Nikol86, Rao89, Boinc90, Kundu90, Katti96a, Katti96b, Yang98]. In the following, the necessary and sufficient conditions for a code to be t-EC/d-ED/AUED are stated :

Theorem 6.2 *[Rao89] A code C^* can correct t or fewer unidirectional errors, if and only if for all X^*, Y^* $\in C^*$*

$\Delta(X^*,Y^*) \geq 2t+1$ *for X^*, Y^* an ordered pair*

$\Delta(X^*,Y^*) \geq t+1$ *for X^*, Y^* unordered*

Theorem 6.3 *[Rao89] A code C^* is t-EC-AUED if and only if*

$N(X^*,Y^*) \geq t+1$ *for all X^*, $Y^* \in C^*$.*

Theorem 6.4 *[Nikol91] A code C^* is t-EC-d-ED-AUED if and only if for all distinct X^*, $Y^* \in C^*$*

$D(X^*,Y^*) \geq t+d+1$

$N(X^*,Y^*) \geq t+1$

$N(Y^*,X^*) \geq t+1$.

The next section presents a simple method to construct a class of t-EC/d-ED/AUED codes, which employs simpler and faster encoding/decoding algorithm with lesser complexity.

The *notations* used in the subsequent Sections of this chapter and their meanings are given below :

D(C) :	Minimum distance of the code C.
N(X,Y) :	Number of $1 \rightarrow 0$ crossovers from X to Y.
D(X,Y) :	Hamming distance between X and Y.
W(X) :	Weight or number of 1's in X.
Δ(X,Y)	= max { N(X,Y), N(Y,X)}: asymmetric distance between X and Y.
$\lfloor Z \rfloor$:	The largest integer less than or equal to Z.
$\lceil Z \rceil$:	The smallest integer greater than or equal to Z.
k :	number of information bits.
n :	k number of information bits and parity check bits for t-EC/d-ED ($t < d$) Code.
n^* :	n bit t-EC code and additional check bits for t-EC/d-ED/AUED code.

In the next sections of this chapter, the theory and design of both the schemes have been described.

6.2 Theory of t-EC/d-ED & t-EC/d-ED/AUED Codes

Let C denote a binary linear systematic t- error correcting (n, k) code with D(C) $\geq t + d + 1$ and C^* be a systematic (n^*, k) t-EC/d-ED/AUED code. For a codeword $X \in C$, the codeword in the proposed t-EC/d-ED/AUED systematic code C^* have the following form -

$$X\ X_1X_2X_3 \cdots X_r.$$

In other words, each codeword of C^* is formed by concatenating a codeword X of C with

$X_1, X_2, X_3, \cdots X_{r^*}$, where X_i's are check bits derived by using Berger's technique. The appended check bits are used to increase the crossover between two codewords. Some of the properties of t-EC/d-ED code have been established, which would provide the foundation for the proposed code.

Let C_i be the set of all codewords of C with exactly i number of 1's. It should be noted that some of the sets C_i (e.g., $C_1, C_2, \cdots C_{t+d}$) are empty because they can not satisfy the minimum distance constraint. *Example 6.1* noted below illustrates the sets C_i for (16,3) 2-EC/5-ED code.

Example 6.1 Consider the (16,3) 2-EC/5-ED code[1] with D(C)=8. It will be partitioned into subsets C_0, C_8, C_{10} and C_{14} as shown in *Table 6.1.* □

Table 6.1: (16,3) 2-EC/5-ED CODE

infor. bits	check bits	in C_i
000	0000000000000	C_0
001	1111110000001	C_8
010	0001111110001	C_8
011	1110001110000	C_8
100	0000001111111	C_8
101	1111111111110	C_{14}
110	0001110001110	C_8
111	1110000001111	C_{10}

Lemma 6.1 *[Nikol91] Let C be a parity check code with minimum Hamming distance D(C)=t+d+1, where $d \geq t$. If $X,Y \in C$, $X \neq Y$ and W(X)-W(Y)=q ≥ 0 then N(X,Y)$\geq$ $t+1$ and N(Y,X)$\geq$ max $\{ 0, t+1-\lceil (q-(d-t))/2 \rceil \}$.*

Theorem 6.5 $\forall$ *$Y \in C_i$ and $X \in C_{i+q}$, the following equality holds:*

$$N(X,Y)\text{-}N(Y,X)=q.$$

Proof : *There are at least q bits where X has a 1 but Y has 0. Let Y' and X' be the bit*

[1]using parity check generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

strings obtained by removing those particular q bits from the respective strings. Hence, W(X') and W(Y') have the same value. So change in bit positions of X' and Y' occur pairwise, i.e.

$$N(X',Y') = N(Y',X')$$

Hence, $N(X,Y) = N(X',Y') + q$ and $N(Y,X) = N(Y',X')$

$\implies$ *N(X,Y) - N(Y,X) = q* □

Corollary 6.1 *$D(X,Y) + q$ is always even.*

Proof :

$$\begin{aligned} D(X,Y) + q &= N(X,Y) + N(Y,X) + q \\ &= N(X',Y') + N(Y',X') + q + q \\ &= 2N(X',Y') + 2q \\ &= 2((N(X',Y') + q) \end{aligned}$$

□

From the *theorem 6.5 & 6.1*, the following fundamental relations for t-EC/d-ED code are obtained :

a) For q=0

D(X,Y)$\geq t + d + 1$ and $d > t$

Hence from *theorem 6.5*, we get

$$N(X,Y) = N(Y,X) \geq t + 1 \tag{6.5}$$

b) For $1 \leq q \leq t + d + 1$,

1) if $t + d$ is even

I) if q is odd then as per *corollary 6.1*

D(X,Y) is also odd i.e. D(X,Y) $\geq t + d + 1$.

Hence from *theorem 6.5*, we get

$$N(X,Y) \geq t + 1 + (q - 1 + d - t)/2 \ \&\ N(Y,X) \geq t + 1 - (q + 1 - (d - t))/2$$

II) if q is even then as per *corollary 6.1*

D(X,Y) is even i.e. D(X,Y) $\geq t + d + 2$

Hence from *theorem 6.5*, we get

$$N(X,Y) \geq t + 1 + (q + (d - t))/2 \ \&\ N(Y,X) \geq t + 1 - (q - (d - t))/2$$

2) if $t+d$ is odd

I) if q is odd then as per *corollary 6.1*
D(X,Y) is also odd i.e. D(X,Y) $\geq t+d+2$.
Hence from *theorem 6.5*, we get

$$N(X,Y) \geq t+1+(q+(d-t))/2 \ \& \ N(Y,X) \geq t+1-(q-(d-t))/2$$

II) if q is even then as per *corollary 6.1*
D(X,Y) is even i.e. D(X,Y) $\geq t+d+1$
Hence from *theorem 6.5*, we get

$$N(X,Y) \geq t+1+(q-1+d-t)/2 \ \& \ N(Y,X) \geq t+1-(q+1-(d-t))/2$$

Hence,

$$N(X,Y) \geq t+1+\lfloor (q+(d-t))/2 \rfloor \ \& \ N(Y,X) \geq t+1-\lceil (q-(d-t))/2 \rceil \quad (6.6)$$

c) For $q \geq t+d+1$
D(X,Y) $\geq q$.
Hence from *theorem 6.5*, we get

$$N(X,Y) \geq t+d+1 \quad \text{and} \quad N(Y,X) \geq 0 \quad (6.7)$$

In the above discussions, some of the fundamental properties of t-EC/d-ED codes have been derived, which will be utilized in the next section to construct our t-EC/d-ED/AUED code.

6.3 Design of Systematic t-EC/d-ED/AUED Codes

Our objective is to construct a t-EC/d-ED/AUED code (i.e. C^*) from a given t-EC/d-ED code(i.e. C) by appending r^* number of extra check bits. These r^* check bits (i.e. $X_1X_2X_3\cdots X_{r^*}$) can be divided into three subsets. First subset covers the last bit, second subset is the group of p-bits (where $t \leq p \leq 2t$) referred to as 'last group' and third one is the subset consisting of groups each with (t+1)-bits (*Fig 6.1*). The 'last group' with p bits can have values $\{11\cdots111, 11\cdots110, 11\cdots100, \cdots, 10\cdots000, 00\cdots000\}$. The 'last group' of check bits can be derived by using a simple combinational logic circuit. The bit patterns in each of the remaining groups have only two values either

$11\cdots111$ or $00\cdots000$- that is either all 1's or all 0's.

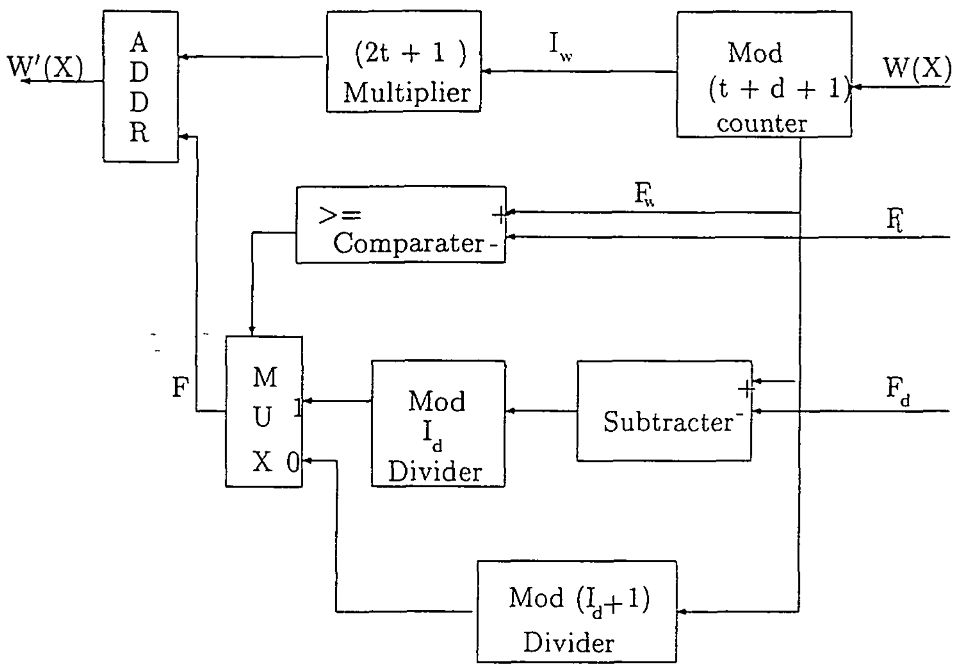


Figure 6.1: Circuit block diagram for mapping function

The Function '*map*' noted below maps weight of a codeword, W(X) in t-EC/d-ED to $W'(X)$, called effective weight which is equal to the weight of a codeword in t-EC. This is a many-to-one mapping function. We design this function by utilizing the results of the equation (6.5), (6.6) and (6.7). The table noted below specifies the value of N(Y,X) for different ranges of q both for t-EC and t-EC/d-ED codes.

Table 6.2: Values of N(Y,X) for Different qs

t-EC	t-EC/d-ED	N(Y,X)
q=0	q=0	N(Y,X)$\geq t+1$
$1 \leq q \leq 2t$	$1 \leq q \leq t+d$	$t+1 \geq N(Y,X) \geq 1$
$q \geq 2t+1$	$q \geq t+d+1$	$N(Y,X) \geq 0$

Keeping the above discussions in view we formulate mapping function as noted below.

Function *map*(u)

```
/* It takes input u, weight of a codeword C in $t$-EC/$d$-Ed, $t$ (the number of errors
to be corrected) and $d$ (the number of errors to be detected) */.
/* It returns effective weight, $u'$ which corresponds to the weight of a codeword in $t$-EC. */
   begin
      D $\leftarrow t + d + 1$; /* Hamming distance for $t$-EC/$d$-ED codes. */
      dist $\leftarrow 2t + 1$; /* Hamming distance for $t$-EC codes. */
      $F_d \leftarrow$ D mod dist;
      $I_d \leftarrow \lfloor$ D/dist $\rfloor$;
      $F_t \leftarrow (I_d + 1)\times F_d - 1$;
      $I_w \leftarrow \lfloor$ u/D $\rfloor$;
      $F_w \leftarrow$ u mod D;
      if ($F_w \geq F_t$) then
         $F \leftarrow \lfloor (F_w - F_d)/I_d \rfloor$ ;
         else $F \leftarrow \lfloor F_w /(I_d+1) \rfloor$;
      $u' \leftarrow I_w \times$ dist + $F$; /* It is effective weight. */
      return($u'$);
   end;
```

Table 6.2 illustrates the Function '*map*' for $t = 2$. For example, in case of W(X)=11, 12 and 13 (sixth row of *Table 6.2* with t=2 and d=8) we get $W'(X)$s=5.

Next, the algorithm find_no_of_bits will be discussed, which will operate on number of information and parity check bits (i.e.n). First it maps n to its effective value by Function '*map*' and gives number of appending bits (i.e. r^*). It also generates information regarding the number of groups and the number of bits in the last group, that is the value of p.

The Algorithm find_no_of_bits : This algorithm accepts 'n' (no. of bits in a Codeword C) as inputs and generates no_of_bits (appendable to the codeword C to form C^*), '*last*' (no. of bits in the last group) and '*max*'(no. of groups in the third subset) as the outputs.

Step 1: Compute maximum number of different values that 'last-group' can have :

$temp\_t \leftarrow (2t+1)$

Step 2: Find effective value of 'n'
$n \leftarrow map(n)$

Step 3: Compute $(n + 1/2)$ and store its ceiling value in '$temp$'

Step 4: Initialize max to 0

Step 5: Repeat while $(temp > dist)$

Step 5.1 increment 'max'

Step 5.2 assign ceiling value of $(temp/2)$ to $temp$

Step 6: Decrement '$temp$' by 1 and store in '$last$'

Step 7: Compute the number of bits to be appended -
no_of_bits $\leftarrow max * (t+1) + last + 1$

Step 8: Return

The 'find_code' algorithm generates the extra check bits i.e. $X_1X_2 \cdots X_{r^*}$. It accepts the number of 1's of a codeword in C as input and other information like '$last$' and 'max' from the previous algorithm.

The 'find_code' Algorithm : This algorithm takes input 'no_of_one' (i.e. the number of 1's, W(X) in the codeword $X \in C$), '$last$' and 'max' as input from the previous algorithm. It uses an array 'group' to store the value of the appended r^*. The steps are

Step 1: Compute effective number of 1's by using function 'map' and store in 'no_of_one'

Step 2: For i = 0 to $max + 1$ initialize 'int' array with 0. The 'int' array is used to store intermediate values of group of bits.

Step 3: Store- 'modulo-2' output of 'no_of_one' to int[0] and floor-value of $(no\_of\_one)/2$ to 'no_of_one'

```
Step 4: Assign 'count' to 1

Step 5: Compute -
        $shift\_val \leftarrow no\_of\_one \bmod (last+1)$
        $no\_of\_one \leftarrow \lfloor no\_of\_one/(last+1) \rfloor$

Step 6: Repeat while $no\_of\_one > 0$

Step 6.1: increment 'count' by 1

Step 6.2: compute int[count]$\leftarrow$ $no\_of\_one \bmod 2$
        $no\_of\_one \leftarrow \lfloor no\_of\_one /2 \rfloor$

Step 7: Increment $max$ by 1 and store in 'count'

Step 8: Repeat while (count > 1)

Step 8.1: if int[count] = 1 assign '00...000' to 'group[count]'
        else assign '11...111' to 'group[count]'

Step 8.2: Decrement 'count' by 1

Step 9: Perform switch(shift_val) -
        case $shift\_val = 0$ : group[1]$\leftarrow$ '111...11'
        case $shift\_val = 1$ : group[1]$\leftarrow$ '111...10'
        case $shift\_val = 2$ : group[1]$\leftarrow$ '111...00'
        :::
        case $shift\_val = last$-1 : group[1]$\leftarrow$ '100...00'
        case $shift\_val = last$ : group[1]$\leftarrow$ '000...00'
        end_of_switch

Step 10: if (int[0] = 1)
        group[0]$\leftarrow$ 0;
        else
        group[0]$\leftarrow$ 1;
```

Table 6.2 illustrates the Procedure 'find_code' for $r^*= 6$ and t=2. For example, in case of $W(X)$=11, 12 and 13 (sixth row of *Table 6.2* with d=8) we get 111 00 0 as the value of check bits. *Table 6.3* shows (22,3) 2-EC/5-ED/AUED codes generated by the proposed

Table 6.3: Output of the Function '*map*' and Procedure 'find_code' for r^*=6 and t=2

		W(X)				
$X_1X_2\cdots X_{r^*}$	$W'(X)$	$d=4$	$d=5$	$d=6$	$d=7$	$d=8$
111 11 1	0	0,1	0,1	0,1	0,1	0,1,2
111 11 0	1	2,3	2,3	2.3	2,3	3,4
111 10 1	2	4	4,5	4,5	4,5	5,6
111 10 0	3	5	6	6.7	6,7	7.8
111 00 1	4	6	7	8	8,9	9,10
111 00 0	5	7,8	8,9	9,10	10,11	11,12,13
000 11 1	6	9,10	10,11	11,12	12,13	14,15
000 11 0	7	11	12,13	13.14	14,15	16,17
000 10 1	8	12	14	15.16	16,17	18,19
000 10 0	9	13	15	17	18,19	20,21
000 00 1	10	14,15	16,17	18.19	20,21	22,23,24
000 00 0	11	16,17	18,19	20.21	22.23	25,26

Table 6.4: 2-EC/5-ED/AUED codes

X		$X_1X_2\cdots X_{r^*}$
000	0000000000000	111 11 1
001	1111110000001	111 00 0
010	0001111110001	111 00 0
011	1110001110000	111 00 0
100	0000001111111	111 00 0
101	1111111111110	000 10 1
110	0001110001110	111 00 0
111	1110000001111	000 11 1

scheme. We now prove that the above algorithm results in a t-EC/d-ED/AUED codes C^*. Let (X, $X_1, X_2, \cdots X_{r^*}$) and (Y, $Y_1, Y_2, \cdots Y_{r^*}$) be two codewords in C^* constructed using the above procedure. The r^* check bits, as discussed earlier, are divided into thre subsets. Let for a codeword X^* in C^* the subsets be denoted as S_{1x}, S_{2x} and S_{3x}. Th subsets S_{1y}, S_{2y} and S_{3y} are defined similarly for codeword $Y^* \in C^*$. Crossover between these three subsets of Y^* with that of X^* are denoted as :
n_1=N(S_{1y}, S_{1x}) = change in last bit from 1 to 0, that is n_1 may have value 0 or 1.

$$n_2 = \mathrm{N}(S_{2y}, S_{2x}) = m, \quad \text{Where} \quad m = \begin{cases} W(S_{2y}) - W(S_{2x}), & \text{if } W(S_{2y}) > W(S_{2x}) \\ 0, & \text{otherwise} \end{cases}$$

that is n_2 may have value 0, 1, 2, $\cdots$, p. Let the number of groups in the third subset is l (say). Then, n_3=N(S_{3y}, S_{3x}) = change in zero or more groups in the third subset from (t+1) 1's to (t+1) 0's ; that is, n_3 may have value 0, (t+1), 2(t+1),$\cdots$, $l(t+1)$.

Lemma 6.2 *If $X \in C_{i+q}$ and $Y \in C_i$, then*

Case (a) If q=0 then

$$N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] = 0 \tag{6.8}$$

Case (b) If $1 \leq q \leq t+d$ then

$$q' = \textit{effective value of } q = \begin{cases} \lfloor (q - F_d)/I_d \rfloor, & \textit{if } q \geq F_t \\ \lfloor q/(I_d + 1) \rfloor, & \textit{otherwise} \end{cases}$$

$$N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq \lceil q'/2 \rceil \tag{6.9}$$

Case (c) If $q \geq t+d+1$ then

$$N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq t+1 \tag{6.10}$$

Proof : *Let $N_{yx} = N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})]$. Here W(X)=$i+q$ and W(Y)=i, i' is effective value of i.*

Case (a) If $q=0$ then X and Y are in the same set which gives $n_1 = n_2 = n_3 = 0$. So $N_{yx} = n_1 + n_2 + n_3 = 0$.

Case (b) If $1 \leq q \leq t + d$ then X and Y are in different sets. So as per the Function 'map':

$$q' = \text{effective value of } q = \begin{cases} \lfloor (q - F_d)/I_d \rfloor, & \text{if } q \geq F_t \\ \lfloor q/(I_d + 1) \rfloor, & \text{otherwise} \end{cases}$$

Now according to the algorithm 'find_code' :

1) for q' odd,

$$n_2 \geq \begin{cases} (q'+1)/2, & \text{where } n_1 = n_3 = 0 \text{ and } i' \text{ is odd} \\ (q'-1)/2, & \text{where } n_1 = 1,\ n_3 = 0 \text{ and } i' \text{ is even} \end{cases}$$

and $n_3 \geq (t+1)$,

$$\text{where } n_2 = 0,\ i' \text{ is odd or even and } n_1 = \begin{cases} 0, & \text{for } q' = 1 \\ 0 \text{ or } 1, & \text{for } q' > 1 \end{cases}$$

So $N_{yx} = n_1 + n_2 + n_3 \geq (q'+1)/2$.

2) for q' even,

$$n_2 \geq q'/2, \quad \text{where} \quad n_3 = 0,\ n_1 \text{ is 0 or 1 and } i' \text{ is odd or even}$$

$$n_3 \geq (t+1), \quad \text{where} \quad n_2 = 0,\ n_1 \text{ is 0 or 1 and } i' \text{ is odd or even}$$

So, $N_{yx} = n_1 + n_2 + n_3 \geq q'/2$

Hence, $N_{yx} = n_1 + n_2 + n_3 \geq \lceil q'/2 \rceil$.

Case (c) If $q \geq t+d+1$ then X and Y are also in different set So according to Function 'map':

$$q' \geq 2t + 1$$

Then as per the algorithm 'find_code':

$$n_2 \geq \begin{cases} t+1, & \text{where } n_1 = n_3 = 0,\ q' \text{ is odd and } i' \text{ is odd} \\ t, & \text{where } n_1 = 1,\ n_3 = 0,\ q' \text{ is odd and } i' \text{ is even} \\ t+1, & \text{where } n_2 = 0,\ n_1 \text{ is 0 or 1},\ q' \text{ is even and } i' \text{ is odd or even} \end{cases}$$

and

$$n_3 \geq (t+1), \quad \textit{where} \quad n_2 = 0,\ n_1 \textit{ is 0 or 1, } i' \textit{ is odd or even and } q' \textit{ is odd or even}$$

So $N_{yx} = n_1 + n_2 + n_3 \geq t+1$. □

Theorem 6.6 : *The code C^* constructed using the procedure find_code is a t-EC/d-ED/AUED code.*
Proof : *As per the* theorem 6.4, *we only need to show that*
$\forall X^*,\ Y^* \in C^* \mid X^* \neq Y^*$,

$$N(X^*, Y^*) \geq t+1$$

Consider $X \in C_{i+q}$ *and* $Y \in C_i$. *Three different cases need to be considered*

Case a) $q=0$
from (1) and (4) , we obtain
$N(X^*, Y^*) \geq N(X, Y) \geq t+1$
and $N(Y^*, X^*) \geq N(Y, X) \geq t+1$.

Case b) $1 \leq q \leq t+d$
As per Function 'map' $0 \leq q' \leq 2t$
from (2) and (5) , we get
$N(X^*, Y^*) \geq N(X, Y) \geq t+1$ *and*
$N(Y^*, X^*) = N(Y, X) + N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})]$
$\geq t+1 - \lceil (q-(d-t))/2 \rceil + \lceil q'/2 \rceil$
i.e., $N(Y^*, X^*) \geq t+1$.

Case c) $q \geq t+d+1$
from (3) and (6) we get
$N(X^*, Y^*) \geq t+1$
and $N(Y^*, X^*) = N(Y, X) + N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq 0+t+1$
i.e., $N(Y^*, X^*) \geq t+1$. □

Hardware implementation :
The algorithm 'find_code' ensures that encoding of information bits in one of the t-EC/d-ED/AUED codes is straight forward and simple. The circuit consists of a t-error correcting/d-error detecting parity check code encoder and another circuit for calculating

the values of $X_1X_2\cdots X_{r^*}$. The values of r^*-bits depend only on the number of 1's in X, W(X). Berger code generator can be used to compute W(X). The circuit for generation of r^* check bits accepts W(X) as the input. It can be implemented in two ways.

I) **Using combinational circuit** : The algorithm 'find_code' calls Function '*map*'. So, we divide our encoding scheme into two parts. First part, Function '*map*' takes W(X) as input and produces $W'(X)$, the effective value of W(X) as output. Second part, the'find_code' algorithm generates r^* check bits with $W'(X)$ as input.

   (a) **Part 1:** Here we have designed the Function '*map*'. In this logic F_t, F_d and I_d are fixed for a given value of t and d. So this circuit consists of one adder, one subtracter, one mod-$(t+d+1)$ counter, one $(2t+1)$ multiplier, one comparator, one mod-I_d divider, one mod-(I_d+1) divider and one 2:1 MUX. The complete circuit is shown in *Fig 6.1*. The size of these components and delay of the circuit depend on the values of t, d and k.

   (b) **Part 2:** This part realizes the algorithm 'find_code'. It generates r^* check bits, which are partitioned into three subsets. A **faster implementation** circuit using a high speed array divider circuit is shown in *6.2*. The LSB of $W'(X)$ is inverted to generate last group of single bit, i.e. S_{1x}. The remaining bits $W'(X)$ are fed into a high speed *mod*$(p+1)$ divider circuit as dividend, D (The p is nothing but *last* mentioned in the function) The 'remainder' output of the divider, i.e. R is fed into a Combinational Logic Circuit to derive the second subset, S_{2x}, i.e. the 'last group'. Every bit of the divider output 'quotient', Q is inverted to generate the bits of each group (i.e. all 0's or all 1's) of third subset, S_{3x}. The size and delay of the circuit mainly depends on the size of the divider dictated by the value of t and k.

II) **Using ROM** : Computation of the check bits of $X_1X_2\cdots X_{r^*}$ can also be implemented using a ROM (or a RAM) which accepts the value of W(X) as address and the contents of this address are the values of r^*-bits. W(X)$\leq n$, thus a ROM of $n+1$ words, is sufficient. For example, for a parity check code C with n=127, a ROM of 128 words is enough. From *Table 6.7* we can find that the number of extra check bits necessary for d=8 is 21. Thus, we need a 128 $\times$ 21 bit ROM. Now according to our code construction method, except the last group and last bit all other groups has a property that in each group all bits have same value i.e. 0 or 1. Therefore, instead of storing $(t+1)$ bits, we can store 1 bit for each of these

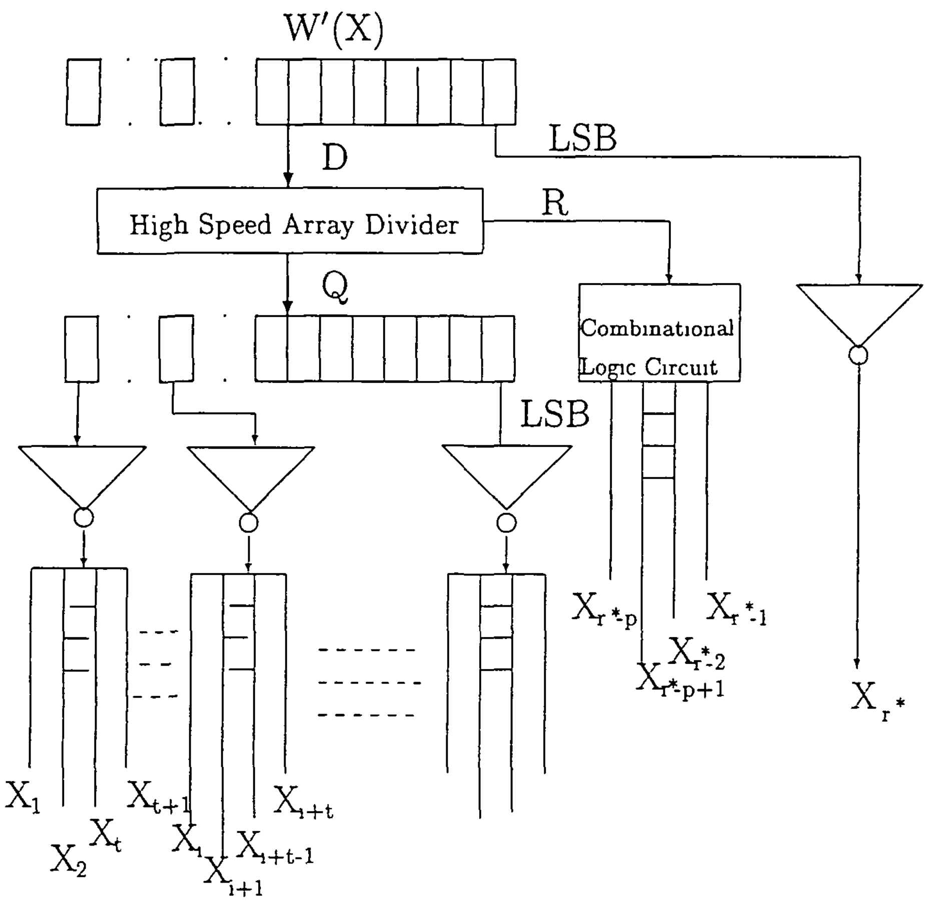


Figure 6.2: Faster Circuit Block Diagram for generation of X_is

groups in ROM. Hence, word length of the ROM gets reduced significantly. Thus, instead of 128×21 bit ROM, we need a 128×9 bit ROM. Further reduction can be obtained if we substitute the ROM with a PLA in order to take into account the fact that many different ROM locations contain identical words. Hence, the hardware implementation of the proposed scheme is significantly lesser than other codes proposed in literature. *Table 6.9* shows the reduction achieved for ROM word size.

6.3.1 Error Detection/Correction Algorithm for the t-EC/d-ED/AUED Codes

Let $Q = XX_1X_2X_3 \cdots X_{r^*}$ be an error free codeword in the proposed t-EC/d-ED/AUED code;
$Q' = X'X_1'X_2'X_3' \cdots X_{r^*}'$ be the received word with some errors in Q;
$Q'' = X''X_1''X_2'' \cdots X_{r^*}''$ be the derived codeword from Q'.

Error Detection/Correction Algorithm :

Step 1 : Compute the syndrome S of X' in the parity check code. Let S corresponds to g multiplicity error. If $g > t$ then the error is only detectable and stop. While for $g \leq t$, correct X' using the correction procedure in the parity check code obtaining X'' as the resulting word.

Step 2 : Compute the values of r^*-bits which correspond to X''. So, $Q'' = X''X_1''X_2'' \cdots X_{r^*}''$. If D$(Q', Q'') \leq t$, then the word Q'' is the correct codeword in C^* and stop, else detectable error (i.e. unidirectional) with multiplicity greater then t has occurred.

The decoding algorithm of the t-EC/d-ED/AUED code has same order of complexity as that of the t-EC/d-ED (n, k) code C. *Example 6.2* illustrates the error detection and correction steps.

Example 6.2 We consider the 2-EC/5-ED/AUED code for the (16,3) 2-EC/5-ED code of *Example 6.1*. The parity check matrix H corresponding to the generator matrix G is given by,

$$H = \begin{bmatrix} 0&0&1&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&1&1&0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&1&1&0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&1&1&0&0&0&0&0&1&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 1&1&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 1&1&0&0&0&0&0&0&0&0&0&1&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 1&0&0&0&0&0&0&0&0&0&0&0&0&1&0 \\ 1&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \\ 1&1&1&0&0&0&0&0&0&0&0&0&0&0&1 \end{bmatrix}$$

Consider the following codeword belonging to the above mentioned 2-EC/5-ED/AUED code

$$Q = \underbrace{100\ 0000001111111}_{X}\ \underbrace{111\ 00\ 0}_{X_t}$$

a) Suppose that one symmetric error has occurred in the above codeword in position marked with $\hat{}$, as shown below.

$$Q' = \underbrace{1\hat{1}0\ 0000001111111}_{X'}\ \underbrace{11\hat{0}\ 00\ 0}_{X'_t}$$

The working of the error correction/detection algorithm is presented below -
By step 1 : We compute the syndrome S of X' in the parity check code

$$S = H.X'^T = [0\,0\,0\,1\,1\,1\,1\,1\,1\,0\,0\,0\,1]^T$$

The syndrome S is equal to the second column of the parity check matrix H. We conclude that a single error has occurred in the second position of X'. The error is corrected and resulting word is $X'' = 100\ 0000001111111$
By step 2 : We consider the value of X_t'' which corresponds to X''. Since W(X'') = 8, as per the algorithm find_code, $X_t'' = 111\ 00\ 0$

Then,

$$Q'' = \underbrace{100\ 0000001111111}_{X''}\ \underbrace{111\ 00\ 0}_{X''_i}$$

Since $\mathrm{D}(Q', Q'') = 2 \leq t{=}2$, the word Q'' is the correct word Q.

b) Next, we consider the occurrence of a symmetric error with multiplicity greater than one in the same word Q. Let the received word be

$$Q' = \underbrace{\hat{0}0\hat{1}\ 00\hat{1}0001\hat{0}11111}_{X'}\ \underbrace{111\ 0\hat{1}\ 0}_{X'_i}$$

By step 1 : We compute the syndrome S of X' in the parity check code

$$S = H.X'^T = [1\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 0]^T$$

Since the syndrome is not equal to a column of H or to the sum of two columns of H, more than two errors have occurred in X' and hence in Q'.

Therefore, the error is detected in the received codeword.

c) Finally, we consider the occurrence of an unidirectional error with multiplicity greater than one in the same word Q. Let the received word be

$$Q' = \underbrace{100\ 000000\hat{0}\hat{0}\hat{0}\hat{0}\hat{0}\hat{0}\hat{0}}_{X'}\ \underbrace{\hat{0}\hat{0}1\ 00\ 0}_{X'_i}$$

By step 1 : We compute the syndrome S of X' in the parity check code

$$S = H.X'^T = [0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1]^T$$

The syndrome S is equal to the first column of the parity check matrix H. We conclude that a single error has occurred in first position (i.e. MSB) of X'. The error is corrected and the resulting word is $X'' = 000\ 0000000000000$

By step 2 : We compute the value of X''_i which corresponds to X''. Since W(X) = 0, as per the algorithm find_code, $X''_i = 111\ 11\ 1$

Then,

$$Q'' = \underbrace{000\ 0000000000000}_{X''}\ \underbrace{111\ 11\ 1}_{X''_i}$$

Since $\mathrm{D}(Q', Q'') = 6 > t = 2$ we conclude that only detectable error has occurred. □

Table 6.5: Proposed 1-EC/d-ED/AUED Codes

r^*	n range				
	d=2	d=3	d=4	d=5	d=6
2	1-5	1-6	1-7	1-9	1-10
3	6-7	7-9	8-11	10-13	11-15
4	8-10	10-13	12-15	14-18	16-21
5	11-15	14-19	16-23	19-27	22-31
6	16-21	20-26	24-31	28-37	32-42
7	22-31	27-39	32-47	38-55	43-63
8	32-42	40-53	48-63	56-74	64-85
9	43-63	54-79	64-95	75-111	86-127
10	64-85	80-106	96-127	112-149	128-170
11	86-127	107-159	128-191	150-223	171-255
12	128-170	160-213	192-255	224-298	256-341

6.3.2 Experimental Results & Comparison with Existing Codes

Number of extra check bits necessary for t=1, 2, 3, 4 with different values of d and different ranges of n are shown in *Table 6.4 to 6.7*. Results show that information rate (k/n^*) of the proposed code varies with t, the number of errors it can correct. From exhaustive simulation study we have observed that for a few cases the scheme noted in [Nikol91] needs lesser number of check bits. However, for higher values of t ($\geq$ 3), the proposed code has information rate higher than that of [Nikol91]. In case of ROM implementation, word length of ROM for this scheme is much less and reduction in word length varies with k and t as shown in *Table 6.8*. This table represents effective t-EC for a t-EC/d-ED code. The major contribution of the proposed scheme is that our encoding/decoding algorithms can be implemented with a simple and faster circuit structure

In the preceding sections (i.e. in *Section 6.2 & 6.3*), a new, generalised class of t-EC/d-ED/AUED codes have been reported. The superiority of the codes is established by comparing it - with some similar type of popular codes. The various properties exhibited by the aforesaid class of codes, have been utilized to make the codes efficient and comparable to the existing codes. However, one important property i.e. every code is of *even-weight*, could not be utilized fully during construction of the codes. Next

Table 6.6: Proposed 2-EC/d-ED/AUED Codes

r^*	n range			
	d=3	d=4	d=5	d=6
3	1-7	1-8	1-9	1-10
4	8-9	9-11	10-13	11-14
5	10-11	12-13	14-15	15-17
6	12-14	14-17	16-19	18-21
7	15-19	18-22	20-25	22-28
8	20-23	23-27	26-31	29-36
9	24-28	28-34	32-38	37-43
10	29-38	35-45	39-51	44-57
11	39-47	46-55	52-63	58-71
12	48-57	56-67	64-77	72-86
13	58-76	68-89	78-102	87-115
14	77-95	90-111	103-127	116-143
15	96-115	112-134	128-153	144-172

Table 6.7: Proposed 3-EC/d-ED/AUED Codes

r^*	n range			
	d=4	d=5	d=6	d=7
4	1-9	1-10	1-11	1-12
5	10-11	11-13	12-15	13-16
6	12-13	14-15	16-17	17-19
7	14-15	16-17	18-19	20-21
8	16-18	18-21	20-23	22-25
9	19-22	22-25	24-28	26-31
10	23-27	26-31	29-35	32-38
11	28-31	32-35	36-39	39-43
12	32-36	36-41	40-46	44-51
13	37-45	42-51	47-57	52-63
14	46-54	52-61	58-68	64-75
15	55-63	62-71	69-79	76-87
16	64-73	72-82	80-91	88-100
17	74-91	83-103	92-115	101-126
18	92-109	104-123	116-137	127-151
19	110-127	124-143	138-159	152-175

Table 6.8: Proposed 4-EC/d-ED/AUED Codes

	n range				
r^*	d=4	d=5	d=6	d=7	d=8
5	1-11	1-12	1-13	1-14	1-15
6	12-13	13-15	14-17	15-18	16-19
7	14-15	16-17	18-19	19-21	20-23
8	16-17	18-19	20-21	22-23	24-25
9	18-19	20-21	22-23	24-25	26-27
10	20-22	22-25	24-27	26-29	28-31
11	23-26	26-29	28-32	30-35	32-38
12	27-31	30-34	33-37	36-40	39-43
13	32-35	35-39	38-43	41-47	44-51
14	36-39	40-43	44-47	48-51	52-55
15	40-44	44-49	48-54	52-59	56-63
16	45-53	50-59	55-65	60-70	64-75
17	54-62	60-69	66-75	71-81	76-87
18	63-71	70-78	76-85	82-92	88-99
19	72-79	79-87	85-95	93-103	100-111
20	80-88	88-97	96-106	104-115	112-124
21	89-106	98-117	107-128	116-139	125-150

Table 6.9: ROM Word-length of the Proposed Code with Effective t-EC Code

	Word length of ROM (in bits) our proposed scheme									
	t=1		t=2		t=3		t=4		t=5	
	reqd.	gain	reqd.	gain	reqd.	gain	reqd.	gain	reqd.	gain
2	2	0	-	-	-	-	-	-	-	-
3	3	0	3	0	-	-	-	-	-	-
4	3	1	4	0	-	-	-	-	-	-
5	4	1	5	0	5	0	5	0	-	-
6	4	2	4	2	6	0	6	0	6	0
7	5	2	5	2	7	0	7	0	7	0
8	5	3	6	2	5	3	8	0	8	0
9	6	3	5	4	6	3	9	0	9	0
10	6	4	6	4	7	3	6	4	10	0
11	7	4	7	4	8	3	7	4	11	0
12	7	5	6	6	6	6	8	4	7	5
13	8	5	7	6	7	6	9	4	8	5
14	8	6	8	6	8	6	10	4	9	5
15	9	6	7	8	9	6	7	8	10	5
16	9	7	8	8	7	9	8	8	11	5
17	10	7	9	8	8	9	9	8	12	5
18	10	8	8	10	9	9	10	8	8	10
19	11	8	9	10	10	9	11	8	9	10
20	11	9	10	10	8	12	8	12	10	10
21	12	9	9	12	9	12	9	12	11	10

section presents a new, specific class of codes, which is designed by full exploitation of such *even-weight* properties (i.e. by using *odd-weight-column* matrix for the *even-parity* check) for the SEC (Single Error Correction) / DED (Double Error Detection) and AUED (All Unidirectional Error Detection) purposes. Such codes are very popular in commercial applications, (as stated in *Section 6.1*).

6.4 Construction of Systematic SEC-DED-AUED Codes

Let C denote a binary linear systematic (n, k) code with D(C) $\geq$ 4 having parity-check-matrix H of odd weight column and C^* be a systematic (n^*, k) SEC-DED-AUED code. For a codeword $X \in C$, the codeword in the proposed SEC-DED-AUED systematic code C^* have the following form

$$\mathrm{X}\ X_1X_2X_3\cdots X_{r^*}.$$

In other words, each codeword of C^* is formed by concatenating a codeword X of C with $X_1, X_2, X_3, \cdots X_{r^*}$, where X_i's are check bits derived by using *Berger's technique.* The appended check bits are used to increase the crossover between two codeword. The proposed SEC-DED-AUED code (i.e. C^*) is constructed from the SEC-DED code(i.e. C) by appending r^* number of extra check bits. These r^* check bits (i.e. $X_1X_2X_3\cdots X_{r^*}$) are divided into groups of 2 bits. Only the last group (i.e. l_g) is one or two bits in width depending on r^* odd or even. Each other -bit group has only two values either 11 or 00 while 'l_g' can have values { 1, 0 } or { 11, 10, 00 } depending on r^* odd or even. The last group can be generated by using a combinational logic circuit. The procedure *compute_code_size* noted below operates on number of information and parity check bits (i.e.n) and gives number of appending bits (i.e. r^*). It also computes size information regarding the number of groups and the number of bits in l_g (1$\leq$ l_g $\leq$ 2).

Procedure compute_code_size(n, l_g, n_g, no_of_bits)

```
/* It takes input n, the number of bits in a codeword in C. */
/* It gives output 'code_size' that are appended to the codeword in C to form $C^*$. */
/* output '$l_g$' gives number of bits in the last group. */
/* output '$n_g$' gives the number of groups excepting the last one. */
    begin
    temp_t ← 3;  /* maximum number of different values that last group of bits can have. */
    temp ← ⌈(n+1)/2⌉;
    $n_g$ ← 0;
    while ( temp > temp_t )
        begin
            $n_g$ ← $n_g$ + 1 ;
            temp ← ⌈temp/2⌉. ;
        end ;
    $l_g$ ← temp - 1 ;
    code_size ←$n_g$ * 2 + $l_g$ ;
  end;
```

The procedure noted below generates the extra check bits i.e. $X_1 X_2 \cdots X_{r^*}$. It accepts the number of 1's of a codeword in C and other information like 'last' and 'max' from the previous procedure as the input.

Procedure compute_code(no_of_one, l_g, n_g)

```
/* It takes input no_of_one, the number of 1's W(X) in the codeword $X \in C$. */
/* Inputs '$l_g$' and '$n_g$' are outputs of the Procedure compute_code_size. */
/* Array 'gvalue' gives output of the procedure, i.e. value of the appended $r^*$ bits. */
  begin
     for i = 0 to  $n_g$ - 1
        int[i] ← 0;
        /* The 'int' array is used to store intermediate values of group of bits. */
     no_of_one ← ⌊no_of_one/2⌋;
     count ← 0;
     shiftleft ← no_of_one mod  ( $l_g$ + 1 ) ;
     /* input to circuit for generation of $l_g$. */
     no_of_one ← ⌊no_of_one/($l_g$+1)⌋;
     Convert_bin ( no_of_one, max, int);
     /* Convert 'no_of_one' to binary of 'max' bits and store in 'int' array. */
     count ← max-1 ;
     while ( count >= 0 )
        begin
           if(int[count] = 1 )
```

Table 6.10: Output of the procedure compute_code for r^*=4 & 5

r^*=4		r^*=5	
number of 1's in C	$X_1X_2\cdots X_{r^*}$	number of 1's in C	$X_1X_2\cdots X_{r^*}$
0	11 11	0	11 11 1
2	11 10	2	11 11 0
4	11 00	4	11 00 1
6	00 11	6	11 00 0
8	00 10	8	00 11 1
10	00 00	10	00 11 0
		12	00 00 1
		14	00 00 0

```
            gvalue[count++] ← 00;
        else
            gvalue[count++] ← 11;
        count ← count - 1 ;
    end ;
  if ( $l_g$ = 1) then
    if ( shiftleft = 1) then
        group[0] = 0;
    else
        group[0] = 1;
  else
        begin
            case shiftleft = 0 : group[0] = 11;
            case shiftleft = 1 : group[0] = 10;
            case shiftleft = 2 : group[0] = 00;
        endcase
    endif
end;
```

Table 6.9 illustrates output of the Procedure compute_code for r^*= 4 and 5. *Table 6.10* shows (12,4) SEC-DED-AUED codes generated by the proposed scheme.

Table 6.11: (12,4)SEC-DED-AUED codes

X		$X_1X_2\cdots X_{r^*}$
0000	0000	11 11
0001	1110	11 00
0010	0111	11 00
0011	1001	11 00
0100	1011	11 00
0101	0101	11 00
0110	1100	11 00
0111	0010	11 00
1000	1101	11 00
1001	0011	11 00
1010	1010	11 00
1011	0100	11 00
1100	0110	11 00
1101	1000	11 00
1110	0001	11 00
1111	1111	00 10

We now show that the above procedure results in a SEC-DED-AUED codes C^*. Let $(X,\ X_1, X_2, \cdots X_{r^*})$ and $(Y,\ Y_1, Y_2, \cdots Y_{r^*})$ be two codewords in C^* constructed using the above procedure.

Lemma 6.3 : *If $X \in C_{i+k}$ and $Y \in C_i$,*

case (a) if k=0 then

$$N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] = 0 \tag{6.11}$$

case (b) if k=2 then

$$N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq 1 \tag{6.12}$$

case (c) if k ≥4 then

$$N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq 2 \tag{6.13}$$

Proof :

Case (a) *If k=0 then X and Y are in the same set. Hence, $X_\imath$ and $Y_\imath$ are identical. So,* $N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})]=0$.

Case (b) *If k=2 then X and Y are in two consecutive sets. According to the Procedure compute_code, crossover between $Y_\imath$ and $X_\imath$ is greater than or equal to one due to either a shift in last group of bit(s) or change of any other group of bits from 11 to 00.*
i.e. $W(Y_1, Y_2, \cdots Y_{r^*}) \geq 1 + W(X_1, X_2, \cdots X_{r^*})$
Hence, $N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq 1$

Case (c) *If $k \geq 4$ then X and Y are in different sets. According to Procedure compute_code, crossover between $(Y_1, Y_2, \cdots Y_{r^*})$ and $(X_1, X_2, \cdots X_{r^*})$ greater than or equal to two, since there is change of bits from '11' to '00' for one or more groups(i.e. two or more crossovers).*
i.e. $W(Y_1, Y_2, \cdots Y_{r^*}) \geq 2 + W(X_1, X_2, \cdots X_{r^*})$
Hence, $N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq 2$ □

Theorem 6.7 : *The code C^* constructed using the procedure find_code is a SEC-DED-AUED code.*
Proof : *As per the* theorem 6.4, *it is only needed to show that*
$\forall X^*, Y^* \in C^* \mid X^* \neq Y^*$, *then*

$$N(X^*, Y^*) \geq 2$$

Consider $X \in C_{\imath+k}$ and $Y \in C_\imath$. Three different cases are to be considered-

Case a) k=0
from (1) and (7) , we obtain
$N(X^*, Y^*) \geq N(X, Y) \geq 2$
and $N(Y^*, X^*) \geq N(Y, X) \geq 2$.

Case b) k=2
from (2) and (8), we get,
$N(X^*, Y^*) \geq N(X, Y) \geq 2$
and $N(Y^*, X^*) = N(Y, X) + N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq 1 + 1$
$i.e. N(Y^*, X^*) \geq 2$.

Case c) $k \geq 4$

from (3) and (9) we get,

$N(X^*, Y^*) \geq 2$

and $N(Y^*, X^*) = N(Y, X) + N[(Y_1, Y_2, \cdots Y_{r^*}), (X_1, X_2, \cdots X_{r^*})] \geq 2$ □

The Proposed SEC-DED-AUED Implementation :

The Procedure 'compute_code' ensures that encoding of information bits in one of the SEC-DED-AUED codes is straight forward and simple. The encoder consists of a single error correcting and double error detecting parity check code encoder and a circuit for calculating the values of the check bits $X_1X_2 \cdots X_{r^*}$. The values of r^* check bits depend only on the number of 1's in X, W(X). Therefore, the circuit for generation of r^* check bits accepts W(X) as the input. Berger code generator can be used to compute W(X). The circuit to generate values of $X_1X_2 \cdots X_{r^*}$ can be implemented in two ways:

I) **Faster Combinational Circuit-based Implementation** (*Fig 6.3 & 6.4*): As per the proposed encoding scheme the complexity in the circuit design depends on the value associated with the last group, i.e. l_g. In 6.3, the circuit for $l_g = 1$ is presented. The LSB of $\lfloor W(X)/2 \rfloor$ is inverted to generate the last group of one bit, X_{r^*}. The other remaining bits of $\lfloor W(X)/2 \rfloor$ are similarly inverted to generate the remaining 2-bit groups. In 6.4, the circuit diagram for $l_g = 2$ is presented. Here, a high speed **Mod** 3 divider circuit is used, which takes $\lfloor W(X)/2 \rfloor$ as input. The remainder output-'R' is fed into a simple combinational logic circuit to generate the all possible 2-bit output (last group), i.e., X_{r^*-1} and X_{r^*}. Every bit of the divider output quotient, i.e. Q is inverted to generate the other remaining 2-bit groups.

II) **ROM-based Implementation :** Calculation of the values of $X_1X_2 \cdots X_{r^*}$ can also be implemented using a ROM (or a RAM) which accepts the value of W(X) as address and the contents of this address are the values of r^*-bits. W(X)$\leq n$, and so a ROM with $n + 1$ words will suffice. For a parity check code C with n=255,a ROM of 256 words is enough. From *Table 6.11*, it is clear that the number of extra check bits is 13. Thus, a 256× 13 bit ROM is required. Except the last group, all other groups has a property that each of the bits in a group has same value, either 0 or 1. Therefore, instead of storing 2 bits, one can store 1 bit for each group in ROM. Hence, word length of the ROM gets reduced significantly. Thus, instead of 256×13 bit ROM, a 256×7 bit ROM is required. Further reduction can be obtained if one substitute the ROM with a PLA in order to take into account the fact that *many different ROM locations contain identical words.*

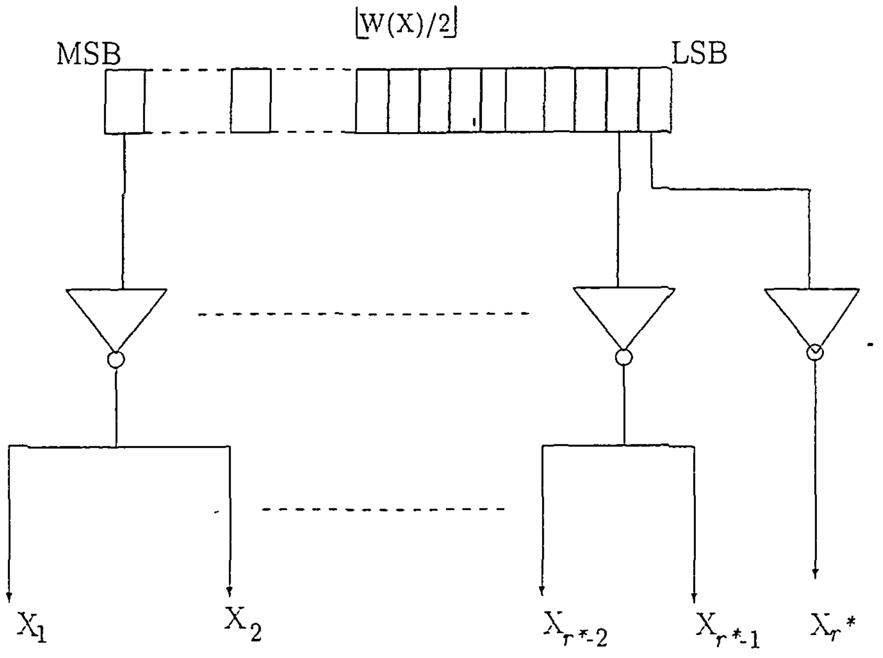


Figure 6.3: Faster Circuit block diagram with l_g=1

6.4.1 Error Detection/Correction Method

Let, $Q = XX_1X_2X_3\cdots X_{r^*}$ be an error free codeword in the proposed SEC-DED-AUED code,

$Q' = X'X_1'X_2'X_3'\cdots X_{r^*}'$ be the received codeword and

$Q'' = X''X_1''X_2'' \cdots X_{r^*}''$ be the derived codeword from Q'.

Step 1 : Compute the syndrome S of X' in the parity check code. If syndrome S have detectable symmetric error then write 'Detectable errors in codeword' and stop, else correct X' using the correction procedure in the parity check code obtaining X'' as the resulting word. Also, if there is no error in X' then take $X'' = X'$ as the resulting codeword.

Step 2 : Compute the values of r^*-bits which correspond to X''. So, $Q'' = X''X_1''X_2'' \cdots X_{r^*}''$. If D($Q', Q''$) $\leq$1 then the word Q'' is a correct codeword in C^* and stop, else a detectable error (i.e.unidirectional) with multiplicity greater then one has occurred.

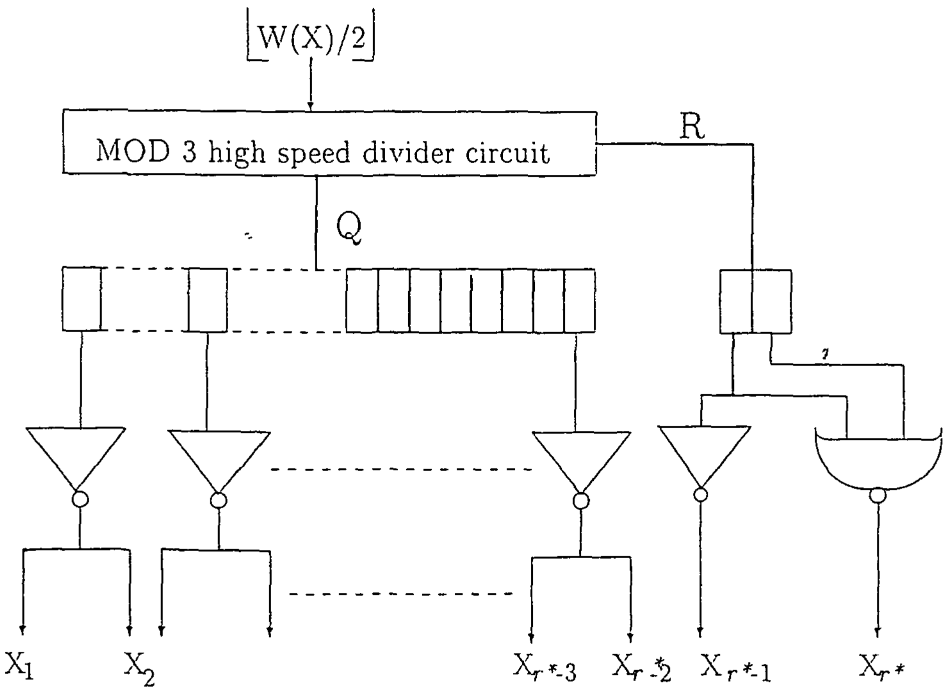


Figure 6.4: Faster Circuit block diagram with l_g=2

Example 6.3 We consider the SEC-DED-AUED code of *Example 6.1.* The parity check matrix H corresponding to the generator matrix G is given by,

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

consider the following codeword belonging to the above mentioned SEC-DED-AUED code

$$Q = \underbrace{10001101}_{X}\underbrace{1100}_{X_t}$$

a) Let us assume that one symmetric error has occurred in the above codeword in position marked with ^ in the received word noted below.

$$Q' = \underbrace{1\hat{1}001101}_{X'}\underbrace{1100}_{X'_t}$$

By step 1 : We compute the syndrome S of X' in the parity check code

$$S = H.X'^T = [1011]^T$$

The weight of syndrome S is odd and it is equal to the second column of the parity check matrix H. We conclude that a single error has occurred in the second position of X'. The error is corrected and resulting word is $X'' = 1000\ 1101$

By step 2 : We consider the value of X''_t which corresponds to X''. Since, W(X'') = 4 we can find X''_t by procedure find_code as $X''_t = 1100$

Then,

$$Q'' = \underbrace{10001101}_{X''}\underbrace{1100}_{X''_t}$$

Since, D(Q', Q'') = 1, the word Q'' is the correct word Q.

b) Next, let us consider the occurrence of a symmetric error with multiplicity two in the same word Q. Let, the received word be

$$Q' = \underbrace{10\hat{1}01\hat{0}01}_{X'}\underbrace{1100}_{X'_t}$$

By step 1 : We compute the syndrome S of X' in the parity check code

$$S = H.X'^T = [0011]^T$$

Since, the weight of syndrome S is even and it is not equal to any column of H, so double error is detected in X' and hence in Q.
Therefore, the error is only detectable.
c) Finally, we consider the occurrence of an unidirectional error with multiplicity greater than two in the same word Q. Let the received word be

$$Q' = \underbrace{1000\hat{0}\hat{0}0\hat{0}}_{X'}\underbrace{1\hat{0}00}_{X_i'}$$

By step 1 : We compute the syndrome S of X' in the parity check code

$$S = H.X'^T = [1101]^T$$

Since, the weight of syndrome S is odd and it is equal to the first column of the parity check matrix H, we conclude that a single error has occurred in first position of X'. The error is corrected and the resulting word is
$X'' = 00000000$
By step 2 : We compute the value of X_i'' which corresponds to X''. Since W(X) = 0, as per the Procedure compute_code, $X_i'' = 1111$
Then ,

$$Q'' = \underbrace{00000000}_{X''}\underbrace{1111}_{X_i''}$$

Since, D$(Q', Q'') = 4$, we conclude that only detectable unidirectional error has occurred
□

6.4.2 Experimental Results and Comparison with Existing Codes

Experimental results and comparison with other codes (i.e. SEC-AUED codes) are tabulated in *Tables 6.11, 6.12 & 6.13*. *Table 6.11* shows values of extra bits for different ranges of k. From a detail experimental study on different ranges of k, it is observed that only for higher values of k, the results cited in [Blaum89] may be better than ours. Further, the codes proposed in [Blaum89] is a SEC-AUED code and the encoding/decoding algorithms of [Blaum89] are more complex than those algorithms presented in this paper. The major advantage of the proposed scheme is that its encoding/decoding algorithm can be implemented with a simple and faster circuit. In case of ROM based implemen-

Table 6.12: Proposed SEC-DED-AUED codes

k range	n range	r^* (size)
4-6	8-11	4
7-10	12-15	5
11-17	16-23	6
18-25	24-31	7
26-40	32-47	8
41-56	48-63	9
57-87	64-95	10
88-119	96-127	11
120-186	128-195	12
187-246	196-255	13

Table 6.13: Comparing the Results of [Nikol86] and [Blaum89] with *Table 6.11*

	SEC-AUED codes				SEC-DED-AUED codes		gain in extra check bits over	
	given in [Nikol86]		given in [Blaum89]		proposed			
k range	n range	r^*	n range	r^*	n range	r^*	[Nikol86]	[Blaum89]
8-10	12-14	6	12-14	6	13-15	5	+1	+1
12-17	17-22	8	17-22	7	18-23	6	+2	+1
19-25	24-30	8	24-30	8	25-31	7	+1	+1
27-40	33-46	10	33-46	8	34-47	8	+2	0
42-56	48-62	10	48-62	9	49-63	9	+1	0
65-87	72-94	12	72-94	10	73-95	10	+2	0
88-119	95-126	12	95-126	10	96-127	11	+1	-1

tation, word-length of ROM for this scheme (tabulated in *Table 6.13*) is significant and the reduction in word-length varies with k.

6.5 Conclusion

In this chapter, two efficient schemes have been introduced to construct t-EC/d-ED/AUED and SEC-DED-AUED systematic codes respectively. The major advantage of these codes are that they can be implemented with simple and faster hardware. In case of t-EC/d-ED/AUED, the detailed experimental study reveals that the results noted in

Table 6.14: Reduction of ROM word length for the proposed code

	ROM Word length (in bits)	
r^*	reqd.	gain
4	3	1
5	3	2
6	4	2
7	4	3
8	5	3
9	5	4
10	6	4
11	6	5
12	7	5
13	7	6

[Nikol91] are better than the proposed codes, only in a limited number of cases. However, for larger value of t, in the majority of cases, the proposed scheme needs lesser number of check bits. Similarly, in case of the SEC-DED-AUED codes, from an exhaustive experimental study on different ranges of k, it has been observed that only for higher values of k, the results noted in [Blaum89] are better. Further, the encoding/decoding algorithms presented in both the schemes are simpler than most of the existing schemes. In case of ROM based implementation, both the schemes significantly reduce the ROM word-length.

The study carried out so far and the schemes developed for authentication, data security and error correction/detection suggest that these could be applied appropriately to several application areas such as distributed database applications, authenticated key exchange, etc (as discussed in *Chapter 1*). Some of the areas where these schemes also could be applied successfully are image and multimedia database applications. Such investigations and applications might lead to enhancement of knowledge not only in the areas of authentication, data security and error control coding, but also to look into the factors such as : design of appropriate means for representations e.g. data model, systems' architecture etc. The following chapters highlight the findings of such applications into the areas of hypermedia application development and image database systems design.

Chapter 7

RMDM and Its Drawbacks

7.1 Introduction

The investigations reported in the previous chapters (*Chapters 2,3,4,5 & 6*) indicated that one of the major applications of authentications, data security and error correction/detection schemes is in multimedia database. However, an integral and essential part of the multimedia database design is to model the real-world application domain in a conceptual manner using a suitable data model. A data model is an abstraction of a portion of real-world situation. It is a group of concepts that help us to specify the structure of a database and a set of associated operations for specifying retrieval and updates on the database [Hughe91]. Basically, one can have two types of data models : high level or conceptual or object-based data model, and low-level or physical or record-based data model. High level data models e.g. E-R model [Chen76] provides concepts that are close to the way many users perceive data, and it uses the concepts such as entities, attributes and relationships. Whereas, low-level data models e.g. Relational, Hierarchical or Network data models provide concepts that describe the detail of how data is stored in the computer. E-R model is popular and often used in representing database schema for any business and industrial database applications. However, with the enormous growth of database technology and in the context of several special database applications, such as CAD/CAM, Cartographic, Geological, hypertext & hypermedia applications, it has been observed that E-R model is not sufficient enough to model their complex object-types and association patterns, in a conceptual manner. In this background, several other semantical data models, including the Object-Oriented and Functional data models have been proposed to meet those complex modelling requirements. However, from the experience it is felt that in the context of the hypermedia applications, which bear a peculiar behaviour, particularly from the navigational aspects, their modelling requirements demand a separate class of data models, which may be capable enough in modelling these complex navigational patterns for any scale of hypertext/hypermedia applications in a conceptual manner. RMDM (Relationship Management Data Model) [Isakw95] is one, which belongs to this class of data models and has been proposed for a class of hypermedia applications exhibiting regular and static structure and which require frequent updating due to its volatile nature of data. Apart from RMDM, several other data models for hypertext/hypermedia systems also have been proposed in this regards. Next subsection presents a brief review of the existing data models.

7.1.1 Reviewing Hypermedia

The terms "hypertext" and "hypermedia" are often used quite interchangeably. Hypertext in the strict sense only applies to text-based systems; hypermedia is simply the extension of hypertext to include multimedia data. The invention of both the terms is credited to Ted Nelson in 1965. Nelson presented a vision of the type of knowledge management environment that hypertext can help to support. The inspirations behind his ideas were due to V. Bush and D. Englebart. Bush proposed a theoretical design for a system, which he called as Memex (Memory Extender) that is known as a hypertext system. Douglas Englebart is credited with the invention of word processing, screen windows and the mouse and thus inspiring the developments in graphical user interfaces that have taken place over the last twenty years. By the end of the 1980s, the state of the art in hypertext systems was best represented by intermedia, but the research community had recognized the need for a hypertext reference model for some while [Wendy96]. Campbell and Goodman [Cmpbl88] suggested a three layered model for hypertext systems essentially consists of the presentation level (user interface), the hypertext abstract machine (HAM) level (*nodes* and *links*) and the database level (storage, shared data, network access etc.). The HAM was used by both the Neptune and CAD systems [Desle86] and the dynamic design [Biglw88] for providing the hypertext object layers. In December 1989, in the hypertext standardization workshop, conducted by NIST (National Instt of Standard and Technology), several new models were proposed. The standard that emerged was Dexter reference model ([Halas90, Halas94]). This model attempted to capture the important abstractions found in a range of hypertext systems that existed at the time, such as Augmented, Intermedia, Hypercard, Hyperties, KMS, Neptune and Notecards, Sun's Link Service [Pearl89] etc. Like the HAM model, the Dexter model divides the hypertext systems into three layers. The *runtime layer* contains the basic hypertext functionality, the *storage layer* contains the network of link and nodes specifications that represent the structure of the particular hypertext. The *within component layer* is the content of the particular component, node, document or frame. Dexter sensibly made no attempt to model the structure of the component leaving this to other modelling tools such as ODA [Wendy96]. Gronbaek and Trigg [Grnbk92] criticized Dexter model and they pointed out that : (i) It should not possible with Dexter to create dangling links made for restrictive interfaces to link creation and addition; (ii) these are multiple orthogonal concepts directionality of links (such as the directionality of a 'supports' typed link and the direction of traversal) ; (iii) it is necessary to extend the Dexter model to support the long term transaction, locking and event notification so that a co-operative hypermedia systems could be designed.

Legett and Schnase [Leget94] point to the shortcomings of using Dexter as an interchange standard, particularly its incomplete specifications of multi-headed links, the problem with a dangling link creation when importing partial hypermedia and the failure of the model to distinguish between separate webs as implemented in Intermedia. They also point out that the model has no concept of the semantics of arrival and departure when following links and that the notion of the composites and their consistencies is complete, as also has been noted by others [Grnbk92]. They suggested that anchors belonging the link services domain rather than within the applications (or components) domains.

A further problem of the Dexter model is that it contains no specifications of how to deal with temporal event specifications, which is well covered by the Hytime standards [Wendy96], and partly for this reason it is probable that Hytime will be seen as a more complete hypertext interchange standard. In October, 1992 and in November 1993, two other workshops held with the motivations behind that the current generations of hypermedia systems would not scale to deal with very large information. Repositories such as those that would be found in digital libraries and the purpose was to separate the area of the research that concentrates on hypermedia systems implementation issues from other issues, such as user interfacesm, evaluations and usability studies. The area of concentrations were : models and architectures, node. link and structure management, version control, concurrency control, transaction management and notification control.

As an outcome, it was considered hypermedia systems architecture to consist of three layers : storage (that provides persistent storage for the data model abstractions (nodes, links, anchors, contexts, etc.)), hyperbase (that provides the hypermedia data model to the application) and application layer (contains typical applications such as text and graphics editors, mail tools and multimedia presentation tools). However, there was a majority agreement that the Dexter model could not provide the flexibility or richness required by the hypermedia community. In this background, several other systems have been evolved, such as : GMD-IPSI's CHS and CoVer [Schut90], SP3 and HB2 [Leget94, Kacmr91, Schns93a], ABC and DGS [Shklf93, Smith91], Hyperform [Wiil92], DHM [Grnbk92], DHT [Noll91], HB3 [Schns93b], Trellis [Stots89], HDM [Garzo93a], HDM2 [Garzo93b], RMDM [Isakw95] etc. HDM [Garzo93a] is appropriate for describing the structure of the application domain and HDM2 [Garzo93b] is a successor of it. HDM and HDM2, both describe representation schemes but provide little information on the procedures for using these representations in the design process; that is they do not describe a hypermedia design and development methodology. RMM [Isakw95] is an extension work of [Balas94], which focus on developing a graphical representation based step-by-step procedure for hypermedia design and development. RMM describes RMDM as an application data model which is the cornerstone of RMM methodology. The objective of the RMDM techniques is to represent data at a higher level of abstraction. It is basically an extension of the popular object-based E-R model where the navigation approach is based on the HDM and its successor, HDM2. The extended part of E-R modelling is very well defined and a strong set of modelling notations is provided. To study RMDM, several specific application domains have been selected for implementation. It has been observed that RMDM is not exactly fulfilling the implementation of some of those complex application domains. This chapter projects one of such applications where RMDM fails to express some of the situations conceptually. To overcome these drawbacks, some remedial suggestions have been put forth, which ultimately indicates towards extension of the present RMDM. The implementation results based on the extended RMDM also have been reported in this chapter. Next section discusses an example multimedia application to study the RMDM.

7.2 Multimedia Example : A Real Life Ecosystem

Several specific hypermedia application domains were chosen to investigate the expressiveness of RMDM. One of such applications is a real-life Ecosystem from Keoladeo National Park, Rajasthan, India [Shukl96]. A common nature of any ecosystem is that they undergo succession and aging process accompanied by significant changes in their structure and function. This natural process of succession could be accelerated by unusual changes in the environment due to which some of the species may die out or eliminated from the habitat. It leads to uncontrolled growth rate of a certain harmful species that had been otherwise kept in check by eliminated species. In the aquatic environment, a typical example of growth rate of noxious weeds (Eichhornia, Salvinia), which is harmful to other species, has been pointed out by many investigators [Salim86] Similar situations also exist in the wetland area of Keoladeo National Park, at Bhatarat-pur (Rajasthan) in India where habitat is degrading due to uncontrolled growth of some wild grasses such as Paspalum Distichum. This leads to decrease in the growth rates of other valuable species (e.g. migration of Siberian Cranes, other birds, etc) which are unable to compete. To control the situation, it becomes very important for the ecologists to collect, store and present realistically - an up-to-date statistic of the huge number of valuable Species, Vegetations and other Ecological entities - their attributes and features, such that necessary preventive measures may be taken. Thus, it could be an ideal multimedia application to study the proposed RMDM data model. Besides, the other reasons behind the selection of this case for illustration are :

R1 It has numerous classes and sub-classes of entities, definable relations which exhibit a regular structure.

R2 It requires regular updating.

R3 It includes some recursive structure-based, generalized entities.

7.3 Modelling the Ecosystem using RMDM

RMDM is a hypermedia application data model that provides a language for describing the information objects and the navigation mechanisms of any regular and static structure-based hypermedia application. To model the information about the application domain, RMDM provides three types of primitives: *E-R, RMD* (Relationship Management Domain) and *Access domain* primitives. To represent and describe the physical or abstract ecological objects, e.g. species, vegetation etc. and their attributes as well as their associations among the entity types- the *E-R* domain primitives can be used. Among the large set of attributes associated with an entity, for natural and practical representation, this model suggests to slice those attributes into several meaningful logical groups, which are here termed as RMDM *Slices.* To access those slices, the RMD domain primitives are provided. To navigate through and access the slices, RMDM supports six types of access primitives, such as : *unidirectional link, bi-directional link, grouping, conditional-indexed, conditional guided* and *conditional indexed guided tour.* The appropriateness of the use of these primitives is based on the type of attributes associated with the RMDM slices, its uniqueness and dependencies among the other attributes. Unless mentioned otherwise, the notations used in this chapter are identical to [Isakw95].

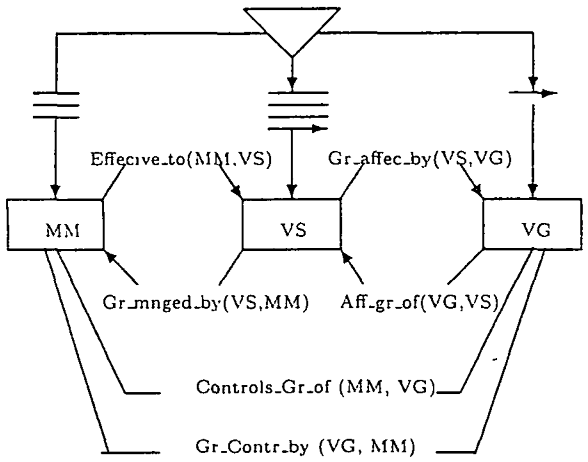
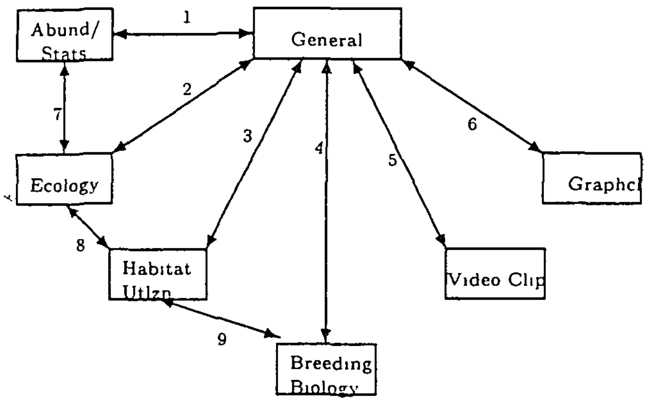


Figure 7.1: A Partial RMDM Diagram for the Ecological Application

Figure 7.1 shows a partial RMDM diagram of the concerned ecological application. It represents the navigation mechanism from the hypermedia users' point of view in the concerned domain. In the diagram, for simplicity and to avoid cluttering, slices of each entity are not included, only the key attributes are shown. Accesses to various entities are provided via the access primitives selected to associate the slices. For example, to access the large collections of uniquely addressable VS (Valuable Species), a conditional indexed-guided-tour is made possible with predicate *affects_gr_of*(VG,VS) from VG (Vegetation) entity as shown in *Figure 7.1*. Similarly, the reciprocal index *gr_affected_by* (VS,VG) accessible from VS is of type conditional guided. These two indices are representing a many-many relationship. Similarly, from the VS there will be accesses of type conditional indexed tour to the entity MM (Management Method) with the predicate *gr_mnged_by* (VS, MM) and through this access, the users can detect the suitable method(s) for management of the species. Appropriateness of an access primitive to a specific entity type entirely depends upon the nature of entity type and their associated attribute slices. To access the management method details, from the main menu (the grouping mechanism on the top of the figure) conditional indexed tour is provided and accordingly, to access the set of vegetation affecting the growth of the valuable species, conditional guided tour has been found to be suitable.

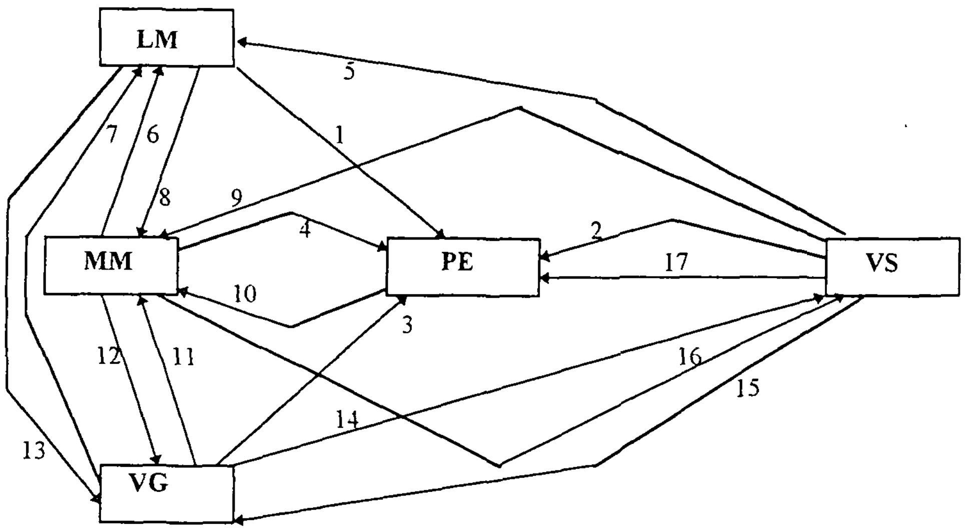
The user will navigate the application through the various *entity-slice-heads* (i.e. a distinguished slice used as a 'default' to anchor links coming into the entity). Access to and presentation of the entity domain entirely depends on how slices are designed and organized into a hypertext network. In *Figure 7.2*, slices of the species entity are shown. Numeric Codes (appearing in the subsequent figures) associated with the directed edge represent various types of associations between the connecting attribute slices.



1	Abund_&_Var_of(Species)	2	Eco_Comp_of(Species)	3	Hab_Utlzn_of(Species)
4	Br_Bio_of(Species)	5	Video_of(Species)	6	Graphical_of(Species)
7	Abund_Stat_pf(Species)	8	HAb_Utlzn_of(Species)	9	Br_Bio_of(Species)

Figure 7.2: Slice Diagram for *'Species'* Entity

Figure 7.3 shows the E-R diagram (an abstract representation with presenting the attributes) for the Ecological application. The various entities such as, Limnology (LM), Physical Environment (PE), Valuable Species (VS), Management Method (MM) and Vegetation (VG) are shown in rectangles. The relationship *bears, possesses, managed_by,*



1 : *bears* (PE, LM),	2 : *possesses* (PE, VS)	3 : *possesses* (PE, VG)
4 : *managed_by* (PE, MM)	5 : *supp_gr_of* (LM, VS)	6 : *results* (LM, MM)
7 : *controls_gr_of* (LM, VG)	8 : *is_a_result_of* (MM, LM)	9 : *effective_to* (MM, VS)
10 : *concerns_to* (MM, PE)	11 : *controls* (MM, VG)	12 : *can_contr_by* (VG,MM)
13 : *gr_contr_by* (VG, LM)	14 : *affects_gr_of* (VG, VS)	15 : *gr_affected_by* (VS, VG)
16 : *managed_by* (VS, MM)	17 : *found_in* (VS, PE)	

Figure 7.3: ER Diagram for the Ecosystem, Keoladeo National Park

supp_gr_of, *results*, and so on, are shown by directed line.

In the context of the above application domain- it has been found that the proposed RMDM fails to express some of the situations conceptually. To model those situations, either, some extra identifications (for entities) have to be introduced or some relationships that are not relevant to the concerned domain have to be made, which ultimately results in *overspecification* [Hofst93]. The three main drawbacks that RMDM suffers from : *firstly*, the application domain has numerous additional sub-groupings of its entities that are meaningful and need to be represented explicitly because of their significance to the concerned application. The present RMDM does not provide sufficient modelling constructs to model the situation in an adequate manner; *secondly*, the concerned domain includes some complex object-types. which are composed of according to some specific recursive rules and to model these recursively composed objects special modelling constructs are essential. However, the present RMDM does not provide any scope to model these objects naturally; *thirdly*, a major step in RMDM modelling is to slice each *entity* based on the logical relationships exist among the attributes of it. However, it does not provide any guidelines, how to handle composite entity-based presentations (i.e presentations with multiple slices from different entities) as also detected in [Balas95]. Thus the drawbacks of RMDM can be summerised as :

D1 Lack of modelling capabilities to naturally model those objects that have many subtypes defined according to some specific *Subtype Defining Rules.*

D2 Lack of powerful constructs to model those complex objects constructed based on some specific rules of recursive nature, such as multimedia document (On line documentation is one of the important aspects of any multimedia application development).

D3 Lack of modelling guidelines to represent *composite slices* combined from slices of different entities.

Next section discusses some of the necessary remedial measures to be taken against these drawbacks in details.

7.4 Modifications in the RMDM

The concerned application domain has numerous additional sub-groupings of its entities that are meaningful and need to be represented explicitly because of their significance. For example, the entity *'species'* has numerous collections of sub-classes as shown in the *Figure 7.4* and sometimes, it is necessary for some instances of sub-classes to be navigated through their certain slices only. Thus, in other words, in case of some special entities, the navigation mechanisms are governed by some specific Subtype Defining Rules [Bomel91]. But with the present RMDM modelling constructs, it is difficult to model such situations conceptually. Appropriate utilization of specialization concept (as an analogous presentation found in the extension of NIAM model [Hofst93]) with additional modelling notations for it, can solve this problem.

7.4.1 Modelling with Specialization

In case of those specific instances of an object type belonging to a huge subtype hierarchy, when only some particular slices have to be navigated- to avoid 'loss in hyperspace' as well as violation of the 'conceptualization principle', it is very essential to organize the

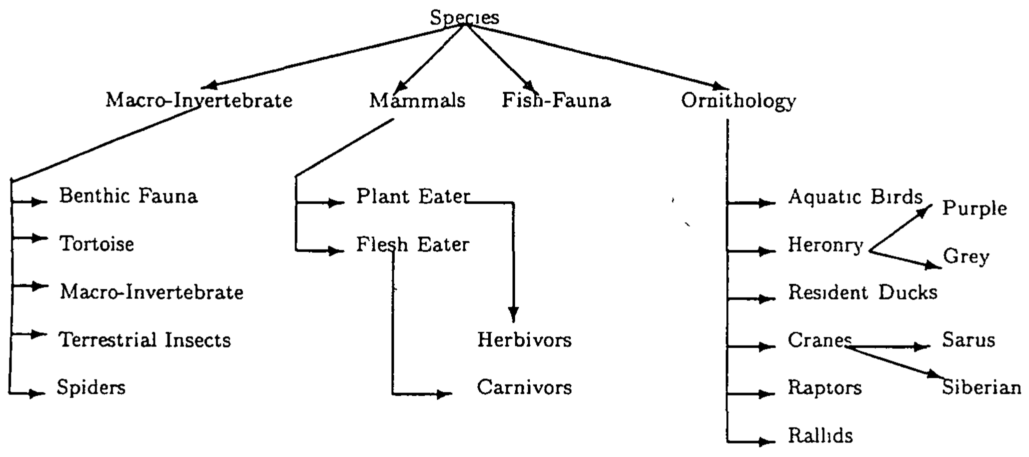


Figure 7.4: Subgroup Collections of *'Species'* Entity

objects as per the subtype defining rules. The *axioms* relevant to it have been reproduced here, from [Hofst93] for the clear understanding of the concept:

S1. Specialization is a partial order (asymmetric and transitive) concept, which prevents occurrences of cycles in the structure. (*Axioms (2,3)*).

S2. Only atomic object types can occur as subtypes and each subtype inherit their identification from their pater familias[1]. (*Axioms (4,5)*).

S3. Subtype hierarchies for label types and entity types do not interfere (*Axiom (6)*), where, label types are species_index, name, etc. of entity type species.

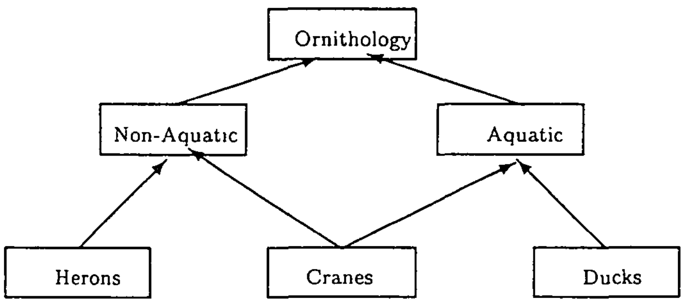


Figure 7.5: A Specialisation Hierarchy from the Ecological Domain

[1]A specialization hierarchy is in fact not a hierarchy in the strict sense, but an acyclic directed graph with a unique top. The top is referred to as the *'pater familias'*

Figure 7.5 shows an application of specialization. In this figure, the dotted directed edge is representing the specialization relationship. Properties in a specialization hierarchy are inherited 'downward' (refer S1) e.g. *Aquatic* subclass inherits their identification from *Ornithology*, but the reverse is not true (refer **S1**) and each of these subtypes is not interfering (refer **S3**). Thus, during RMDM modelling, applying the concept of Specialization and by utilizing the Subtype defining Rules [Bomel91], the first drawback could be overcome.

The *second drawback* mainly related with some entities that are complex in nature due to their nested composition structure. One such entity very common to any hypermedia database application is *multimedia document* (not covered in the *Figure 7.1 & 7.3*), which has its own layout structure. The layout structure of a document precisely defines where information is to appear on a plane surface when displayed on a terminal or printed on paper and also organizes it to aid in understanding. Document structure is most often expressed in terms of the abstract objects, hierarchical links between them, ordered and unordered objects and shared components. One can describe and represent this organization graphically by using a tree structure. For the sake of understanding, a systematic representation (layout structure) of a multimedia document analogous to one reported in [Karmc96] has been reproduced in *Figure 7.6*.

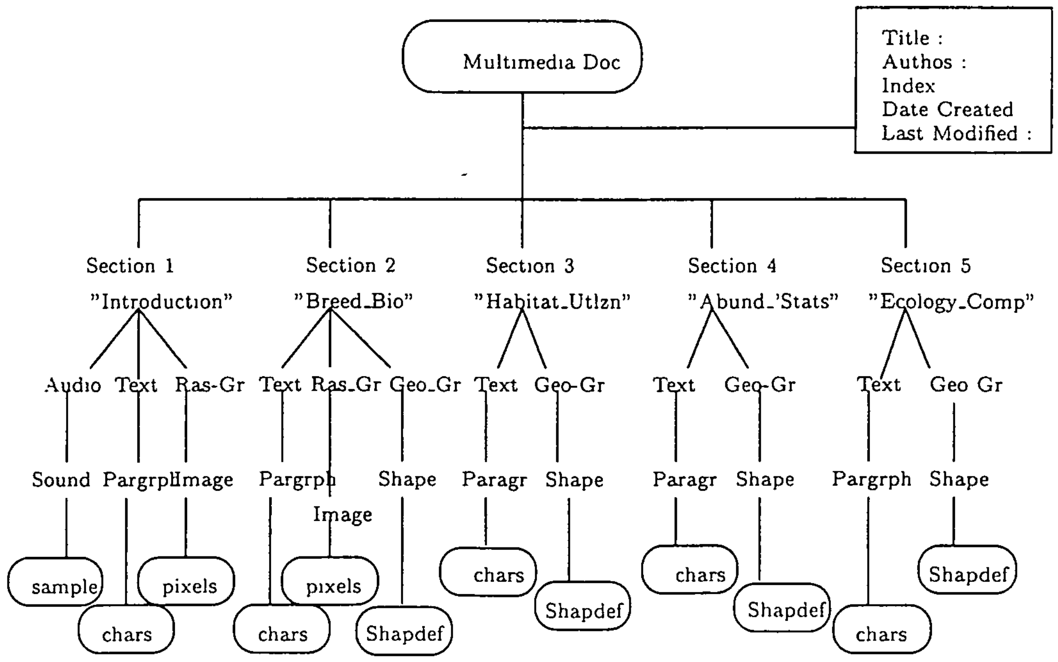


Figure 7.6: A Partial layout structure of the Multimedia Document

Current research on document representations are carried out in two areas : one focusing on *passive* documents; the other on *active* documents. In *passive* documents, the author integrates continuous media simply by representing them in a static visual form, later activated by the user. For example, the static visual form of a video sequence is a frame and the static visual form of audio is an icon. These can be embedded into traditional document type (i.e. those containing only static media type) and are played only upon user activation. In case of *active* document - since it requires to integrate the continuous media in the document itself, each media item has to be treated as an object to be presented in time for a duration. Basically, the structure of such multimedia document provides a set of rules for creation, manipulation and representation of this logical tree-shaped structure. As stated in [Karmc96], representing such document structure in terms of levels of hierarchies of abstract objects, has several purposes :

- to facilitate integration of diverse media types.
- to organize the document so that one can easily understand and manipulate it.
- to permit advanced queries based on attribute values and object types etc.

In general, such entities are modelled according to some specific rules. To model such entity with the present RMDM, it requires some additional modelling constructs with appropriate utilization of *generalization* concept. The next section describes the modelling of complex nested structure using generalization concept.

7.4.2 Modelling with Generalization

Generalization is a mechanism that allows for the creation of new object types by uniting existing object types. The following *axioms* and *lemmas* [Hofst93] provide the basis of generalization :

G1 Generalization is a partial order (asymmetric and transitive) concept, which prevents occurrences of cycles in the structure. (*Axioms (7,8)*).

G2 Only atomic object types can participate in generalization and each generalized object type requires the identification from its instances (*Axioms (9,10)*).

G3 If α is a generalization of β or β is a generalization of γ then α is not a specialization of β, i.e., α Gen $\beta \vee \beta$ Gen $\gamma \dashrightarrow \neg\alpha$ Spec β .

Thus, during construction of the 'hypertext network' with the present RMDM, by appropriate utilization of the aforesaid *axioms* and *lemmas*, one can easily overcome the second drawback. In *Figure 7.7*, the modelling of multimedia document based on generalization concept has been depicted. Here, the generalized object-type *document* is covered by its abstract object types :*title*(i.e. stream of characters), *Section* (i.e. stream of characters), *Audio type* (i.e. icon), *Raster Graphics*(i.e. cluster of pixels) and *Geometric graphics*(i.e. shape definitions), which are atomic object types in nature and the document type borrows the identifications from these atomic object types (refer G2). In generalization, each generalized object is covered by its constituent object types or specifiers. Hence, a decision criterion as in the case of specialization (the subtype defining rule) is not necessary. Furthermore, properties in a generalization hierarchy are inherited 'upward', but the reverse is not possible, i.e. properties of *Section* will not be inherited from *Document* (refer G1) which is the case of specialization. Finally, *Section* is a Generalization of *Characters* and at the same time *Section* is a Specialization of *Document*, but *Character* is not a Specialization of *Document* (refer G3). In most of the data models, it is seen that generalization is treated as an inverse of specialization. However, it is contradictory, as they are originating from two different axioms in set theory and they have a different expressive power. For the sake of validity, an illustration is given in in *Figures 7.8 & 7.9.*

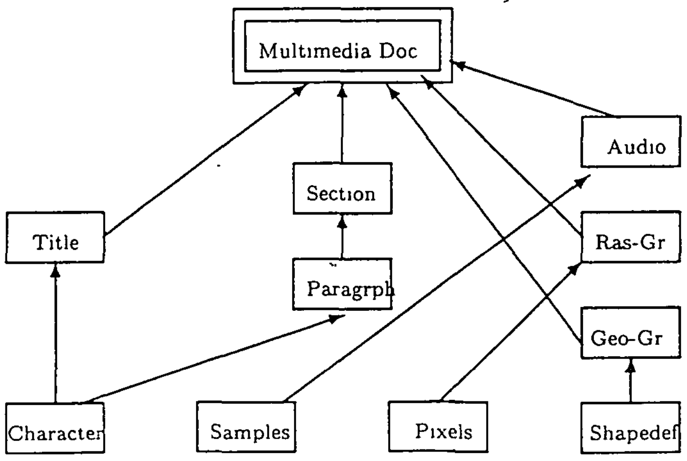


Figure 7.7: Modelling Multimedia Document using Generalisation

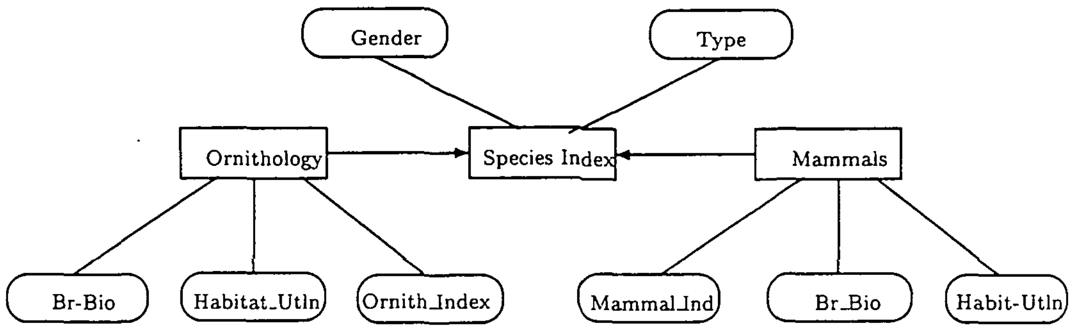


Figure 7.8: Specialisation instead of Generalisation

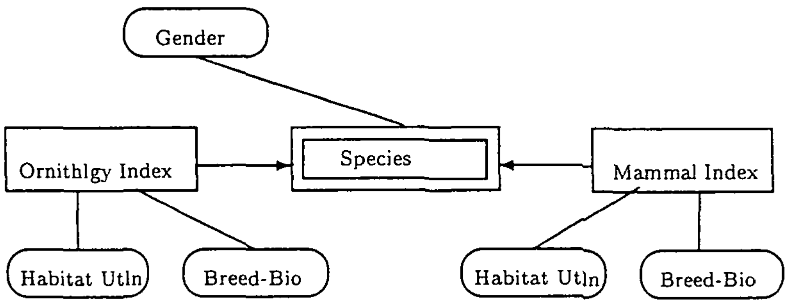


Figure 7.9: Modelling using Generalisation

Consider, a navigation problem for listing all female Species, in the concerned Ecological domain. A Species is either an instance or a set of instances of Ornithology or Mammals. An instance of Ornithology is identified by an 'Ornithology Index', whereas, a Mammal is identified as a 'Mammal Index'. In this case, Species is the generalization of Ornithologies and Mammals. Species have attributes 'Gender' and each of its instances has attributes 'Habit_Utlzn', 'Breed_Bio' and 'Index'. With the present RMDM, based on the 'specialization' concept such a situation could be modelled as shown in *Figure 7.8*, where, the dashed directed line represents the specialization relationship and the oval shapes are used to represent the attributes associated with each entity. But, such a modelling results in *overspecification*, due to the following reasons:

- It introduces an extra attribute 'Species Index' to identify the Species.
- Differentiating both the types of Species instances as total and disjunct, it necessitates a special attribute 'Type', to determine the type of the Species.

However, this situation could be modelled in an appropriate manner by utilizing the 'generalization' concept (as shown in *Figure 7.9*). In the figure, the directed solid lines are used to represent the generalization relationship and the double-outlined rectangle is used to represent the entities of 'generalized type'. As *Species* inherits their identification from its sub-species, such as, *Ornithologies* and *Mammals* (refer **G2**) and as these instances are expressed as total and disjunct (as per the 'Generalization rule'), both of the attributes 'Species Index' and 'Type' are no longer required. Hence, in some specific cases, instead of specialization, only the generalization concept is applicable and appropriate.

The *third drawback* deals with modelling of multiple slices from different entities to be appeared in the same window. In most of the hypermedia applications, it is often necessary to compose multimedia scenes based on multiple *slices* (possibly with *minimized* information) from different *entities*. Such slices with minimized attributes (referred here as *'representative slice'*) may be used as an 'window' to the corresponding *head-slice* for accessing the entity detail. Similarly, considering the all slices from different entities composing a multimedia scene may togetherly be defined as a *'composite slice'* for better organizing the 'hypertext network'. However, the present RMDM does not provide any guidelines - how to handle such situations. As such, these two additional slices (i.e *representative* & *composite* slices) have been introduced in the present RMDM to model any complex navigation pattern in a better manner. The *composite* slice used in the extended RMDM is analogous to the *'cross-entity'* slice found in [Balas95]. The *representative* slice is defined to be the summarized attribute information, to represent all the slices of an entity. Thus apart from the *'head slice'*, the modified RMDM includes one more additional slice for each entity to be used as an anchor to help in determining the appropriate of information to be displayed and hence by increasing the *local coherence*.

For the verification as well as for validation of the extended RMDM, a prototype Multimedia Information System (MMIS) has been developed based on this model as well as the MMIS architecture reported in [Bhatt96b]. A brief overview of the implementations is reported in the next section.

7.5 Overview of the Extended-RMDM Implementation

A prototype software has been developed based on the extended RMDM, and implemented in HTML and Visual Studio Platform. A partial view of the high level access structure mechanism utilized in the proposed software is reported in *Figure 7.10.* Access Structures are designed by grouping the items according to their nature and types of interest. Basically, the hypermedia end-users can select any of the six menu options to access the application database. In case of Species entity, accesses are organized into further subgroups according to the 'Index' and 'group type' associated with each instance of the Species as well as the 'name' and 'habitat type' attached with each Species. Similarly, also in case of the large set of Vegetations, sub-menu structures are provided.

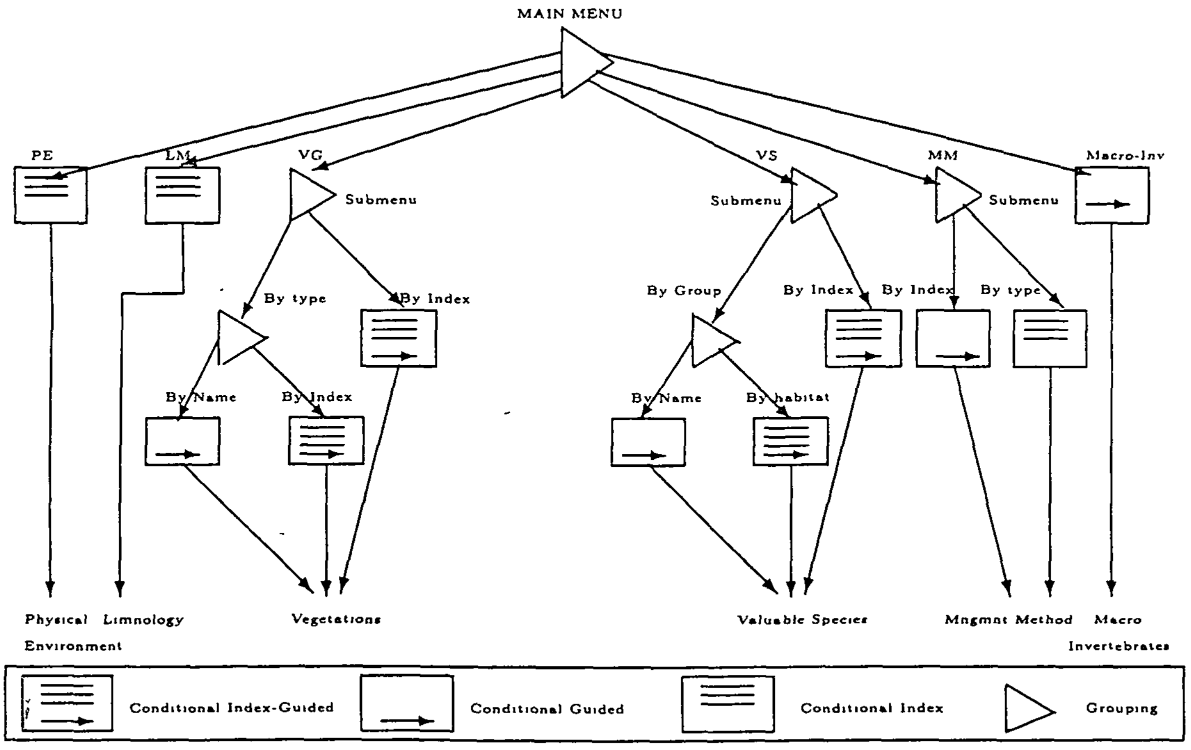


Figure 7.10: Partial view of the High Level Access Structure Mechanism

The HTML implementation shows the hypermedia presentation of the material collected in the Keoladeo National Park Project [Salim86]. This collection includes textual, graphical and image data about the species, vegetations, limnological and environmental statistics etc. The application is intended to present the material of the project in two modes : *contextual reference* based and *graphical RMDM* based. *Figure 7.11* shows the introductory screen, also serves as the main-menu, where user has a choice of viewing the materials- topic wise either from the textual content (by selecting any anchor) or from the graphical presentation (as shown in the upper-right corner, where each entity or relationship box serves as an anchor). Each screen may contain references to documents and photographs that are part of collections. For instance, by clicking on the contextual topic 'Ornithology', one can find out more information about the species belong to the 'Ornithology class', about their population, habitat-utilization, breeding biology, etc. *Figure 7.12* shows information about the Siberian crane. Clicking on the photograph of the crane, a larger view can be obtained; in the textual part, the brightened word gives contextual information regarding the population, breeding biology, Food and feeding habitats, etc.

7.6 Conclusions

In this chapter, a hypermedia application data model, called RMDM has been studied and analyzed in the context of a real-life ecological application. From the analysis, it has been found that the above model fails to express some of the situations of the concerned application domain conceptually. This chapter projects the drawbacks of the RMDM and subsequently, the necessary remedial measures have also been suggested. With the proposed remedies, the present RMDM has been extended and the extensions are validated by several example hypermedia domains. A prototype software for the Keoladeo National Park, Rajasthan, India, has also been developed based on the extended RMDM and it has produced satisfactory results.

With the tremendous growth of multimedia technology, image and video databases have consequently become the central component of many future applications. The efficiency of an image or video database system mainly depends on its performance in answering the queries. To handle the complex queries over the large repository of image or video databases, a suitable and user-friendly database system architecture is essential. The next chapter presents a new image database systems architecture which would cover a wide variety of applications.

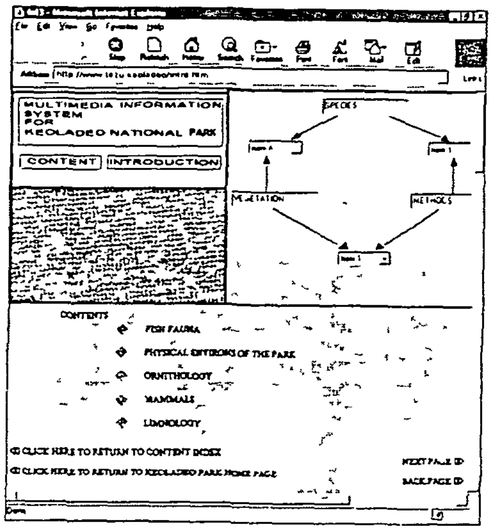
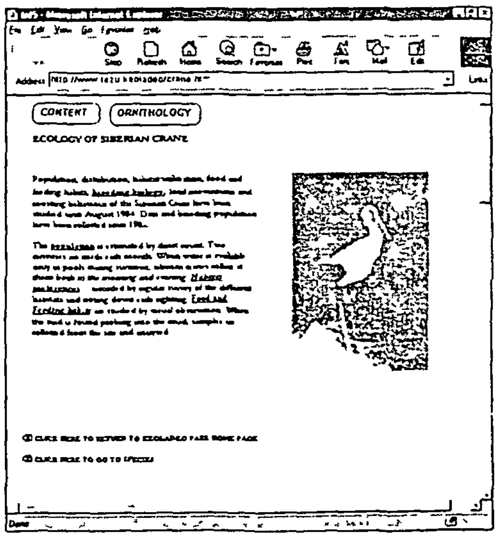

MULTIMEDIA INFORMATION SYSTEM FOR KEOLADEO NATIONAL PARK
CONTENT
INTRODUCTION
SPECIES
VEGETATION
METHODS
CONTENTS
FISH FAUNA
PHYSICAL ENVIRONS OF THE PARK
ORNITHOLOGY
MAMMALS
LIMNOLOGY
CLICK HERE TO RETURN TO CONTENT INDEX
CLICK HERE TO RETURN TO KEOLADEO PARK HOME PAGE
NEXT PAGE
BACK PAGE

Figure 7 11 RMM based prototype implementation of Keoladeo Park


CONTENT
ORNITHOLOGY
ECOLOGY OF SIBERIAN CRANE
CLICK HERE TO RETURN TO KEOLADEO PARK HOME PAGE
CLICK HERE TO GO TO SPECIES

[illegible]

Figure 7 12 A screen showing information about Siberian Crane

Chapter 8

Image Database System Architecture

8.1 Introduction

The advances in multimedia technology (as reported in *Chapter 7*) basically centers around new ways to store, retrieve as well as transmit digital information. Consequently, image and video databases have become the central component of many future applications. An appropriate image database system (IDS) architecture would aid in providing convenient and efficient access to the data contained in an image database.

Information relevant to an image application, may need to be stored in multiple formats. Answering a user-query typically requires handling of such information in multiple forms and representations. Correlating these information at the physical level by pre-analysis is not an attractive option. For example, there could be thousands of images, and each image can have several 'objects of interest', thus during query processing, retrievals and matching of all these objects at the physical level would be very time consuming and un-rewarding process. Therefore, it is believed that to support the quicker query processing activities, it is quite necessary to represent as well as process the digital data in a *semantic level* (referred here as *metadata*). However, in general humans are more capable of abstracting information efficiently from images, displayed on the computer. This enables them to correlate and match information of a queried object at a higher *semantic level* with the stored objects' representations such as the symbolic representation of data in structured databases. This semantical correlation and matching of information in an efficient manner, across various representations is lacking in the current IDS architectures and has been characterized as a "semantic bottleneck" [Subra96]. To overcome this limitation, this chapter introduces a three-layered metadata-based IDS architecture, which rather than processing the complex queries at the 'physical level', prefers to handle it at the 'logical' or *meta-level* for more economic and quicker responses. For the sake of validity, the various implementation issues of the proposed architecture in the context of a museum-cum-archival application has also been reported in this chapter. The next Section reports a brief review of the existing image retrieval systems and metadata based architectures.

8.1.1 Reviewing the Image Database Systems

There is a multitude of application areas that consider image retrieval as a principal activity [Gudv96]. Tamura and Yokoya provide a survey of image database systems that were in practice around the early 1980s [Tamur84]. Chock also provides a survey and comparison of functionality of several image database systems for geographic applications [Gudv96]. Recently, Grosky and Mehrotra [Grosk88, Grosk92] and Chang and Hsu [Chang92] discuss the recent advances, perspectives and future research directions in image database systems. More recently, [Gudv95] provides a comprehensive survey and relative assessment of Picture Retrieval Systems. The survey reveals that to support quicker query processing activities, it is essential to represent as well as process the digital data in a *metadata-level*, which would enable the users for better correlation and matching of information of a queried object. However, from the perspective of query solving, in a metadata based architecture, the metadata should be capable enough for representing the *contents* of the image data in an appropriate manner. Semantics of an image and its objects include both the "meaning" and "use" of an object [Kashp96]. The metadata level represents the level at which the information from the various media formats shall be viewed and compared. Some of the significant recent work in developing metadata for digital media can be found in [Subra96, Klaus94]. Bohm and Rakow have provided a classification of metadata in the context of multimedia documents. Jain and Hampapur have characterized video metadata and its usage for content-based processing. Kiyoki et al. have used metadata to provide associate-search of images for a set of user-given keywords. Anderson and Stonebraker have developed a metadata scheme for management of satellite images. Grosky et al. have discussed a data model for modelling and indexing metadata and to provide the definition of higher abstraction. However, the level of abstraction at which the content of the images is captured and represented, is very important.

From the survey, it appears that in the context of the diverse application areas, the characteristics of most of the existing image database systems have essentially been evolving from domain specific considerations, and hence, a very little consensus exists among these systems. It is strongly felt that there is a lack of a standard IDS architecture, which would be 'easy to use' and would cover a wide variety of applications. The next Section describes the design of the proposed metadata based three-layered architecture.

8.2 Image Database System (IDS) Architecture

Th ost important characteristics of an Image Database System are :

- it supports image data which are conceptually semi-structured in nature. The semantics of an image entity is a function of the semantics of its components;
- it supports multiple user views, and
- each image entity possesses basically two types of attributes and relationships : *content-based* and *content-independent*, where the content-based attributes and relationships are required to undergo appropriate decoding procedures, whereas the others do not

To enable the users for better handling of the complex, storage-intensive image data pertaining to an IDS, the designer is to emphasize two aspects : the extraction of *content-based* and *text-based* features for each image and its logical components; and secondly the generation of metadata from the extracted 'raw' features. For the purpose of feature-extraction and for better representation of the visual semantics in a conceptual manner, it introduces a data model, referred here as *Visual Semantic Model (VSM)*. Next subsection discusses the importance of image data model in the context of complex query processing and it also presents the proposed VSM.

8.2.1 Image Data Modelling

The objective of an image data model is to represent the semantics of a scene such as the image *entities*, their *attributes*, *associations* among the entities and the logical *organizations* of each entity, in order to determine the view(s) of the content of an image. Thus, an image data model should be capable enough for expressing these semantics at desired level(s) of abstractions in a conceptual manner. However, due to the diverse nature of image database applications, it is intrinsically difficult for the designers to conceive a general image data model. A standard formalization of an image data model, which can serve as a platform on which other aspects of image database systems, such as query processing, retrieval etc. can be realized, has been a major 'bottleneck' for long time. As found in [Gudv96], in an advanced image database systems, one can find four categories of data : formatted, structured, complex and unformatted. *Formatted* data are most commonly found in traditional databases. *Structu*[illegible] data are heterogenous types of data about an object that need to be stored and [illegible]ieved together. When the structured data possess variable number of components, they are referred to as *complex* data. *Unformatted* data are usually the 'string data' (also referred to as byte string, long field, Binary Long Object Box (BLOB)), whose structure is not understood by the database management system. To understand the semantics of the unformatted data types, special *procedures/methods* are essential. In this background, several data models can be found in the traditional DBMSs, e.g. relational, hierarchical, network, etc, where the data to be managed are of *formatted* type. However, though these traditional DBMSs based on the relational data model, have also been used for image database management, they are not "true" IDBMS (Image DBMS) [Gudv96]. For, the level of abstraction offered by these data models for representing the images is too high; and the data model, the query specification language and the retrieval strategy are essentially those of the traditional DBMSs. However, to overcome the limitations faced with the traditional DBMS, there has been a great interest in providing several extensions to the relational data models to overcome the limitations imposed by the flat tabular structure of relations for geometric modelling and engineering applications [Gudv96]. The resulting data model is characterized by the addition of application-specific components to an existing database system kernel. They include nested relations, procedural fields, a query-language extension for transitive closures, among others. The primary objective of all these extensions is to overcome the fragmented representation of the geometric and complex objects in the relational data model. Image data is stored in the system as formatted data. However, to a database user this view of data is made transparent through these extensions.

In this background, several other data models have also been proposed in [Chen76, Bomel91, Hofst93, Grifn93]. Some systems perform *'spatial reasoning'* as a part of the query processing while other systems have attempted cognitive approaches to query specifications and processing [Chang88, Chang91, Sisla94, Sisla95, Tanaka88]. However, in the context of complex content-based image information, the existing image data models still have been found to be inappropriate and the level of abstraction at which the image information is represented is too low. Next subsection presents a suitable, semantically rich image data model, based on the concept available in [Gudv96].

8.2.2 The Visual Semantic Data Model : VSM

The proposed VSM provides scopes to represent the semantical information of any image scene at a desired level of abstraction as per the application requirements. In *Figure 8.1*, an *image-level* representation of a scene based on the VSM notations and conventions, has been shown. The rectangle shape symbolizes the abstract *image class*, the double bordered rectangle represents the abstract classes of *image entities* identified from the image. The oval shapes represent the *objects of interest* associated with the image as well as its entities, such as the features or attributes, positional constraints, logical compositions, semantical associations etc. It supports almost all types of associations often necessary in a complex scene representation. A solid line with *Double headed* arrow represents the 'multi-valued' attributes, a *single-headed* arrow represents 'single-valued' attribute, a *dotted directed line* represents the 'part-of' relationship and a line with *thick headed* arrow represents the 'depends on' relationship etc. As the proposed VSM helps in modelling an image at various levels of abstractions, to provide supports in modelling the entity-level relationships, it includes an additional class of notations and symbols. An abstract class of interface is provided to facilitate the model-information-acquisition process. Interfaces are designed based on the concept available in [Gudv96, Jack83, Flick95]; the underlying *'Object-based'* concept in the interface helps in identifying the 'image-objects' (i.e. the entities) and their relationships; the *'query-by-example'* environment assists the user in providing relevant features for each entity and a *'background/foreground'* model-based interface provides the necessary tools to derive the shape-features for the image and its entities. The various components and relevant 'objects of interest' of the proposed VSM are described below.

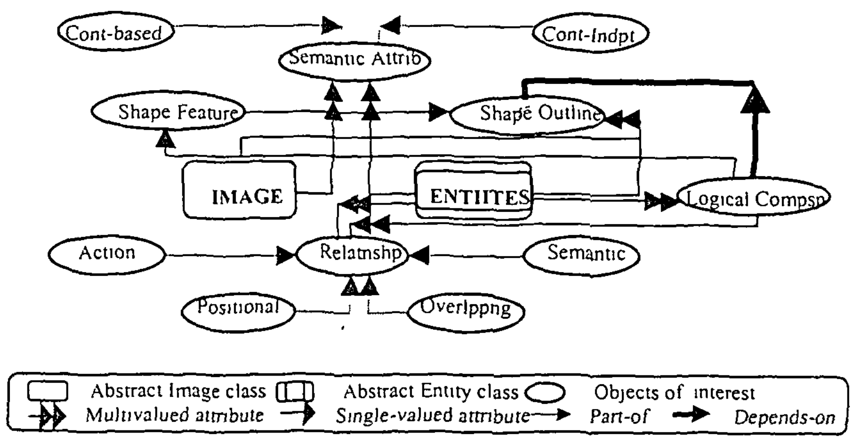


Figure 8.1: Modelling an Image using VSM

Image and Its Entities : The proposed VSM provides a distinct set of modelling constructs to represent the image, its entities and their relationships. Identification of the entities are made through a semi-automatic process at the time of image insertion as well as query solving. The interface is designed based on a object-based approach [Jack83] which accepts a set of domain-specific *key-sentences* from the user and based on a 'filtration' and 'identification' procedure, it separates out the entities. The existence of entities in an image scene, entirely depends on the angle of interpretation used by the designers. As for the same content of an image, there can be multiple interpretations. So, the entities located by a user for a specific scene may not be expected for another user.

Shape outline and Features for Image and its Entities : *The shape outline* of an image basically provides the boundary sketch of the raw image, and the entity shape provides the outline for each individual entity. To derive the shape outline of the image and its entities, a model-based [Flick95] interface with a set of tools, is provided. Finding the outline of an image is rather easier and simpler than finding its entities. However, many efficient shape-detection tools have evolved during the current decade [Gudv95] in order to support this activities. The interface also provides a database of 'textures' and 'color-palates' to provide necessary supports in the appropriate identification of the shape features associated with the derived 'boundary-sketch'.

Content-based and Content-Independent Features : Due to the richness of the information content in the images, there can be many interpretations for the same image, and the interpretations depend entirely upon the requirements from information retrieval point of view. The interpretation of the content of a 'painting' made by an artist may be different from its interpretation made by common people. Both image and its entities may have some attributes derived externally or based on the content of the image or image-entities. The attributes or features based on the content are known as 'content-based' attributes and the others (i.e. derived externally) are referred to as 'content- independent' attributes. For example, the shape-features of an image-object, stroke-patterns in a painting, etc are content-based, whereas, the date_of_acquisition, image_index, etc are content-independent features.

Entity-Relationships : The proposed VSM supports various semantical relationships among the entities of the image, such as *spatial, actions, overlapping* and other conventional *semantic associations.* Among the spatial or positional relationships, specifically the relationships, such as *left_of, right_of, in_front_of, behind, above* and *below* are used. The overlapping relationships include *left_overlaps, right_overlaps, bottom_overlaps, top_overlaps* etc. Relationships which describe the various *actions* (or, *role*) played by an entity are belonging to action-relationships group, e.g. *shaking_hand, moving_to, smiling_at, holding* etc. A detailed discussion on these three types of relationships can be found in [Sisla94, Sisla95]. Other semantic relationships includes the conventional associations among the entities, such as *consists_of, inherits, abstraction_of* etc. These relationship information are extracted a priori (by a combination of both algorithmically as well as by manually).

Thus, the construction of VSM can be summarized as a sequence of three basic transformations, as noted below :

1 : the transformation in which the image-level content-based and text-based attributes (e.g. type, class, shape etc.) are derived.

2 : the transformation which deals with the relationships among the objects of interests from compositional, spatial as well as semantical associations point of view.

3 : the transformation, where the entity-level content-based features (e.g. shape, textures etc) are extracted and derived.

The model information are used as input to a *conversion mechanism* to generate the corresponding metadata for each image-object, and the proposed architecture (will be discussed next) utilizes the metadata in answering the various queries.

8.2.3 The Proposed Architecture

The proposed three-layered, metadata based architecture supports all the common characteristics of an IDS. It is designed based upon the concepts available in [Gudv96, Kashp96, Bhatt96b]. The various layers of the architectures are :

- *Database layer*, to store the physical images as well as the associated logical data in an easily manageable form. To manage properly and to provide quicker database responses, the database layer is further sub-divided into two sublayers namely, *logical* or *meta-sublayer*, and *physical* or *digital-sublayer*;
- *Systems* or *processing layers*, to provide necessary supports for better utilization of the precious image-data stored in the physical layer. It consists of the various application modules, where each module is designed, in order to achieve some pre-specified objectives;
- *Application* or *interface* layer, to provide better and realistic presentation support against the IDS requests, made by the 'end-user'. It is basically a collection of several abstract interface classes, each of which corresponds to an application processing module.

The main objective of this architecture is to provide a 'means' for easy mapping the real-world problem domain into an image database application domain. The various layers of the proposed architecture are discussed next.

Database Layer

The physical data in an image database may need to be stored in multiple formats as per the applications requirements. During query processing, it is often essential to manipulate these digital data across the different storage formats, either at the 'image-level' or at the 'component-object level'. However, for easy manipulation, it is essential to treat them at the 'semantic' or 'meta-level' [Gudv96]. During new object capturing or insertion process, or during query processing, relevant features of each object are extracted, filtered and then utilized in the construction of a corresponding VSM. Finally, the model-information is transformed into the respective 'metadata', based on a *conversion* procedure (analogous to [Rishe93]). These metadata related to each of the physically stored image-objects are collectively stored as a 'meta-sublayer' (as shown in *Figure 8.2*). The two sub-layers of the database layer are discussed next.

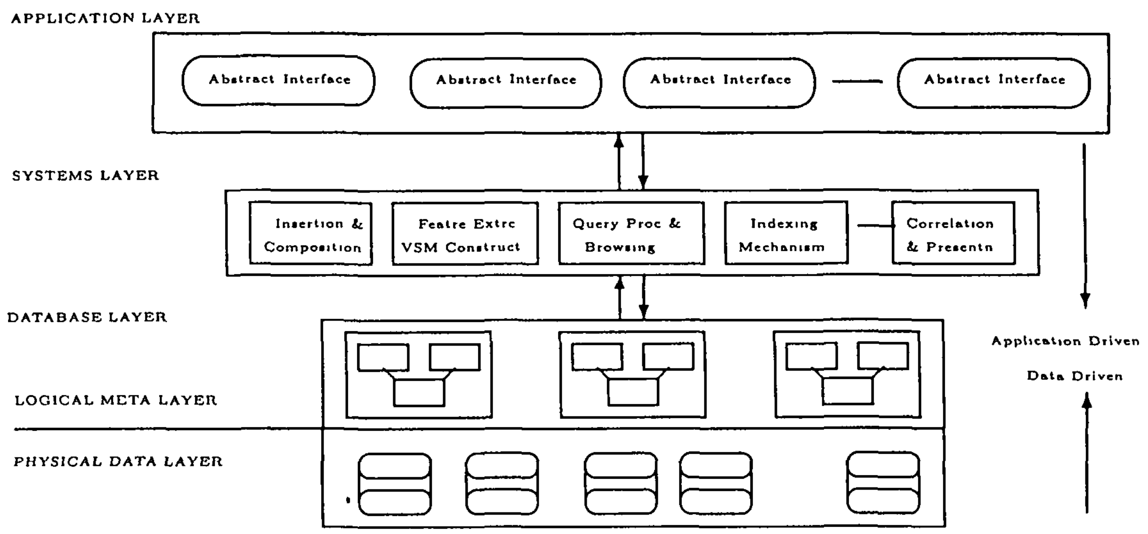


Figure 8 2: The proposed three layered IDS architecture

Physical Sub-layer

This sub-layer contains the actual (raw) data which might be stored in any of the image data formats Images may be of different modalities like Paintings, Legal Photographs, X-Ray, MRI scan etc.

Logical or Meta Sub-layer

This sub-layer holds information referred here as 'metadata', which can be described as the summary of the information content about the 'images' of the physical data layer in an intentional manner. These metadata are the representation of both the *content-based* and *content-independent* features of the physical images and their component objects. The *content-based* metadata includes the attribute information about each of the 'objects of interest' (e.g. the physical characteristics of the object, etc.), as well as the information about the relationships which exist among the objects or entities. *Relationships* are categorized, based on their nature of occurrences, into three: *spatial, actions, overlapping* and other conventional *semantical associations*. Besides the content-based, the proposed architecture also incorporates metadata representing the text-based content descriptions (such as location, date, time_of_creation, etc.) for each of the objects. For the purpose of extraction of these features, a class of user-friendly interfaces are used, and these raw features are finally 'filtered' by using a *conversion procedure* (which assigns appropriate 'weigtages' (in 10-point scale) to each feature) the corresponding metadata (i.e. a k- dimensional feature tuple, where k is the number of features) are generated. These metadata are associated with the digital media through the 'internal disk-addresses' maintained for each physically stored object.

To enhance the performance of an IDS, implementation of an appropriate indexing mechanism is very essential. The current literature [Subra96, Gudv95] reveals that for optimal utilization of the content-based information in retrievals, a content-based indexing mechanism is always preferable. The proposed architecture also uses an indexing mechanism based on the *nearest-neighbour-search* concept, as available in [Chiuh94]. Before serving any query requests, the k-dimensional *feature vectors* i.e the metadata collections are indexed using a $k - d$ *search tree* method based on an n-ary *partitional approach*. If the ranked list of images located from the indexed database, is still found to be not browsable, then it keeps provision for a *finer-entity level* sequential searching mechanism to identify the most minimum set of 'desired' images.

Systems or Processing Layer

This layer basically comprises a set of processing modules, designed based on object-oriented concept. Each module performs some pre-specified activities or operations defined for a concerned application domain, such as : insertion & composition, feature extraction & representation, metadata generation, query processing & browsing support, presentation, etc. Each of the modules is briefly described below.

Image Insertion & Composition Module

This module helps insert a new 'object' into the database as well as compose a new object from the existing objects, based on some pre-defined *constructor operators*. Since, the image database applications are generally insertion-and-query-intensive in nature, this module has an important role in the creation of image database. It supports a strong set of tools and techniques (as found in [Flick95]) for capturing and composing new objects to the database.

Feature Extraction/Representation and Metadata Generation Module

As the extraction & representation of 'features' and subsequently the 'metadata' generation are inter-related processes in an IDS application, they may be discussed together in the same sub-section In this architecture, metadata are at the most critical level and as it is generated entirely depending on the feature-extraction and representation processes, this module has a significant role. The extraction / representation process occurs basically in three phases as shown in *Figure 8.3.* Each of these phases is discussed below:

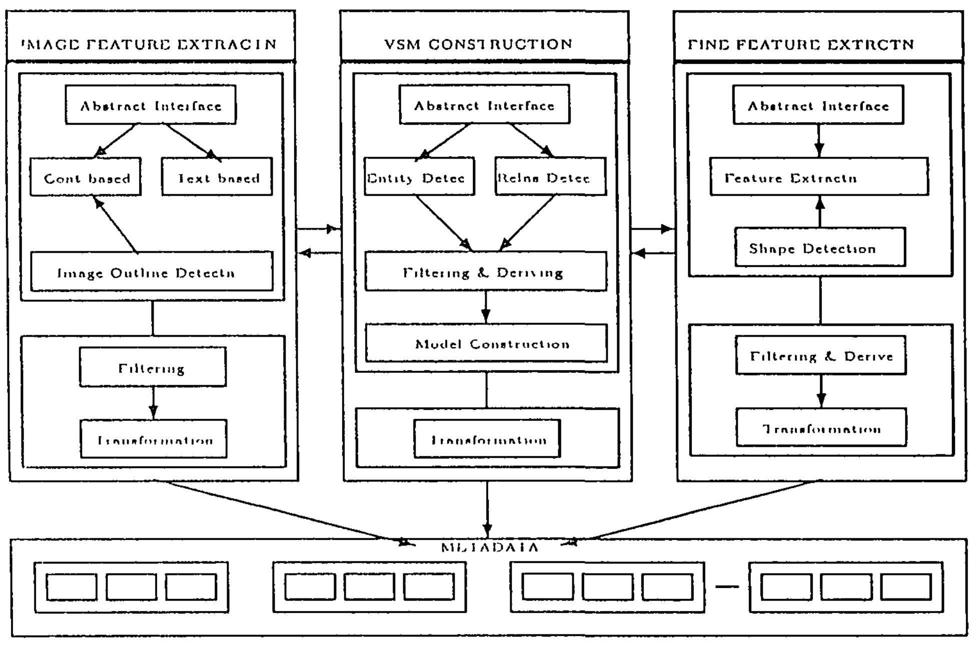


Figure 8.3. Feature extraction and metadata generation

[A] Image Feature Extraction Phase : This phase basically deals with the acquisition of various content-based and content-independent attributes and relationships for each of the images and its objects-of-interests. During new object insertion or at the time of query solving, this phase invokes a class of user-friendly interfaces, designed based on *query-by-example* principle, to acquire the various image-level and entity-level attributes (e.g. the physical characteristics, shape features, content-independent description information, etc) their compositional, spatial as well as other semantic relationships information etc. To provide necessary supports at the various levels of extractions, this module also includes a strong set of tools for the purpose of feature selection (such as texture database, color wheel, etc.), shape drawing etc.

[B] **VSM Construction Phase :** The proposed extraction/representation module also supports a graphical tool for presentation of the captured object-relationship information in an understandable form, based on the notations defined with the proposed VSM. The model is constructed in a semi-automatic manner. The main objectives of this model is : (a) to understand and represent the complex semantics of an image scene, and (b) to assist in solving complex queries. *Figure 8.1* shows representation of a typical image scene based on the proposed VSM notations.

[C] **Fine Feature Extraction Phase :** This phase deals with some special tools and techniques (for example, an *object-outliner* with transformation utility, *colour composition detector*, etc.) to extract the finer features of each entity, such as *entity-shape*, *texture-variations*, *intensity-distribution*, etc. A class of interfaces are designed to assist in this extraction process. The feature extraction tools are utilized in several passes for finer property detection and subsequently a set of 'filters' are used to eliminate the redundant information. Due to the complex nature of entity-types, it has been found that in some situation, the appropriate extraction becomes possible, only when there are manual interventions. After extraction, the features information are transformed in order to generate the corresponding metadata.

Once the model information comprising both the content-based and text-based feature information for each image as well as its 'objects of interest' are transformed (based on a *conversion mechanism* analogous to [Rishe93]), the next step is to construct a k-dimensional feature vector, to represent each image as a point in a k-dimensional space. Such a k-dimensional vectors or *feature-points* for each image, together constitute the *logical meta layer* for the physically stored *image data layer*. The *feature-vectors* are exploited in an indexing mechanism to facilitate especially, in the content-based searching mechanism. The indexing mechanism used in the proposed architecture is a variant of

[Chiuh94]

Query Processing and Browsing Support

The activities of the query processing and browsing support modules are inter-related to some extent. In case of both the modules, accession of image database, its retrieval as well as filtering techniques would be same. So, both these modules could be discussed together. The query processing in an IDS takes place in a similar way as in other database systems. Due to the possibilities of multiple interpretations for the content of the same image for different users, the logic in querying an image database would not be straight-forward and simple. Thus, it is essential to be careful enough, in finding an appropriate query-solving approach. The proposed architecture employs a query-solving method based on a unified approach. It attempts to utilize both the content-based and content-independent information for the input 'candidate object' at various levels, and based on these information, similarities with the stored features of 'prototype objects' are computed using a set of pre-defined *rules* [Sisla94, Sisla95, Gudv95]. For faster query responses through optimum utilization of the content-based features, a $k-d$ search tree based indexing mechanism is used, analogous to [Chiuh94].

In *Figure 8.4*, the schematic of the query-processing mechanism has been shown, which includes a set of processing modules. Initially, for the query-object, through an interactive session, all possible image-level features along with the relevant entity-relationship information are extracted as well as derived. The redundant information are eliminated based on a set of *semantic filters*. With the 'refined' entity-relationship features, using the defined VSM notations, a corresponding data model for the query-image scene is constructed. To support the construction as well as for better presentation, a graphical tool is designed, which works semi-automatically. Later, the text-based and content-based model-information are input to a *conversion* procedure to assign appropriate weightage for each of the features to ultimately generate the corresponding metadata, i.e. the k-dimensional feature vector.

An entity-level finer-feature based sequential matching mechanism has also been defined (optionally) with this query processing and browsing mechanism, which may be invoked, if the users are not satisfied with the size of the ranked list derived due to the above indexing mechanism. However, the complexity in the query processing logic depends upon the complexity of the features to be handled for the image entities.

Application or Interface Layer

The user interface layer is designed to let users easily select content-based and text-based features, allow these features to be combined with each other, and let users reformulate queries and generally navigate the database. For better presentation of the queried results, it also helps by providing an interactive presentation tools.

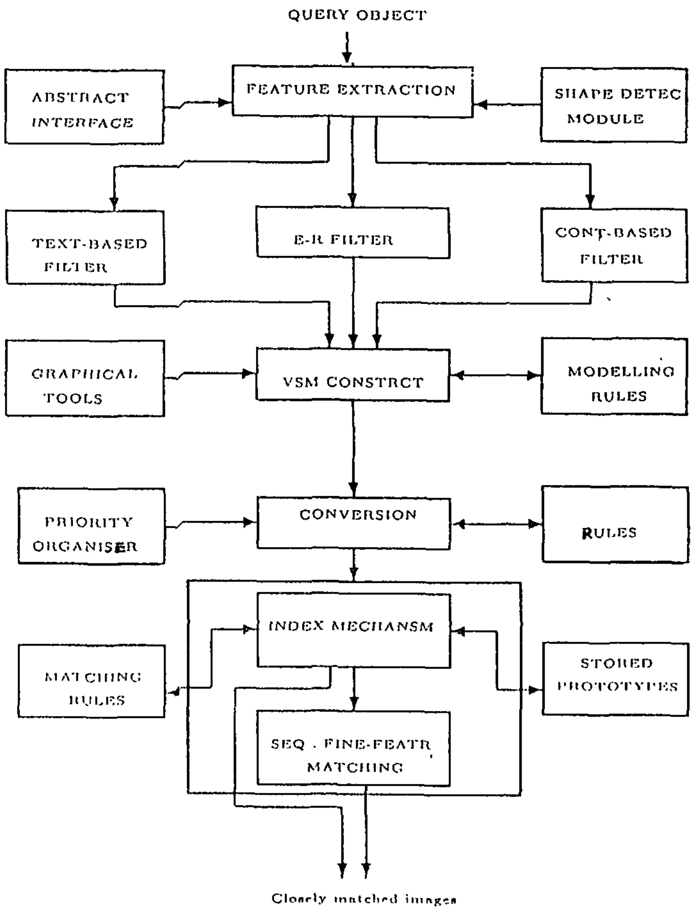


Figure 8.4: Query processing mechanism

8.3 Implementation Issues

The proposed architecture was tested in the context of a real-life archival and museum application. The application is a database for managing the various archives and museum items found in North-East of India, collected and preserved by the Traditional

Culture and Art Forms (TCAF) Department of Tezpur University, India The intended users of this system are the students, faculty members and other visitors to the TCAF Department The collection contains around 2000 full-colour photographs of the various traditional dances, beauty and costume items, around 650 black & white photographs of the primitive cultural items, 400 colourful paintings (acquired in the form of coloured photographs) and photographs of some renowned personalities related the culture and art form of this part of country A prototype system called "Archival and Museum Database System (ARMUDS)" has been developed using *Visual Studio* utilities in a Pentium-Pro platform The system has generated some of the test results, which are found to be satisfactory The software is still under refinement

8.4 Conclusion

A new, three-layered, metadata-based Image Database System architecture, designed based on modularity, code-reusability and abstraction mechanisms has been introduced in this chapter The architecture handles the physically stored digital data, at meta-level to provide easy manipulation and quicker query responses For better understanding of the complex queries as well as to represent the image semantics at a desired level of abstraction, it also proposes a visual semantic data model. To justify the usefulness of the architecture, a prototype software has also been developed and the responses are found to be satisfactory

Next chapter summarizes the contributions of the work and indicates the future extensions of the work

Chapter 9

Conclusion and Future Work

This chapter summarizes the main contributions of the work and provides directions for future extensions Some studies on characterization on additive CA based on matrix algebra have been reported in this research work. We have extended the characterization of ENCAs to some extent. Some of the significant results of ENCA can be summarized as follows :

1. there exists a class of group ENCAs (*maximum-length* and *non-maximum-length*) which validates the properties of 3-neighbourhood additive group CAs reported in [Nandi94a].

2. there exists a class of non-group ENCAs, which validates all the properties of TPMA CAs, as reported in [Nandi94a].

Cryptography has become an essential requirement for protecting private information against unauthorized access. It is the only practical means for sending information over an insecure channel. The increasing use of electronic media for data communication, coupled with the phenomenal growth of computer usage, has significantly raised the necessity of constructing suitable schemes for authentication, data security and authenticated key exchanges In this context, using CA as a basic building block, new schemes for message authentication and data security have been established. Based on these proposed CA based schemes, a new password-only authenticated key exchange scheme has also been established for user authentication.

Reliable communication and reproduction of data is desirable in a distributed as well as storage systems With the emergence of large-scale, high-speed data networks for exchange, processing and storage of digital information in the military, governmental and private spheres, the demand for such reliable systems has been increasing. In this background, we have proposed a t-EC/d-ED ($t < d$ and AUED codes to control errors, so that reliable reproduction of data can be obtained. Based on even-parity check logic using odd-weight column matrix, a SEC-DED-AUED symmetric code has also been proposed. It has been established that both the schemes show significant results by reducing the ROM word-length.

With the proliferation of the various heterogenous form of media data (such as image, video, audio, graphics, text, etc.) as well as the proliferation of a number of commercially available tools to manipulate these data, an explosion of interest has been noticed in hypermedia application developments. However, a standard, easy to use data model for conceptual modelling of any scale of hypermedia application domain is still lacking. We have studied, analysed and extended a data model, referred as RMDM to enable the modellers to represent the semantics of any complex hypermedia domain naturally.

With the enormous growth of information technology and the presentation technique, in particular, there has been a great demand for high quality image processing and faster retrievals- that require a suitable image database systems (IDS) architecture. In this thesis, a new architecture for image database processing and retrieval has been proposed, which utilizes both the content-based and text-based features for the physically stored digital images. For the better understanding of the complex queries, as well as to represent the content-enriched semantics of the image scenes at a desired level of abstraction, a visual semantic data model has been proposed too. The implementation aspects of the proposed architecture in the context of a real-life case has also been described.

9.1 Future Works

In the following, some of the possible directions of future works in the field of Cellular Automata, data security, authentication, error control coding and image database systems are outlined :

- The characterization of group and non-group ENCAs (Extended Neighbourhood CA) has been reported in this thesis. Such study can be extended for complete and more generalized characterization of ENCAs.

- The one-way hash-function and enciphering scheme reported in this thesis can be extended to hardwired implementation of the schemes.
- The non-group, non-linear MACA properties can be further studied for its utilization in *Stegenography*.
- The ENCA based block cipher system reported in this thesis can be further extended to the design of a compact hardware-based unified scheme, which operates both in block and stream cipher mode.
- the work on t-EC/d-ED/AUED code construction can be further enhanced to reduce the overall code-length.
- The work on Image Database System reported in this thesis can be further enhanced by incorporating an improved version of content-based indexing mechanism.

Bibliography

[Alady76] V. Aladyev, "The behavioral properties of Homogeneous Structures," *Mathematical Biosciences*, vol. 29, pp. 99–134, 1976.

[Alexn93] N. Alexandris, M. Burmester, V. Chrissikopoulos, and Y. Desmedt, "A Secure Key Distribution System," in *3rd Int'nl Conf Proc. on Cryptology*, pp. 30–34, 1993.

[Amoro72] S. Amoroso and S. K. Yap, "Decision procedures for surjectivity and injectivity of parallel maps for tesellation structures," *J. Comput. System Sc.*, vol. 6, pp. 448–464, 1972.

[Apsim70] H. G. ApSimon, "Periodic forests whose largest clearings are of size 3," *Philos. Trans. R. Soc. London*, vol. A 319, p. 113, 1970.

[Balas94] P. Balasubramanian, T. Isakowitz, and E. A. Stohr, "Designing Hypermedia Application," in *Proc. of HICSS*, pp. 354–365, IEEE CS, 1994.

[Baldn97] R. W. Baldwin and C. V. Chang, *Locking the E-Safe.* IEEE Spectrum, pp. 37-41, 1997.

[Balas95] V. Balasubramanian, B. M. Ma, and J. Yoo, "A Systemtic Approach to Designing a WWW Application," *Comm. of the ACM*, vol. 38, pp. 47–48, August 1995.

[Barde90] P. H. Bardell, "Analysis of Cellular Automata used as pseudo-random pattern generators," in *International Test Conference '90*, pp. 762–768, 1990.

[Basam94] S. Al-Bassam and B. Bose, "A Symmetric/Unidirectional Error Correcting and Detecting codes," vol. C-43, pp. 590–597, May 1994.

[Bclvn92] S. M. Bellovin and M Merrit, "Encrypted Key Exchange: Password based protocols secure against Dictionary Attack," in *IEEE Symposium on Research in Security andPrivacy*,pp. 72–84, IEEE CS, 1992.

[Bclvn93] S M. Bellovin, "Augmented Encrypted Key Exchange: a password based protocols secure against dictionary attack and password file compromise," in *ACM Conf Proc on Computer and Communication Security*, pp. 244–250, ACM Press, 1993.

[Biglw88] J. Bigelow, "Hypertext and CASE," *IEEE Software*, vol. 5, March 1988.

[Bhatt96a] D. K. Bhattacharyya and B Dubey, "Multimedia in Ecosystem Modelling : An Application to Keoladeo National Park," in *Proc of SIP'98, IASTED*, Annecey, France, 1996.

[Bhatt96b] D K Bhattacharyya, M. C. Bora, and S. Nandi, "An Architecture of Multimedia Information System," in *Trends in ADvanced COMPuting*, pp. 178–181, Tata McGraw Hill, 1996.

[Bhatt97a] D. K. Bhattacharyya, S. N. Pandit, and S. Nandi, "A New Enciphering Scheme," in *Proc on Int'nl Conf on Telematics*, pp. 1–11, NERIST, Nirijuli, India, 1997.

[Bhatt97b] D K Bhattacharyya, S Nandi and M. C Bora, "Efficient Design of One-way Hash Function using Cellular Automata," in *Current Trends in ADvanced COMPuting*, Tata Mc GrawHill, 1997.

[Bhatt97c] D. K Bhattacharyya, S. Nandi and M C. Bora, "Application of RMDM and its Drawbacks," in *Proc. of CUPUM'97*, pp. 866-877, Narosa, 1997.

[Bhatt97d] D. K. Bhattacharyya and S. Nandi, "A New Class of t-EC/d-ED ($d > t$)/AUED Codes," in *Proc of PRFTS'97*, pp. 41-46, IEEE CS, 1997.

[Bhatt97e] D K. Bhattacharyya and S Nandi, "A Efficient Class of SEC-DED-AUED Codes," in *Proc. of I-SPAN'97*, pp 410-416, IEEE CS, 1997.

[Bhatt97f] D K Bhattacharyya and S N Pandit, 'Basin Structure with Ultimate Periodicity," in *Proc of Internet for India*, pp 95-100, Hyderabad, India, 1997.

[Bhatt98a] D. K Bhattacharyya and S Nandi, "Theory and Design of SEC-DED-AUED Codes," in *IEE(E) Proc*, vol 145-2, pp. 1-6, March, 1998.

[Bhatt98b] D. K. Bhattacharyya and S. Nandi, "A Secure Block Ciphering Scheme," in *Proc of ISIT'98*, IEEE CS, August, 1998.

[Blaum89] M. Blaum, Tilborg, and H. van, "On t-error Correcting/all Unidirectional Error Detecting codes," *IEEE Trans. on Computers*, vol. C-38, pp. 1493–1501, November 1989.

[Bomel91] P. V. Bommel, A. H. M. Hofstede, and T. P. Weide, "Semantics and Verifications in Object-Role Model," *Information System*, vol. 16, pp. 471–495, 1991.

[Bose82] B. Bose and T. R. N. Rao, "Theory of Unidirectional error correcting/detecting codes," *IEEE Trans. on Computer*, vol. C-31, pp. 521–530, June 1982.

[Bose85] B. Bose and D. J. Lin, "Systematic Unidirectional error-detecting codes," *IEEE Trans. on Computer*, vol. C-34, pp. 1024–1032, November 1985.

[Boinc90] F. J. H. Boinck and H. C. A. V. Tilborg, "Construction and Bounds for Systematic t-EC/AUED Codes," *IEEE Trans. on Infor. Theory*, vol. 36, pp. 1381–1390, November 1990.

[Bruck92] Bruck and Blaum, "New techniques for constructing EC/AUED codes," *IEEE Trans on Computer*, vol. 41, pp. 1318–1324, November 1992.

[Burks70] A. W. Burks, "Essays on Cellular Automata," Tech. rep., Univ. of Illinois, Urbana, 1970.

[Burms94] M. Burmester, "On the risk of opening distributed keys," in *Advances in Cryptology: CRYPTO'94, LNCS 839*, pp. 308–317, Springer-Verlag, 1994.

[Chang91] C. Chang and S. Lee, "Retrieval of Similarity Pictures on Pictorial Databases," *Pattern Recognition*, vol. 24, no. 7, pp. 675–680, 1991.

[Chang92] S. K Chang and A. Hsu, "Image Information Systems: where do we go from here?," *IEEE Transac. on Knowledge and Data Engg.*, vol. 4, no. 5, pp. 431–442, 1992.

[Chang88] S K. Chang and A. Hsu, "An intelligent Image Database Systems," *IEEE Transac on Software Engg* , vol 14, pp. 681–688, 1988.

[Chen76] P. P Chen, "The Entity-Relationship model- towards a unified view of data," *ACM Transaction Database System*, vol. 1, pp. 1–26, 1976.

[Chiuh94] Z. Chiueh, "Content Based Indexing," in *VLDB'94 Conference Proceedings*, September 1994.

[Chowd92] D. R. Chowdhury, *Theory and Applications of Additive Cellular Automata for Reliable and Testable VLSI Circuit Design.* PhD thesis, I.I.T. Kharagpur, India, 1992.

[Chu94] W. W. Chu, I. T. Leong, and R. K. Taira, "A Semantic Modelling approach for Image Retrieval by Content," *VLDB Journal*, vol. 3, 1994.

[Cmpbl88] M. Campbell and J. Goodman, "HAM: A General Purpose Hypertext Abstract Machine," *Communication of the ACM*, vol. 31, pp. 856–861, July 1988.

[Codd68] E. F. Codd, *Cellular Automata.* Academic Press Inc., 1968.

[Daemn93a] J. Daeman, R. Govaerts, and J. Vandewalle, "A Framework for Design of Hash Function," in *Advances in Cryptology: ASIACRYPT'92*, pp. 82–86, Springer-Verlag, 1993.

[Daemn93b] J. Daeman, R. Govaerts, and J. Vandewalle, "Fast Hashing both in Hardware and Software," in *Advances in Cryptology: CRYPTO'93*, Springer-Verlag, 1993.

[Das90a] A. K. Das, *Additive Cellular Automata : Theory and Application as a Built-in Self-test Structure.* PhD thesis, I.I.T. Kharagpur, India, 1990.

[Das90b] A. K. Das and P. P. Chaudhuri, "Efficient characterization of Cellular Automata," *Proc. IEE (Part E)*, vol. 137, pp. 81–87, January 1990.

[Das91] A. K. Das, A. Sanyal, and P. P. Chaudhuri, "On the characterization of Cellular Automata," *Information Science*, 1991.

[Das93] A K. Das and P. P.Chaudhuri, "Vector space theoretic analysis of additive cellular automata and its applications for Pseudo-exhaustive Test Pattern Generation," *IEEE Trans. on Computers*, vol. 42, pp. 340–352, March 1993.

[Davis83] D W. Davies and D. Clayden, *Security for Computer Network.* John Wiley and Sons, New York, 2nd edition, 1983.

[DES67] FIPS PUB. 46, "Data Encryption Standard," January 1967.

[Denni82] D. E. Denning, *Cryptography and Data Security*. Addison-Wesley Publishing Company, Reading, Mass, 1982.

[Deslc86] N. Desisle and M. Schwartz, "Neptune : A Hypertext System for CAD Application," in *ACM SIGMOD'86, Washington D.C.*, pp. 132–142, ACM Press, 1986.

[Desmd93] Y. Desmedt and M. Bermester, "Towards practical 'proven secure' authenticated key distribution," in *ACM Conf on Computer and Communication Security*, pp. 228–231, ACM Press, 1993.

[Diffi76a] W. Diffie and M. E. Hellman, "Multiuser Cryptographic Technique," in *Proceedings of AFIPS*, pp. 109–112, National Comp. Conference, 1976.

[Diffi76b] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Transac. Info. Theory*, vol. 22, pp. 644–654, 1976.

[Diffi79] W. Diffie and M. E. Hellman, "Privacy and Authentication : An Introduction to Cryptography," vol. 67, no. 3, pp. 397–427, 1979.

[Diffi92] W. Diffie, P. C. van Oorschot, and M. Wiener, "Authentication and Authenticated Key Exchange," in *Designs Codes and Cryptography*, vol. 2, pp. 107–125, 1992.

[Dmgrd90] I B. Damgard, "A Design Principle for Hash Function," in *Advances in Cryptology: CRYPTO'89*, pp. 416–427, Springer-Verlag, 1990.

[Dobbi96] H. Dobbertin, A. Bosselaers, and B. Praneel, "RIPEMD-160: a strengthened version of RIPEMD," in *Fast Software Encryption: 3rd Int'nl Workshop (LNCS 1039)*, pp. 71–82, Springer-Verlag, 1996.

[Ellis96] C. Ellison, "Establishing Identity without Certification Authority," in *Proc. of Sixth Annual USENIX Security Sympo.*, pp. 67–76, July 1996.

[Feist73] H Feistel, "Cryptography and Computer Privacy," *Sci. Am.*, vol. 228, pp. 15–23, May 1973.

[Flick95] M. Flickner, H. Sawhney, W. Niblack, and J. Ashley, "Query by Image and Video Content: The QBIC system," *IEEE Computer*, pp. 23–32, September 1995.

[Fujiw78] E. Fujiwara, "Odd-Weight-Column b-adjacent Error Correcting Codes," *Transactions of the IECE Japan*, vol. E61, pp. 781–787, October 1978.

[Gaizo93a] F. Gaizotto, P. Paolini, and D. Schwabe, "HDM: A Model-based approach to Hypermedia Application Design," *ACM Transac. Info. Syst.*, vol. 11, pp. 1–26, 1993.

[Garzo93b] F. Garzotto, P. Paolini, and L. Mainetti, "Navigation Pattern in Hypermedia Databases," in *Hawaii Int'nl Conf. on System Sciences*, pp. 370–379, 1993.

[Gong93] L. Gongx, M. Loams, R. Needham, and J. Saltzer, "Protecting Poorly Chosen Secrets from Guessing Attacks," *IEEE Journal on Selected Areas on Comm.*, vol. 11, pp. 648–656, 1993.

[Grifn93] J Griffioen, R. Mehrotra, and R. Yavatkar, "A Semantic data model for Embedded Image Information," in *Proc. of 2nd Int'nl Conference on Infor. and Knowledg. Managmnt*, pp. 393–402, 1993.

[Grnbk92] K. Gronbaek and R. H. Trigg, "Design Issues for a DEXTER based hypermedia system," in *Proc of ECHT'92*, pp. 191–200, ACM Press, 1992.

[Grosk88] W. I. Grosky, "Image Database System," *IEEE Computer*, vol. 22, pp. 7–8, 1988.

[Grosk92] W. I Grosky, *Advances in Computer*, pp. 237–291 Academic Press, NY, 1992.

[Gudv95] V. N Gudivada and V. Raghavan, "Content-based Image Retrieval Systems," *IEEE Multimedia*, pp 18–31, September 1995.

[Gudv96] V. N. Gudivada, V V. Raghavan, and K. Vanapipat, *Multimedia Database Systems : Issues and Research Directions*, pp. 37–73. Springer, 1996.

[Gupta91] A. Gupta, T. Weymouth, and R. Jain, "Semantic Queries with Pictures: The VIMSYS model," in *17th VLDB Conference Proceedings*, 1991.

[Halas90] F Halasz and M. Schwartz, "The Dexter Hypertext Reference Model," in *Hypertext Standardization Workshop*, pp. 95–133, NIST, 1990

[Halas94] F G Halasz and M Schwartz, "The Dexter Hypertext Reference Model," *Comm ACM*, vol 37, pp 30–39, 1994

[Harao78] M Harao and S. Naguchi, "On some dynamical properties of finite Cellular Automata," *IEEE Trans. on Computers*, vol. C-27, no. 1, 1978.

[Hofst93] A. H. M. Hofstede and T. P. Weide, "Expressiveness in conceptual data modelling, " *Data and Knowledge Engg.*, vol. 10, pp 65–100, 1993.

[Hughe91] J. G. Hughes, *Object-Oriented Databases.* Prentice-Hall, Hoare Series, 1991.

[Isakw95] T. Isakowitz and P. Balasubraman, "RMM: A methodology to structured hypermedia design," *Comm. ACM*, vol. 38, pp. 34–44, 1995.

[Jabln96] D. P. Jablon, "Strong Password-only Authenticated Key Exchange," in *ACM SIGCOMM Computer Comm. Review*, pp. 5–25, 1996.

[Jack83] M. Jackson, *Systems Design.* Prentice-Hall International, Englewood cliffs, N.J., 1983.

[Jain94] R. Jain, "Semantics in Multimedia Systems," *IEEE Multimedia Magazine*, vol. 1, 1994.

[Kacmr91] C. J. Kacmar and J. J. Legett, "PROXHY: A process oriented extensible hypertext architecture," *ACM Transaction on Information System*, vol. 9, pp. 299–319, October 1991.

[Kahn67] D. Kahn, *The Codebreaker.* Macmillan Co., New York, 1967.

[Karmc96] A. Karmouch and J. Emery, "A Playback Schedule model for Multimedia Document," *IEEE Multimedia Magazine*, pp. 50–61, 1996.

[Kashp94] V Kashyap and A. Sheth, "Semantic-based Information Brokering," in *Int'nl Conf on Information and Knowledge Management*, 1994.

[Kashp95] V Kashyap and A. Sheth, "Semantic and Schematic Similarities between Database Objects," *IEEE Trans on Computers*, 1995

[Kashp96] V. Kashyap, K. Shah, and A. Sheth, *Multimedia Database System*, pp. 297–319. Springer, 1996

[Katti96a] R Katti, "A note on SEC/AUED codes," *IEEE Trans on Computers*, vol 45, pp 244–246, February 1996.

[Katti96b] R Katti and M Blaum, "An improvement on the construction of *t*-EC/AUED codes," *IEEE Trans on Computers*, vol. C-45, pp. 607–608, February 1996 .

[Klaus94] W. Klaus and A. Sheth, "Metadata for Digital Media," *SIGMOD Record, Special Issue on Metadata for Digital Media*, vol 23, no. 4, 1994.

[Kundu90] S. Kundu and S. M. Reddy, "On Systematic Error Correcting and All Unidirectional Error Detecting Codes," *IEEE Trans. on Computers*, vol. C-39, pp. 752–761, June 1990.

[Laih96] C. S. Laih, "On the analysis and design of Group Theoretical t-syEC/AUED codes," *IEEE Trans. on Computers*, vol. 44, pp. 103–108, January 1996.

[Lange94] D. Lange, "An Object-Oriented design method for Hypermedia information systems," in *Proc. of 27th Annual Hawaii Int'nl Conference on Systems Sciences*, pp. 204–213, January 1994.

[Leget94] J. J. Legett and J. I. Schnase, "Viewing Dexter with Open eyes," *Comm. ACM*, vol. 37, pp. 76–86, 1994.

[Lin83] S. Lin and D. J. Costello, *Error Control Coding : Fundamentals and Applications.* Prentice-Hall, Englewood Cliffs, N.J., 1983.

[Lin88] D. J. Lin and B. Bose, "Theory and Design of t-Error Correcting and $d(d > t)$-Unidirectional error detecting (t-EC/d-AUED) codes," *IEEE Trans. on Computers*, vol. C-36, pp. 444–439, April 1988.

[Lomas89] T. M. A. Lomas, L. Gong, J. H. Saltzer, and R. M. Needham, "Reducing risks from poorly chosen keys," *Operating System Review (Special issue)*, vol. 23, pp. 14–18, 1989.

[Mcrly88] K. C. McCurley, "A Key Distribution System equivalent to factoring," *Journal of Cryptology*, vol. 1, pp. 95–105, 1988.

[Melln73] G. E. Mellen, "Cryptography, Computers and Common Sense," *NCC*, vol. 42, pp. 569–579, 1973.

[Menez97] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Hand book of Applied Cryptography.* CRC Press, 1997.

[Merkl79] R. C. Merkle, *Secrecy, Authentication, and Public Key Systems.* UMI Research Press, Ann Arbor, Michigan, 1979.

[Merkl90] R. C. Merkle, "A Fast Software One-Way Hash Function," *Journal of Cryptography*, vol. 3, pp. 40–58, 1990.

[Millc70] J. C. P. Miller, "Periodic forests of stunted trees," *Trans. R. Soc. London, Ser.*, vol. A 266, no. 63, 1970.

[Misra92] S. Misra, *Theory and Application of Additive Cellular Automata for easily testable VLSI circuit design.* PhD thesis, I.I.T. Kharagpur, India, 1992.

[Montg90] Montgomery and Kumar, "Systematic Random Error Correcting and all Unidirectional error detecting codes," *IEEE Transaction on Computer*, vol. 39, pp. 836–840, 1990.

[Nandi94a] S. Nandi, *Additive Cellular Automata: Theory and Application for Testable Circuit Design and Data Encryption.* PhD thesis, I. I. T. Kharagpur, 1994.

[Nandi94b] S. Nandi, B. K. Kar, and P. P. Chaudhuri, "Theroy and Application of Cellular Automata in Cryptography," *IEEE Trans. on Computers*, vol. 43, pp. 1346–1357, December 1994.

[Neuma66] J. V. Neuman, *The theory of self-reproducing Automata, A. W. Burks ed.* Univ. of Illinois Press, Urbana and London, 1966.

[Nikol86] D. Nikolos, N. Gaitanis, and G. Philokyprou, "Systematic t-Error-Correcting/All Unidirectional Error Detecting Codes," *IEEE Trans. on Computers*, vol. C-35, pp. 394–402, May 1986.

[Nikol91] D. Nikolos, "Theory and design of t-error correcting/d-error detecting ($d > t$) and all Unidirectional error detecting codes," *IEEE Trans. on Computers*, vol. C-40, pp. 132–142, February 1991.

[NIST93] NIST. FIPS. PUB. YY, "Secure Hash Standard," National Instt of Standards and Technology, April 1993.

[Noll91] J. Noll and W. Scacchi, "Integrating diverse information repositories: Distributed Hypertext approach," *IEEE Computer*, vol. 24, pp. 38–45, December 1991.

[Nybrg93] K. Nyberg and R. Reuppel, "A new Signature scheme based on the DSP giving message recovery," in *ACM Conf on Computer and Communication Security*, pp. 58–61, ACM, 1993.

[Oorsc91c] P. V. Oorschot and M. Weiner, "A known plaintext attack on two-key triple encryption," in *Proceedings of EUROCRYPT'90, LNCS 473*, pp. 318–325, Springer-Verlag, 1991.

[Oorsc92] P. C. V. Oorschot, *Contemporary Cryptology: The Science of Information Integrity*, pp. 289–322. IEEE Press, 1992.

[Oorsc96] P. C. V. Oorschot and M. J. Wiener, "On Diffie-Hellman key agreement with short exponent," in *Proceedings of EUROCRYPT'96*, Springer-Verlag, 1996.

[Palch97] P. P. Choudhuri, D. R. Choudhuri, S. Nandi, and S. Chattopadhyay, *Additive Cellular Automata: Theory and Application, Vol I.* IEEE CS, 1997.

[Panur96] P. Panurach, "Money in electronic commerce: Digital Cash, electronic fund transfer, and Ecash," *Comm. ACM*, vol. 39, pp. 45–50, 1996.

[Pearl89] A. Pearl, "Sun's link service : A protocol for open linking," in *HYPERTEXT'89*, pp. 137–146, 1989.

[Pohln78] S. Pohling and M. Hellman, "An improved algorithm for computing logarithm over GF(p) and its cryptographic significance," *IEEE Transac. Info. Theory*, vol. IT-24, pp. 106–110, January 1978.

[Pradh80] D. K. Pradhan, "A new class of error-correcting/detecting codes for fault-tolerant computer applications," *IEEE Trans. on Computers*, vol. C-29, pp. 471–481, June 1980.

[Preen94] B. Preneel, "Cryptographic hash function," *Transaction on Telecommunication*, vol. 5, pp. 431–448, 1994.

[Pries86] W. Pries, A. Thanailakis, and H. C. Card, "Group properties of Cellular Automata and VLSI applications," *IEEE Trans. on Computers*, vol. C-35, pp. 1013–1024, December 1986.

[Rabin78] M. O. Rabin, R. DeMillo, D. Dobkin, A. Jones, and R. Lipton, *Digitalised Signatures : Foundation of Secure Communication*, pp. 155–168. Academic Press, 1978.

[Rao89] T. R. N. Rao and E. Fujiwara, *Error-control Coding for Computer Systems.* Prentice-Hall, Englewood Cliffs, N.J., 1989.

[Reupl86] R. U. Reuppel, *Analysis and Design of Stream Ciphers.* Springer, Berlin, 1986.

[Reupl90] R. U. Reuppel, "Kee agreement based on functional composition," in *Advances in Cryptology - EUROCRYPT'89, LNCS 434*, pp. 423–428, Springer-Verlag, 1990.

[Rhee94] M. Y. Rhee, *Cryptography and Secure Communication.* Mc GrawHill, 1994.

[Rishe93] N. Rishe, "A methodology and tool for top-down relational database design," *Data and Knowledge Engineering*, vol. 10, pp. 259–291, 1993.

[Richa72] D. Richardson, "Tessellations with local transformations," *J. Comput. Systems Sci.*, vol. 6, pp. 373–388, 1972.

[Rivst76] R. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining Digital Signatures and Public Key Cryptosystem," *Comm. ACM*, vol. 21, pp. 120–126, February 1976.

[Rivst92] R. Rivest, "The MD5 Message Digest algorithm," in *RFC 1321*, April 1992.

[Salim86] S. Ali and Vijayan, "Keoladeo National Park, Ecological study, Summary Report 1980-85," tech. rep., Bombay Natural History Society, 1986.

[Sam79] E. Sam, "Musical Cryptography," *Cryptologia*, vol. 3, pp. 193–201, October 1979.

[Schni96] B. Schneier, *Applied Cryptography.* John Willey and Sons, 1996.

[Schno91] C. P. Schnorr, "An efficient Cryptographic Hash Function," in *Advances in Cryptology: CRYPTO'91*, Springer-Verlag, 1991.

[Schno93] C. P. Schnorr, "FFT-Hash II: Efficient Cryptographic Hashing," in *Advances in Cryptology: EUROCRYPTO'92*, pp. 45–54, Springer-Verlag, 1993.

[Schns93a] J. I. Schnase, J. J. Legett, D. I. Hicks, and D. I. Szabo, "Semantic Data Modelling of Hypermedia Associations," *ACM Transaction Information System*, vol. 11, pp. 27–50, January 1993.

[Schns93b] J. I. Schnase, J. J. Legett, D. L. Hicks, P. J. Nurnberg, and J. A. Sanchez, "HB1: Design and Implementation of a Hyperbase Managmnt System," *EPODD*, vol. 6, pp. 35–63, March 1993.

[Schut90] H. A Schut and N. A. Streitz, "Hyperbase: A Hypermedia Engine based on a Relational database management system," in *Hypertext INRIA, France*, pp. 95–108, 1990.

[Sebry89] J. Seberry and J. Pieprzyk, *Cryptography. An Introduction to Computer Security.* Prentice-Hall of Australia, 1989.

[Serra90] M. Serra, J. C. M. T. Slater, and D. M. Miller, "Analysis of one dimensional Cellular Automata and their aliasing probabilities," *IEEE Trans. on CAD*, vol. 9, pp. 767–778, July 1990.

[Shann49] C. E. Shannon, "Comm. theory of secrecy systems," Tech. Rep. 28, Bell Syst. Tech. J., 1949

[Shamr79] A. Shamir, "On the Crypto-complexity of Knapsack System," in *Proc. 11th Annual ACM Sympo., Theory of Computing*, pp. 118–129, 1979.

[Shklf93] D. E. Shakelford, J. B. Smith, and F. D. Smith, "The Architecture and Impelmentation of a Distributed Hypermedia Storage System," in *Hypertext'93*, November 1993.

[Shukl96] J. B Shukla and B. Dubey, "Effect of changing Habitat on Species: Application to Keoladeo National Park," *Ecological Modelling*, vol. 86, pp. 91–99, 1996.

[Simon92] G. J. Simmons, *Contemporary Cryptology: The Science of Information Integrity*, pp. 379–419. IEEE Press, 1992.

[Simon95] G. J. Simmons, *Codes and Ciphers . Cryptography and Coding*, pp. 383–394. IMA, 1995

[Sinko66] A Sinkov, *Elementary Cryptanalysis.* Math Assoc., 1966.

[Sisla94] A. P. Sistla, C. Yu, and R. Haddad, "Reasoning about Spatial Relationship in Picture Retrieval System," in *20th VLDB'95 Proceedings*, 1994.

[Sisla95] A. P Sistla and C. Yu, "Similarity based retrieval of pictures using Indices on Spatial Relationships," in *21st VLDB'95 Proceedings*, 1995.

[Smith91] J B Smith and F. D. Smith, "ABC. A hypermedia system for artefact based collaboration," in *HYPERTEXT'91*, pp. 179–192, ACM Press, 1991.

[Steve74] P. S. Stevens, *Patterns in Nature.* Little, Brown, Boston, 1974.

[Stein95] M. Steiner, G. Tsudik, and M. Waidner, "Refinement and Extension of Encrypted Key Exchange," *Ooperating Systems Review*, vol. 29, pp. 22–30, July 1995.

[Stots89] P. D. Stotts and R. Furuta, "Programmable Browsing Semantics in Trellis," in *Hypertext'89*, pp. 27–42, ACM Press, 1989.

[Subra96] V. S. Subramanian and S. Jajodia, *Multimedia Database Systems: Issues and Research Directions.* Springer, 1996.

[Tamur84] H. Tamura and N. Yokoya, "Image Database Systems: A Survey," *Pattern Recognition*, vol. 17, no. 1, pp. 29–43, 1984.

[Tanaka88] M. Tanaka and T. Ichikawa, "A Visual User Interface for Map Information Retrieval based on Semantic Significance," *IEEE Transac. on Software Engineering*, vol. 14, no. 5, pp. 666–670, 1988.

[Tao88] D. L. Tao, C. R. P. Hartman, and P. K. Lala, "An efficient class of Unidirectional error-correcting/detecting codes," *IEEE Trans. on Computers*, vol. C-37, pp. 879–882, July 1988.

[Teory86] T. J. Teorey and W. Yang, "A logical design methodology for Relational Databases using EER model," *ACM Computer Survey*, vol. 18, no. 2, pp. 197–222, 1986.

[Toffo77] T. Toffoli, "Computation and construction universality of reversible cellular automata," *J. Comput. System Sci.*, vol. 15, p. 213, 1977.

[Vaude95] S. Vaudenay, "On the need for multipermutations: Cryptanalysis of MD4 and SAFER," in *Fast Software Encryption, 2nd Int'nl Workshop on (LNCS 1093)*, pp. 286–297, Springer-Verlag, 1995.

[Welsh88] D. Welsh, *Codes and Cryptography.* Clarendon Press, Oxford, 1988.

[Wendy96] W. Hall, H. Davis, and G. Hutchings, *Rethinking Hypermedia, The Microcosm approach.* Kluer Academic Publisher, 1996.

[Wickr96] S. B. Wicker, *Error Control System for Digital Communication System and Storage.* Prentice Hall, NJ., 1996.

[Wiil92] U. K. Wiil and J. J. Legett, "Hyperform: Using Extensibility to develop a dynamic, open and distributed Hypertext System," in *Proc of ECHT'92*, pp. 251–261, ACM Press, 1992.

[Wills75] S. J. Willson, "On the Ergodic theory of Cellular Automata," *Math. System Theory*, vol. 9, pp. 132–141, 1975.

[Wldvg93] C. P. Waldvogel and J. L. Massey, "The Probability Distribution of the Diffie-Hellman key," in *AUSCRYPT'92, LNCS 718*, pp. 492–504, Springer-Verlag, 1993.

[Wolfr83] S. Wolfram, "Statistical mechanics of Cellular Automata," *Rev. Mod. Phys.*, vol. 55, pp. 601–644, July 1983.

[Wolfr84] S. Wolfram, O. Martin, and A. M. Odlyzko, "Algebraic properties of Cellular Automata," *Mathematical Physics*, vol. 93, pp. 219–258, 1984.

[Wolfr85a] S. Wolfram, "Twenty problems in the theory of Cellular Automata," *Physica Scripta*, vol. T9, pp. 170–183, 1985.

[Wolfr85b] S. Wolfram, "Undecidability and Intractability in Theoretical Physics," *Phys. Rev. Lett.*, vol. 54, pp. 735–738, 1985.

[Yacob90] Y. Yacobi and Z. Shmuely, "On Key Distribution Systems," in *Advances in Cryptology-CRYPTO'89 (LNCS 435)*, pp. 344–355, Springer-Verlag, 1990.

[Yacob91] Y. Yacobi, "A Key Distribution Paradox," in *Advances in Cryptology-CRYPTO'90 (LNCS 537)*, pp. 268–273, Springer-Verlag, 1991.

[Yamad71] H. Yamada and S. Amoroso, "Structural and Behavioral Equivalences of Tessellation Automata," *Infor. Contr.*, vol. 18, pp. 1–31, 1971.

[Yang98] C. Yang and C. Laih, *DCm Codes for Constructing t-EC/AUED Codes*, *IEEE Trans. on Computer*, vol 47-4, pp. 492–495, April 1998.

[Zheng93] Y Zheng, J Pieprzyk, and J. Seberry, "A One-way Hashing Algorithm with variable length of output," in *Advances in Cryptology-AUSCRYPT'92 (LNCS 718)*, pp 83–104, 1993.

Biodata

Dhruba Kumar Bhattacharyya was born on 25th day of February, 1966 in the state of Assam, India. He received the *Bachelor of Science* (*Honours*) degree in *Physics* in 1987 and the *Master degree* in *Computer Application* in 1991, both from Dibrugarh University. During 1992-1995, he was faculty at the Computer Engineering and Application Department, Jorhat Engineering College, Assam, India. Since March 1995, he has been serving as a faculty at Computer Science Department, Tezpur University (Central), Assam, India. His research interests include Data security, Error control coding, Data modelling and Content based image retrievals. Presently, he has also been engaged as an Investigator in a research project on *Content based Image Retrieval System*, funded by AICTE, Govt of India.

Publications of Author

[1] D. K. Bhattacharyya and B. Dubey, "Multimedia in Ecosystem Modelling : An Application to Keoladeo National Park," in *Proc of SIP'98, IASTED*, Annecey, France, 1996.

[2] D. K. Bhattacharyya, M. C. Bora, and S. Nandi, "An Architecture of Multimedia Information System," in *Trends in ADvanced COMPuting*, pp. 178–181, Tata McGraw Hill, 1996.

[3] D. K. Bhattacharyya, S. N. Pandit, and S. Nandi, "A New Enciphering Scheme," in *Proc on Int'nl Conf on Telematics*, pp. 1–11, NERIST, Nirijuli, India, 1997.

[4] D. K. Bhattacharyya, S. Nandi and M. C. Bora, "Efficient Design of One-way Hash Function using Cellular Automata," in *Current Trends in ADvanced COMPuting*, Tata Mc GrawHill, 1997.

[5] D. K. Bhattacharyya, S. Nandi and M. C. Bora, "Application of RMDM and its Drawbacks," in *Proc. of CUPUM'97*, pp. 866-877, Narosa, 1997.

[6] D. K. Bhattacharyya and S. Nandi, "A New Class of t-EC/d-ED $(d > t)$/AUED Codes," in *Proc. of PRFTS'97*, pp. 41-46, IEEE CS, 1997.

[7] D. K. Bhattacharyya and S. Nandi, "A Efficient Class of SEC-DED-AUED Codes," in *Proc. of I-SPAN'97*, pp. 410-416, IEEE CS, 1997.

[8] D. K. Bhattacharyya and S. N. Pandit, 'Basin Structure with Ultimate Periodicity," in *Proc. of Internet for India*, pp. 95-100, Hyderabad, India, 1997.

[9] D. K. Bhattacharyya and S. Nandi, "Theory and Design of SEC-DED-AUED Codes," in *IEE(E) Proc*, vol. 145-2, pp. 1-6, March, 1998.

[10] D. K. Bhattacharyya and S. Nandi, “A Secure Block Ciphering Scheme,” in *Proc of ISIT’98*, IEEE CS, August, 1998.

[11] D. K. Bhattacharyya, S. Nandi and M. C. Bora, ”An Image Database System Architecture”, communicated to *COMAD98.*

[12] D. K. Bhattacharyya, S. Nandi and M. C. Bora, ”Password-only Authenticated Key Exchange using Cellular Automata”, communicated to *ADCOMP98.*

[13] R. Das, D. K. Bhattacharyya and S. Nandi, ”A New Image Enciphering Scheme”, accepted in *SIP’98* organised by IASTED, to be held at Las Vegas, during 28-31 October, 1998.