Data Mining of Sequences through Extended Rough Set Theory - Protein Sequences

A thesis submitted in partial fulfillment of the requirements for the award of the degree of

DOCTOR OF PHILOSOPHY

by

RAMADEVI YELLASIRI
[Reg. No. 2KMCPC05]

Department of Computer & Information Sciences
School of Mathematics and Computer / Information Sciences
University of Hyderabad
Hyderabad - 500 046, Andhra Pradesh, INDIA

July, 2008

Department of Computer & Information Sciences
School of Mathematics and Computer / Information Sciences
University of Hyderabad
Hyderabad – 500 046, Andhra Pradesh, INDIA

C E R T I F I C A T E

This is to certify that the thesis titled, **Data Mining of Sequences through Extended Rough Set Theory - Protein Sequences**, has been carried out by **Ramadevi Yellasiri** under our supervision and submitted in partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy in Computer Science,** University of Hyderabad, Hyderabad. The results presented in this thesis have not been submitted to any other University or Institute for the award of the same degree.

Place: Hyderabad
Date: 18/07/2008.

Prof.C.Raghavendra Rao
(Supervisor)

Dean
School of Mathematics and
Computer /Information Sciences
University of Hyderabad.

(Prof. Arun Agarwal)
Head
Department of Computer &
Information Sciences,
University of Hyderabad.

DECLARATION

I hereby declare that the work embodied in this thesis has been carried out by me under the supervision of **Prof. C. Raghavendra Rao**, Department of Computer and Information Sciences, University of Hyderabad, Hyderabad. I also state that the result presented in this work has not been submitted to any other University or Institute.

Place: Hyderabad (**Ramadevi Yellasiri)**

Date: 18/07/2008.

ACKNOWLEDGEMENTS

At the outset, I wish to express my profound gratitude to **Prof. C. Raghavendra Rao**, for providing continuous guidance throughout the research work and its successful completion. His critical observations and valuable comments were of great help in finalizing this work and report.

I wish to thank **Dr. K. Chandra Sekhar Reddy** for his initial guidance and encouragement to take up this work in the emerging field of bioinformatics.

It is a great pleasure to convey thanks to **Prof T. Amaranath**, Dean, MCIS, and **Prof. Arun Agarwal**, Head, DCIS for extending their cooperation and encouragement by providing all the amenities during the period of my work.

I am honored to thank distinguished Professors, **Prof. A.K.Pujari**, **Dr. S. Bapi Raju**, **Prof. K. Narayana Murthy**, and **Dr Rajeev Wankar** Doctorial Review Committee members for their valuable suggestions and encouragement in the completion of this work. I also thank the teaching and non-teaching staff of the Department of Computer & Information Sciences for the help they have rendered.

I wish to take this opportunity to place on record my sincere gratitude to **Prof. I. Ramachandra Reddy**, Director and **Prof B. Chennakesava Rao**, Principal, Chaitanya Bharathi Institute of Technology (CBIT), Gandipet, Hyderabad for their constant encouragement in completing my research.

I am also grateful to **Prof G.V.Anjaneyulu**, Head, Dept. of CSE, CBIT who has reposed confidence in me and permitted me to proceed with my research despite pressing academic schedules in general and during the thesis period in particular.

I thank **Dr. Hari RamaKrishna**, **N.Ramadevi**, **Dr. M. Seetha** , and **T.Prathima** for their continuous support and participation in the intellectual discussions, which were of great help in completion of my research.

I take this opportunity to thank all my colleagues in the Department of CSE, CBIT for extending their help and encouragement.

I take this opportunity to thank **K. Ramesh Babu**, ADE/ APTransco and **G. Lakshman Raju** ADE/ APTransco for extended technical help in the earlier stages of designing the modules.

I also thank **Sri S. Madhusudhan Rao** for his valuable suggestions, while presenting the work in this thesis.

I am highly thankful to my husband **D. Brahma Reddy**, who has been the source of inspiration throughout and who stood by me, with his unstinted support and care. I also appreciate the patience, love and understanding shown by my children **Samyukta** and **Gopi Madhav Reddy**, my mother-in-law and parents throughout my work and especially during my thesis writing. I thank all other members of my family and friends who have supported me all through.

Last, but not the least, I sincerely thank all the people, mentioned or not mentioned inadvertently who have been helpful in one way or the other in the completion of my thesis.

Ramadevi Yellasiri

ABSTRACT

In the recent times, a large amount of data relating to new protein sequences are progressively being accumulated in various databases. The challenging task for researchers in bioinformatics is to classify these proteins into families based on their structural and functional properties, thereby helping in the prediction of the functions of these new protein sequences.

The problem of classifying/ identification of proteins has been addressed by Hulo [Hulo, 2004], Attwood [Attwood, 2003], Bateman [Bateman 2003]. They considered information of significant sites, motifs, profiles, multiple sequence alignment, etc., for classifying proteins.

Protein sequences are of varying length, comprising of twenty Amino Acids (AA). The present study considers the problem of classification/ identification of a questioned protein with the composition, frequency of occurrence of amino acids and neighborhood characteristics of amino acids (which was not attempted till date by researchers).

The processes developed in this thesis for the purpose of classification/ identification has led to various contributions for the reduction of the complexity and building knowledge.

The methodology in the present work, enables to classify a queried protein in levels i) Family, ii) Class, and iii) Subclass and thus leading to search space reduction. The present study gives a novel approach for handling problems related to sequences, in particular proteins for constructing derived information (in structured form), and henceforth extracting the rules. These rules can be utilized in Meta searching

The techniques developed in the present study, *viz*., i) Sequence Arithmetic, to identify family information, ii) hybridization of Rough Sets with Sequence Arithmetic helps to identify the class/ classes, iii) hybridization of Rough Sets and features based on

neighborhood characteristics to classify the protein further into subclasses, and iv) finally the technique of hybridizing the concept lattice with neighborhood characteristics further maps the queried protein into sub/ sub-sub classes. It is novel soft computing contribution for multilevel search space reduction.

The developed techniques have produced various bye-products, which will be handy in Decision Support Systems and Knowledge Support Systems. They are

- Sequence Arithmetic Knowledge Database (SAKD) - The rules generated from the innovative technique, Sequence Arithmetic (SA), have been stored in this database. They are useful in family identification.
- Reduct based Decision Tree Rules Database (RDTRD) - Hybridization of Rough Sets and decision tree in a novel way is called Reduct based Decision Tree (RDT). The decision rules generated from the RDT have been stored in this database.
- Neighborhood Association Rules Database (NARD) – It contains rules generated from the hybridization of Rough Sets and features based on neighborhood analysis.
- Concept lattice Association Rules Database (CARD) – It contains Association rules generated from hybridization of Concept Lattice and neighborhood characteristics of amino acids.

The integrated soft tool of all the modules, *viz.*, i) Sequence Arithmetic, ii) Reduct based Decision Tree, iii) Neighborhood Analysis, and iv) Concept Lattice modules has been named as Rough Set Protein Classifier (RSPC) in this work. It is also demonstrated that the RSPC accelerates the process of classification/ identification by reducing the domain search space level-by-level.

Two protein families, *viz.*, Myoglobin and G-Protein Coupled Receptors (GPCR) have been considered for demonstrating the RSPC tool developed in this thesis.

The empirical study reveals, based on randomly selected sample size of 60 sequences from the 1440 proteins, the following observation:

The Sequence Arithmetic module helps in reducing the search by about 40%. The coupling of RDT to SA further reduced the search space by 10% making it to 50%, implying that one needs to compare 50% of the proteins in the database for classifying a questioned protein. By considering the spatial information through the neighborhood associations, the search space has been further reduced to 37%. The exploitation of power of concept lattice further reduced the search space to 9%.

It is also observed in levels, that the questioned protein is one among the confused proteins set, resulting in the search space reduction from 100% to 9% without losing the accuracy of detection. It is also observed that classification accuracy rate using RSPC is around 97.7%.

The Rough Set Protein Classifier tool is applicable to sequences and helps in compiling information of the sequences in tables, which help in adopting feature extraction tools in particular, Rough Set tools. Though it is demonstrated by considering only two families, the power of Rough Set Protein Classifier reducing the search space from 100% to 9% without compromising the accuracy of classification/ identification is applicable for any problems associated with sequences (systems involving sequences). Further, these modules will be versatile for the machine-learning environment.

TABLE OF CONTENTS

List of Tables

List of Figures

Glossary

Symbol Definition	Abbreviations
AA	Amino Acids
BAM	Binary Association Matrix
BAMD	Binary Association Matrix with Distance
CAR	Concept lattice Association Rules
CARD	Concept lattice Association Rules Database
CL	Concept Lattice
DNA	Deoxyribonucleic Acid
GPCR	G-Protein Coupled Receptors
HMM	Hidden Markov Model
LFC	Least Frequent occurring Character
NA	Neighborhood Associations/ Analysis
NAR	Neighborhood Association Rules
NARD	Neighborhood Association Rules Database
NM	Neighborhood Matrix
PA	Predominant Attributes
PDB	Protein Data Bank
RCA	Reduct Computation Algorithm
RDT	Reduct based Decision Tree
RDTR	Reduct based Decision Tree Rules
RDTRD	Reduct based Decision Tree Rules Database
RNA	Ribonucleic Acid
RS	Rough Set
RSPC	Rough Set Protein Classifier
SA	Sequence Arithmetic
SAR	Sequence Arithmetic Rules
SAKD	Sequence Arithmetic Knowledge Database
SVM	Support Vector Machine

CHAPTER I
INTRODUCTION

Proteins are building blocks of cells and tissues; they play a vital role in executing and regulating many biological processes. Amino Acids (AA) are the building blocks of all proteins. There are 20 AA that combine to generate uncountable number of protein sequences. In addition to these, there are two more characters {B, Z} that appear in these sequences, which are substitutions/ derivations for some amino acids. In reality, only small subsets of sequences appear [Mount, 2001]. Proteins are categorized into families, classes, subclasses based on functions and structures [Hooman, 2005].

A large number of new proteins are being added to genomic databases by research community as and when they are discovered. Several methods (PROSITE, PRINTS, PFAM, etc.,) are available to classify these proteins into families based on their structural and functional properties. Precisely, there is an acute need for development of faster, more sensitive and accurate methods to meet the present challenges of classification.

PROSITE, PRINTS and PFAM are protein classification methods and they utilize multiple sequence alignments to build Hidden Markov Models. PROSITE is a database consisting of information of significant sites, patterns, and profiles that specify different protein families [Hulo, 2004]. PRINTS is a database of protein fingerprints [Attwood, 2003]. Fingerprints are sets of short sequence motifs (patterns) conserved among members of a protein family. PFAM is a database of alignments and profile-hidden markov models of protein families [Bateman, 2003]. They can predict protein functions only if sequences are sufficiently conserved. Although sequences share some structural similarities, these methods may fail, lest there is an inadequate sequence similarity. In general, they require large amount of time for building models as well as for predicting functions based on them.

The generation of reliable multiple alignments becomes problematic, while dealing with extremely diverged protein sequences. G-Protein Coupled Receptors (GPCR) belongs to highly diverged protein families and they are transmembrane

proteins. They play a significant role in various signal transmission processes, which are directly associated with a variety of human diseases [Predix, 2000].

Protein classification algorithms use multiple sequence alignment, profiles, motifs, short sequences, etc,. Profiles and motifs need expert knowledge for attributing structural or functional aspects of a protein. It is possible to derive results through the existing methods based on profiles and motifs with the aid of a good expert system. The simple machine learning methods will not be helpful in the absence of expert intervention.

Machine-learning systems can be developed with

a) Expert interference (Human Computer Interface).
b) Machine learning systems by naive methods.

The naive machine learning methods build rules for decision-making. These rules may not be physically interpretable, but it doesn't entail expert interference and provides flexibility to build the system automatically.

1.1 PROBLEM SPECIFICATION

This research aims at building rules and ensuing rule based Expert System for protein classification with the following objectives:

1. Protein Classification / Identification.
2. Search Space Reduction and Creation of the Databases.
3. Rule Extraction.

A motif is a consecutive string of amino acids which frequently occur in a given protein sequence whose general character is repeated, or conserved, in all sequences in a multiple alignment at a particular position. Motifs motivated the present study to consider the occurrence and frequencies of amino acids as a characteristic of a given protein. These characteristics have been used for introducing novel sets in Sequence Arithmetic (SA) module.

Profiles are position-specific scoring table that encapsulates the sequence information within complete alignments. They define the residues allowed at given positions; which positions are conserved and which degenerate; and which positions,

or regions, can tolerate insertions. The neighborhood characteristics of an amino acid in a protein have been introduced by the motivation from the concepts of profiles. The characteristics based on the neighborhood introduced in this thesis accounts for extraction of location effects, which has not been possible to acquire from Sequence Arithmetic. Therefore, the neighborhood analysis accounts for extracting spatial information.

1.2 METHODOLOGY

The methodology is predominantly based on set theory, relations, classifiers and binarization processes.

A protein is a sequence; it is a collection of characters with some frequencies. It contains character information and positional information. A primitive information system, which is the set of characters that occurs in the sequence, has been build. The absence of character is a potential for grouping sequences. A schematic information building has been proposed, developed and demonstrated, which is named as Sequence Arithmetic.

The family, class, subclass information of the sequence is wrapped to develop a classifier. A novel way of constructing decision tree has been proposed by hybridizing the concepts of Rough Sets and decision tree. This tree is called Reduct based Decision Tree, which differs from the hybrid Reduct based Decision Tree proposed by Minz [Minz, 2005]. The method of construction of the proposed decision tree and hence, extraction of rules is discussed and demonstrated in the present work.

The Sequence Arithmetic is free from arrangement of amino acids in sequence, thus the measures/ decisions arrived have been insensitive to the spatial variation. For accounting the spatial aspects (in weak sense), a neighborhood of an amino acid has been introduced, and hence binarizing the neighborhood matrix develops computation of neighbor's association of a character with other characters. A representation of this derived information is named as Binary Association Matrix (BAM). Class information of the protein is wrapped to this Information System and classifiers are developed by building Reduct based Decision Tree (RDT) in the present thesis.

To reduce the confused set of proteins further the BAM of the class has been subjected to concept lattice and hence rules have been generated. A new decision tree constructed based on Rough Sets hybridization has been validated by considering standard datasets.

A Schematic method of quantification of performance of multilevel classifier has been introduced.

1.3 CONTRIBUTIONS

The following conceptual contribution *viz.,* i) Sequence Arithmetic, ii) Neighborhood Analysis, and iii) Binary Association Matrix have been made in the present research. A novel tool called Reduct based Decision Tree has been developed by hybridizing the Rough Sets and decision tree.

- Different sets: A-Set, D-Set and P-Set as characteristics of sequences of a family have been introduced under Sequence Arithmetic.
- Rule base developed based on the derived information related to P-Sets of the sequences, which has been utilized to derive class information.
- Binary Association Matrix construction methodology designed and hence rule base derived for class/ subclass identification.
- The subclass identification rules extracted by adopting concept lattice tools on Binary Association Matrix.

1.4 OVERVIEW OF ROUGH SET PROTEIN CLASSIFIER

The Rough Set Protein Classifier is explained with a block diagram. The demonstration details are given in the sections to follow.

1.4.1 Rough Set Protein Classifier

Rough Set Protein Classifier consists of four modules, *viz.*, i) Sequence Arithmetic module, ii) Reduct based Decision Tree module, iii) Neighborhood Associations/ Analysis module, and 4) Concept Lattice module. The Rough Set Protein Classifier process is shown in Figure.1.1 with unknown sequence ‘y’ as the input and set of sequences < S > as output.

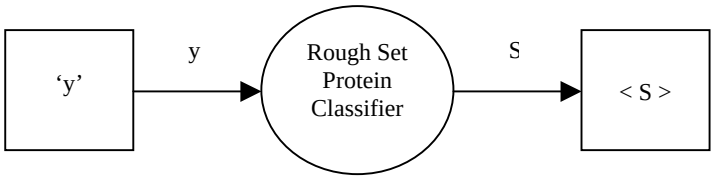

Figure 1.1: Rough Set Protein Classifier

The different modules of the Rough Set Protein Classifier are depicted in Figure 1.2. Given an unknown sequence 'y' to the Sequence Arithmetic module, the family information < F > is obtained by mapping with the information in Sequence Arithmetic Knowledge Database.

The family information < F > along with sequence 'y' is given to Reduct based Decision Tree module to get class < C > information. Reduct based Decision Tree rules have been used for extracting class information. These rules have been stored in Reduct based Decision Tree Rules Database.

Neighborhood associations have been performed to obtain localized information. This localized information has been used in the construction of Binary Association Matrix in the Neighborhood Associations module.

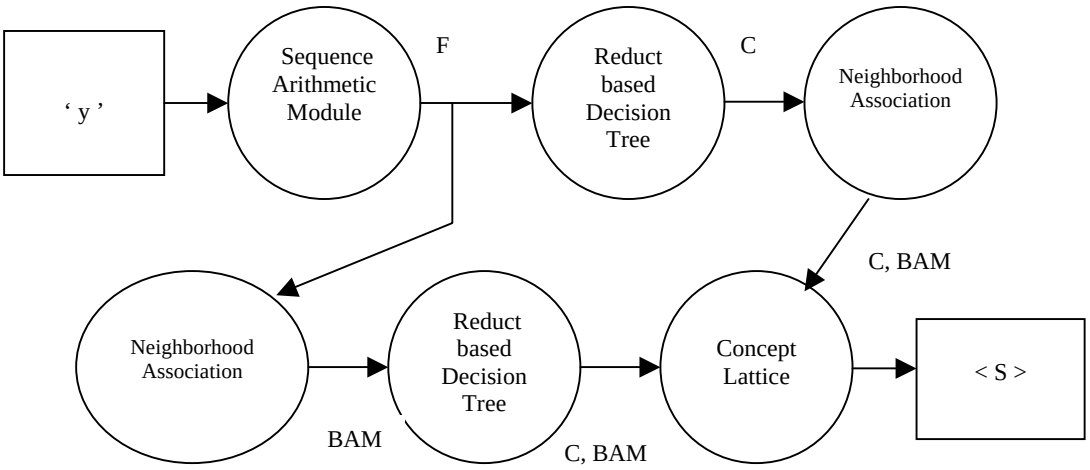

Figure 1.2: Different modules in the Rough Set Protein Classifier process

The Binary Association Matrix has been used in the construction of concept lattice. The node information of the concept lattice assigns the relationship between

the attributes and the objects. Set of sequences < S > within a class closer to unknown sequence ‘y’ is the result of Concept Lattice module.

1.4.2 Demonstration of Rough Set Protein Classifier

G-Protein Coupled Receptors protein sequences and Myoglobin protein sequences are taken for experimentation. G-Protein Coupled Receptors consists of 3896 sequences divided into 5 classes [Brookhaven NL] of which 1041 have been taken for demonstration. Samples of 399 sequences out of available 727 sequences of Myoglobin are considered. These are divided into 36 classes. The derived database of various modules is a one-time process. They have been computed and stored for knowledge extraction (Appendices A, B and C).

The development of multilevel tool, Rough Set Protein Classifier is elucidated in Chapters III, IV, V and VI.

It has been observed that Rough Set Protein Classifier is effective in reducing domain search space.

1.5 ORGANIZATION OF THE THESIS

Chapter II deals with a thorough study of proteins, protein structures and various classification algorithms, followed by a brief overview of the data mining algorithms. The composition of proteins and their importance has been emphasized. A description of two datasets G-Protein Coupled Receptors and Myoglobin proteins that are considered for experimentation are presented. Foundations of set theory, Rough Set Theory, Decision Tree, Concept Lattice are introduced in this chapter.

Chapter III presents arrangement invariant methodology named as Sequence Arithmetic evolved based on the concept of motifs of a sequence and subsequently used for family (families) identification.

Chapter IV demonstrates a typical organization of derived information based on Sequence Arithmetic (confining to P-Sets) and is wrapped with protein decision information like family, class to result in a decision table. The process of extracting the predominant attributes (approximate reducts) is demonstrated. Hybridization of the Rough Sets (approximate reduct) with decision tree is named as Reduct based

Decision Tree (RDT) has been developed and demonstrated. Classification rules have been generated based on RDT and organized in a database called Reduct based Decision Tree Rules Database (RDTRD). These rules assist in attaching a new protein to a class within a family. The performance evaluation of RDT has been carried out on standard datasets.

Chapter V deals with extraction of localized information. To reduce the search space for a given queried protein, one needs to solicit additional information from the protein. Neighborhood metric for other amino acids has been defined and compiled in a matrix for pair of amino acids with one as center and other as neighbor for a predefined neighbor's distance. A binarization scheme has been evolved to derive a Binary Association Matrix (BAM). A novel way of constructing the decision table has been introduced by considering a row corresponding to selected center from BAM for a protein and appending the class information to it, for a set of proteins of the training set. Rules are compiled by administering RDT on the derived decision table. The rules are organized in a database named as Neighborhood Associations Rules Database (NARD). An illustrative demonstration has been carried out on few test proteins.

Chapter VI deals with a hybridized approach of Neighborhood Associations and Concept Lattice. The Galois concept lattice tool has been implemented on BAM of a class of proteins. The output of the tool has been compiled and stored in a database called Concept lattice Association Rules Database (CARD).

Chapter VII presents the design and analysis issues associated with novel tool RSPC developed in this thesis. Performance Evaluation of RSPC has been demonstrated empirically.

Chapter VIII summarizes the contributions and presents future scope of the work.

The thesis has four Appendices A, B, C and D. Schematic compilations of databases of various levels of RSPC are present in Appendices A, B and C. Appendix D gives the Performance Evaluation of RSPC. A few samples of the rules are appended. The complete lists of rules are available in the enclosed CD.

CHAPTER II
LITERATURE SURVEY

Classification of sequences has been a challenging task for researchers. Sequence is defined in many ways depending upon the instance, location and domain. Proteins are sequences of varying length and consist of Amino Acids (AA). A brief introduction to proteins and various classifying algorithms are elucidated in Section 2.1. Different protein classification algorithms based on sequence similarity, full sequence similarity, domain and motifs, profiles, structure, etc., including the data mining techniques are elucidated in Section 2.2. The limitations of existing algorithms are discussed in Section 2.3.

Basic concepts of set theory are used to perform various operations on a sequence. These concepts are used to extract knowledge. Set theory [Mott, 1999] is discussed in Section 2.4.

A Rough Set is a formal approximation of a crisp set in terms of a pair of sets, which give the lower and upper approximation of the original set. Rough Set concepts and their properties are discussed with illustrations in Section 2.5. Decision Tree induction methods are presented in Section 2.6. A brief introduction to the construction and interpretation of Concept Lattice is explained in Section 2.7. The datasets considered for demonstration of methodology is presented in Section 2.8.

2.1 PROTEIN

Proteins are molecules that form much of the functional and structural machinery of every cell in every organism. The name *protein* is derived from the Greek word ‘protos’, which means first. Proteins are the most fundamental substances of life, as they are the key components of the protoplasm of all cells. They are building blocks of cells and tissues, and also execute and regulate most biological processes. They carry out the transport and storage of small molecules and are involved in the transmission of biological signals. Enzymes, hormones, transcription factors, pumps and antibodies are examples of the diverse functions fulfilled by proteins in a living organism [Sasson, 2003].

Proteins are macromolecules, and consist of combinations of amino acids in peptide linkages, that contain Carbon, Hydrogen, Oxygen, Nitrogen, and Sulphur atoms. There are twenty different types of amino acids (each amino acid is labeled by a character) and two more characters {B, Z} (used for occurrence of specific amino acids) as shown in Table 2.1. These 22 (20+2) characters are combined to generate an infinite number of sequences. Only a small subset of all possible sequences appears in nature [Mount, 2001].

In the study of proteins, the three important attributes considered are sequence, structure, and function of a protein. Protein is a sequence of permutations of amino acids known as the primary structure (of the protein). The average length of a protein sequence is about 350 amino acids, but it can vary from as short as a few amino acids to as long as a few thousand (the longest being more than 5000 amino acids). Each amino acid is treated as a character, and proteins are viewed as long sequences of these characters.

Protein structure is the way the protein is outlaid in the 2-D or 3-D space. Secondary structures are local sequence elements. They have a well-determined regular shape, such as a helix or a strand, loops or coils (sequences do not have either helix or strand). Secondary structures when packed are called tertiary structures. Helix and strands form core of the protein structures. Loops or coils are found on the surface of the molecule and therefore the shape is called a fold.

The protein function is perhaps most important, yet most elusive. The protein function is its actual role in the specific organism in which it exists. Understanding the protein function is critical for most applications, such as drug design, genetic engineering, or pure biological research. The three dimensional structure of a protein gives most of the information about its biological function. However, determining the three dimensional structure of a protein is difficult and not always feasible.

The advent of advanced techniques for sequencing proteins in the last two decades, spurred the explosive growth witnessed today in protein databases. Due to this rate of growth, the biological function of a large fraction (between one third and half, depending on the organism) of sequenced proteins remains unknown.

Table 2.1: The Single- and Three- Letter Amino Acid Codes

[IUPAC, 1969] [IUPAC-IUB, 1983]

Amino Acid	Single letter Code	Code	Chemical Properties
Alanine	A	Ala	Hydrophobic Aliphatic
Cysteine	C	Cys	Disulphide bond former
Aspartic acid	D	Asp	Acidic
Glutamic acid	E	Glu	Acidic
Phenylalanine	F	Phe	Hydrophobic Aromatic
Glycine	G	Gly	Special structural properties
Histidine	H	His	Basic
Isoleucine	I	Ile	Hydrophobic Aliphatic
Lysine	K	Lys	Basic
Leucine	L	Leu	Hydrophobic Aliphatic
Methoionine	M	Met	Hydrophobic Aliphatic
Asparagine	N	Asn	Polar neutral
Proline	P	Pro	Special structural properties
Glutamine	Q	Gln	Polar neutral
Arginine	R	Arg	Basic
Serine	S	Ser	Polar neutral
Threonine	T	Thr	Polar neutral
Valine	V	Val	Hydrophobic Aliphatic
Tryptophan	W	Trp	Hydrophobic Aromatic
Tyrosine	Y	Tyr	Hydrophobic Aromatic
Asparagine/Aspartate	B	Asx	
Glutamine/Glutamate	Z	Glx	

The function of many proteins is defined by the context in terms of protein pattern and localization. Thus difficulty arises in assigning a certain function to a particular protein. In addition, a protein string may be subjected to large number of modifications that may affect its function and fate. The complexity of protein function prediction uses database searches, to find proteins similar to a new protein, thus inferring the protein function. Protein clustering or classification, where databases of

proteins are organized into groups or families in a manner that attempt to capture protein similarity, generalizes the method.

The focus is on available systems of protein clustering and classification. Such systems use the protein sequence, and at times structure, to classify proteins into families. The classification may be leveraged towards function inference.

2.2 PROTEIN CLASSIFICATION ALGORITHMS

The most commonly used algorithms for sequence similarity, and the manner in which they can be used directly for protein classification is discussed. The following subsections describe classification systems based on the methodology used: Motif-based classifications, Full sequence analysis classifications, Phylogenetic classifications, Structure based classifications and Aggregated classifications. The primary focus is on widely available software systems for protein classification.

2.2.1 Sequence Similarity

One of the most common approaches to classify proteins is using sequence similarity. Sequence similarity is a well-studied subject, and numerous software packages suited for biological sequences are available. Such packages (e.g., BLAST) are probably the most widely used software in the fields of biology and bioinformatics.

Sequence similarity algorithms take two sequences as input and provide a measure of distance or similarity between them: higher the distance, lower is the similarity, and vice versa. Levenshtein [IUPAC-IUB, 1983] used dynamic programming algorithm for determining the distance between sequences, referred to as ‘edit distance’. The ‘edit distance’ between two sequences is defined as the number of insertions, deletions, and replacements of characters from the first sequence to obtain the second sequence. Some variants of the edit distance allow for reversals of subsequences. The edit distance problem is strongly related to the problem of sequence alignment. The problems are essentially equivalent, as the alignment can be easily produced from a set of insertions and deletions of characters.

Similarity and alignment of biological sequences were studied by Needleman and Wunsch [Needleman, 1970]. The Needleman-Wunsch sequence similarity and

sequence alignment are usually referred to as *global* sequence alignment. In other words, this is an alignment of full-length sequences. In practice, exceeding similar sequences only can be globally aligned satisfactorily. However, many proteins exhibit strong *local* similarity. Smith and Waterman studied local alignment problem [Smith, 1981].

Algorithms for calculating either local or global similarity for protein sequences do not confer equal weight to all amino acids. Instead, scoring matrices are used to achieve an alphabet-weighted similarity. An alphabet-weighted 'edit distance' can also be defined using such scoring matrices, giving different weights to replacement of different characters. The most commonly used scoring matrices are BLOSUM [Henikoff, 1992], and the earlier PAM [Dayhoff, 1978]. The most popular algorithm used for global and local similarity calculation is BLAST, with a variant of it called PSI-BLAST. Other algorithms are FASTA and Smith-Waterman's algorithm.

2.2.1.1 Smith-Waterman

The Smith-Waterman method searches for local alignment [Smith, 1981]. In other words, instead of looking at the entire length of each sequence, it compares subsequences of all possible lengths. The Smith-Waterman algorithm is based on dynamic programming. This is an algorithmic technique where problems are solved by caching the solutions of sub-problems and are used in later stages of the computation. In the case of sequence alignment, the concept is to calculate the best local alignment score at a certain location along the sequence based on the best scores in the previous locations.

The alignment scores are based on the notion of 'edit distance', i.e., counting the number of transformations, one sequence requires obtaining from the second sequence. Transformations include substituting one character for another, insertion of a string of characters, or deletion of string of characters. The actual score is calculated using score matrices, which associate a weight with each pair of characters. The algorithm uses two penalties, one for opening gaps and another for extending them [Mount, 2001].

2.2.1.2 FASTA

The Smith-Waterman algorithm provides a good measure of local alignments, at a penalty cost. In a naïve implementation, its running time is cubic in the lengths of the sequences compared. As sequence database grew, the need for a more efficient comparison method emerged. FASTA [Pearson, 1990] is a heuristic method, which provides an approximation of the local alignment score. It is based on the observation that good local alignments typically stem from identities in sequences. The FASTA algorithm constructs a lookup table, which stores all instances, *k*-tuples of amino acids appearing in the sequence. The typical value of *k* used is k=2, which means the lookup table stores all instances of pairs of proteins. Using this lookup table, the best regions with the highest density of identities are identified. These identities, viewed on full dynamic programming table, can be considered as k-length diagonals. With these diagonals at hand, FASTA searches for the best 'diagonal runs', which are sequences of consecutive identities on a single diagonal.

2.2.1.3 BLAST

BLAST [Altschul, 1997] is another heuristic method for local alignment. Instead of looking for identical k-tuples in sequences, it looks for *similar* k-tuples. It looks for k-tuples (the typical value used for k in protein comparison is k=3) in one sequence that scores atleast 'T' when aligned with the other sequence, again using a scoring matrix. Such local similarities are extended in both directions in an attempt to find locally optimal ungapped (i.e., continuous) alignments, with a score of atleast 'S'. These alignments are called High Scoring Pairs (HSPs), and are combined to provide the best local alignment of the two strings. The neighborhood threshold 'T' and the score threshold 'S' are adjustable parameters. BLAST is even faster than FASTA, as it does not use dynamic programming. It is considered to be as sensitive (and thus as accurate) as FASTA, and for this reason it is currently the most popular sequence search and comparison tool, both for amino acid and nucleic acid sequences. BLAST outputs the similarity score, sequence alignments, and the statistical significance of the similarity, referred to as E-score. The latter provides an estimate of the probability of having similarity of this quality with a random string.

2.2.1.4 PSI-BLAST

PSI-BLAST [Altschul, 1997] is a variant of BLAST, which performs database search in an iterative fashion. Given a query sequence and a database of sequences, BLAST is used to find the sequences in the database that are most similar to the query sequence. A multiple alignment is constructed for these sequences, based on which a profile is generated. The profile is a position specific scoring matrix, which holds the probability of having each AA in each one of the positions in the sequence. This profile is now compared to the protein database, seeking for local alignments, using an algorithm similar to BLAST. A possibly new set of sequences in the database is found which match the profile. This process can be repeated for this set rapidly until convergence occurs (i.e., the BLAST result is identical to the set of sequences from which the profile was constructed) or for a fixed number of iterations. PSI-BLAST is considered the program of choice for detecting remote homologues. Yet, the exhaustive iteration scheme often results in considering non-related sequences among the correct hit list.

2.2.2 Classification with Sequence Similarity

Several inherent challenges exist in classification of proteins that are based on their sequence similarities. The current protein databases combine proteins having different evolutionary history. For example, the sequences of many proteins that were evolved from a common ancestor were already diverged beyond detection by any of the search programs like BLAST, PSI-BLAST, and FASTA. Other proteins may still exhibit extremely high degree of conservation in their sequences despite very long evolutionary distances. Thus, it is evident that for any sequence based classification, the following operations are fundamental: (i) varying the distance matrix selected, (ii) defining the penalty for opening and extending gaps in the alignments and (iii) choosing the preferred statistical and computational parameters.

The sequence similarity measures such as BLAST and Smith-Waterman [Smith, 1981] are used as the building blocks for some classification systems. Such measures can be used directly for classification, using the well-known "Nearest-Neighbor" paradigm for supervised learning. In other words, a new protein is associated with the protein nearest to it in the database of proteins with known function.

Similarity based classification is done by searching for all the proteins akin to a given protein upto a specified threshold. One caveat of such classification is the choice of threshold. If the threshold chosen is too low, it is possible that no matches are found. Alternatively, if the threshold chosen is high, it is likely that a lot of non-related sequences, or 'false positives' shall be retrieved. This problem frequently surfaces while classifying the sequences, which have diverged during evolution. Consequently similarity is difficult to detect, at least without having a large fraction of false positives.

2.2.3 Classification based on Domain and Motif Analysis

Although sequence comparison is the most widely used tool for classifying proteins, it is not sufficient in all cases. An alternative approach is based on the notion that domains form the building blocks of proteins and not amino acids. Based on this notion, proteins with similar domains are associated with each other, inspite of having low similarity scores. A variety of classification systems were developed which are based on domain and motif analysis. Such systems use multiple sequence alignments to detect conserved regions in sequences. The differentiating factor between the diverse classifications based on domain and motif analysis is the underlying computational representation of a motif or domain.

2.2.3.1 PROSITE

PROSITE [Falquet, 2002] is the oldest motif-based classification of proteins. The goal set out by PROSITE is to identify and represent all biologically significant signatures or fingerprints. Signatures are described either as *regular expressions* or as profiles. Regular expressions in PROSITE are described using the following rules:

1. Standard one-letter codes for amino acids are used.
2. The symbol '*' is used as a wildcard, in a position where any amino acid is accepted.
3. At positions where one of several amino acids may be used, square parentheses are used. For example [AT] stands for 'A' or 'T'.
4. At positions where most amino acids can be used, curly brackets are used. For example {AT} stands for any amino acid other than 'A' or 'T'.
5. Elements in a pattern are separated by a hyphen ('-').

6. Number in parenthesis indicates the repetition of an element. For example x (3) means xxx and x (2,3) means xx or xxx.
7. '<' and '>' indicate when a pattern is restricted to either the N- or C-terminal of a sequence respectively.
8. Patterns end with a dot ('.').

PROSITE is essentially a manually maintained database. This allows each entry to be associated with all relevant literature references. In fact, many of the entries are generated via published multiple alignments. In addition, cross-references to PDB [Berman, 2000] are provided when applicable. PDB is a database of proteins for which three-dimensional structures are known. Regular expressions are not suitable for classifying families whose members are highly diverged. To this end, PROSITE was extended to include profiles, thus extending its coverage. While PROSITE provides a dictionary of motifs and domains, it has several drawbacks as a classification system. Two of the problems are:

(i) Missing patterns: As patterns are detected mostly manually, not all patterns existing in real world proteins have been detected.

(ii) Low information patterns: Due to the fact that pattern lengths vary from a few amino acids to thousands of amino acids, some patterns have lower value in classifying a protein.

2.2.3.2 BLOCKS

BLOCKS [Henikoff, 2000] are a database of highly conserved protein regions. A 'block' is defined as a contiguous segment corresponding to the most conserved regions of proteins. Such blocks are automatically detected. In contrast to PROSITE, blocks are not associated with function or known literature, as the process is completely automated. Blocks are derived by performing multiple alignments of protein families as defined in InterPro, and also by searching for segments with a high number of identities.

It is important to note that the PROSITE pattern is not used to build BLOCKS database. BLOCKS entry may or may not contain the PROSITE pattern corresponding to the InterPro group from which the entry was derived.

An important application of BLOCKS is generation of amino acid substitution matrices. Such matrices are used for the sequence similarity algorithm (Smith-Waterman, FASTA, and BLAST). The BLOSUM [Henikoff, 1992] matrices are derived from the BLOCKS database, and currently the substitution matrices are most widely used for sequence comparison.

2.2.3.3 PRINTS

A PRINTS [Attwood, 2002] is a database of domain family fingerprints. A fingerprint is defined as a group of conserved motifs used to characterize a protein family. Fingerprints are more powerful constructs than single motifs, in the sense that they can more effectively narrow down the families. The fingerprinting method used in PRINTS relies on performing an iterative process of multiple sequence alignments. The process starts with only a small number of proteins. Once a set of conserved regions forming a fingerprint is identified, the full database is scanned to find matching proteins. Larger sets of proteins are then analyzed to generate more motifs. The whole process is repeated with the probable larger set until convergence. PRINTS share the main drawback of other domain-based systems due to the lack of full coverage of the protein space. Another limitation is that fingerprints are at times too restrictive. Thus it is possible that a new protein will merely match a portion (several motifs) in a fingerprint. In such cases, the protein might belong to a subfamily. However, there is no clear-cut classification for it.

2.2.3.4 PFAM

PFAM [Bateman, 2000] is a database of protein alignments and Hidden Markov Model. A Hidden Markov Model is an abstraction used to statistically describe the consensus sequence for a protein. Such a model consists of a sequence of nodes, with a designated begin and end states. Each node in a Hidden Markov Model has three ‘invisible’ states associated with it. These are a Match state (M), Insert state (I), and Delete state (D). Each has a transition probability associated with it. This probability is node specific, and hence is position specific in terms of the sequence. The match state has a probability of matching a particular amino acid. This probability is referred to as ‘emitting’ probability. Similarly, in the insert state there is a probability associated with each amino acid. The probability of no amino acid associated with a node is captured by the probability of transitioning into the delete

state. Hidden Markov Model can be automatically generated from multiple alignments. One of the most popular software packages for generating Hidden Markov Model, called HMMER [Sonnhammer, 1998] is used in PFAM.

Given a new sequence, it is possible to evaluate the probability of that sequence belonging to the family modeled by certain Hidden Markov Model. A similarity score is associated with a new sequence based on the most probable path through the Hidden Markov Model, which generates the input sequence. The advantage of Hidden Markov Model is by the virtue of the fact that they implement position specific scoring, in contrast to ordinary sequence similarity measures. This allows more accurate modeling of families, with less chance of similarities occurring. The disadvantage of Hidden Markov Model (compared to regular expressions for example) is that they are relatively large and thus difficult to understand. Another caveat is that it is by and large difficult to detect subfamilies.

In biological sequence analysis, Hidden Markov Models are built based on a multiple alignment as shown in Figure 2.1. In general, the multiple alignments are generated from a training set consisting of positive examples of protein sequences that belong to a certain functional family sharing a level of sequence similarities.

When considering two dice (loaded die and the fair die) as two models (A); each of them has its own Markov chain with transition probabilities between different numbers in a die. While considering the two dice as two states of a Hidden Markov Model (B); it has transition probabilities between the both states (dice) and emission probabilities within each state.

As shown in Figure 2.1, a Hidden Markov Model, which can be visualized as a finite state machine, has a start and an end state in addition to the previously identified match, insert, and delete states. Each of these states has position specific transition probabilities for transitioning into each of these states from the previous state (represented by arrows as shown in Figure 2.1). Match states have position-specific emission probabilities for each of the twenty amino acids. Insert states also have position-specific emission probabilities for inserting each of the twenty amino acids at that state. When no residue is associated with a node, it is a delete state, and no emission probability is associated with it. To obtain the probability that a new

sequence belongs to the family of the model, the new sequence is compared to the Hidden Markov Model by aligning it to the model. The most probable path taken to generate the sequence similar to the new sequence gives the similarity score. It is calculated by multiplying the emission and transition probabilities along the path. The most likely path through the model is computed with the *Viterbi* algorithm or the *forward* algorithm [Durbin, 1998]. One could also generate the most probable sequence obtained from a particular HMM by summing over all possible paths and choosing the path with the maximum score. In both ways, the most probable path can be efficiently and optimally calculated.

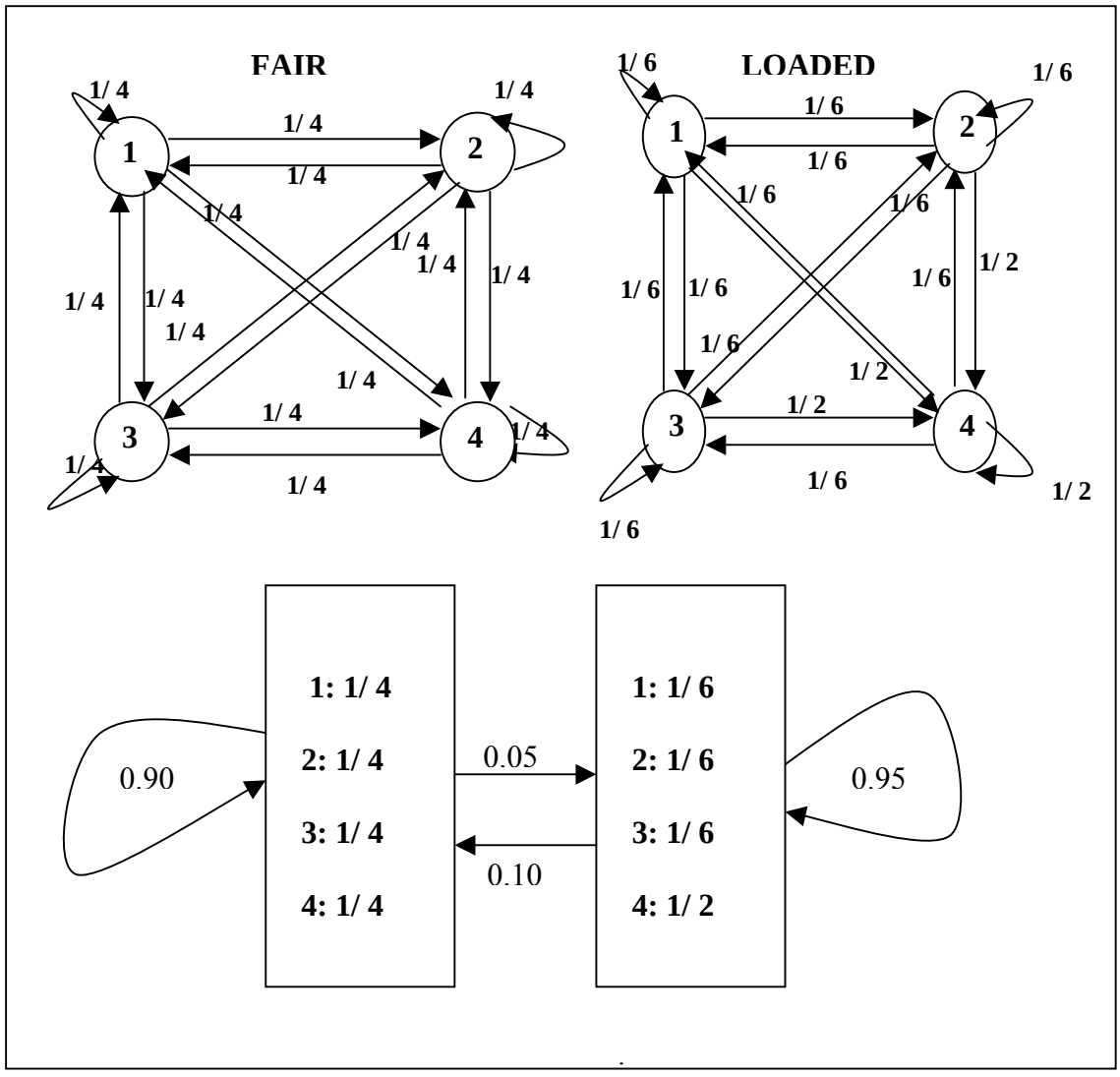

Figure 2.1: Model of a Markov chain and a Hidden Markov Model

2.2.3.5 Support Vector Machines (SVMs)

A Support Vector Machine (SVM) is a learning machine that makes a binary classification based on a separating hyper plane on a remapped instance space [Christianini, 2000]. The goal of the classification is to remap the input vectors onto a multidimensional space so that the instances are linearly separable. SVMs learn from labeled examples from a training set including both positive and negative samples. Depending upon a set of attributes, SVMs find a hyper plane that classifies the positive and negative data in the training set. The hyper plane is optimized in such a way that the distance called the *margin*, between the hyper plane and the closest training example is maximized. The data points nearest to the margin on both sides are called *support vectors*. The assumption is that there is a mapping or target function between the data and their labels the machine will learn [Karchin, 2002]. A kernel function, which is a dot product that is used in remapping input feature vectors, is used to find the hyper plane. Once the hyper plane is found, unlabeled examples from the test set can be classified. Classification can be made solely based upon the support vectors found.

A feature vector is used for representing each sequence (a collection of the attributes in a vector format). If the dimension of the feature vector is l (l attributes), a sequence 'x' can be represented by x = [x1, x2,.., xl]. In a two class problem, the label of the sequence can be either 1 or -1. Let us represent the label of the sequence x with $y_x = \{1,-1\}$. A classifier is then built using the feature vectors of the training set. A weight vector of the same dimensions as the feature vector is represented by w = *[w1, w2, . . .wl]*. The label of the sequence is then predicted as 1 if $w \cdot x > b$ (b is a threshold), else the label is -1.

The equation of the margin γ_x is given by

$$\gamma_x = y_x(w \cdot x + b)$$

If 'γ' is positive, then the sequence is correctly classified, otherwise, it is not correctly classified. Every time a sequence is incorrectly classified, the weight vector 'w' and the threshold 'b' are updated.

Some commonly used kernel function includes: linear, polynomial, radial basis, and sigmoid functions. The equations for the respective kernel functions are listed in Table 2.2.

Table 2.2: The four types of kernel functions frequently used with SVM

Linear Kernel	$K(x\ y) = (x \cdot y + 1)$
Polynomial Kernel	$K(x\ y) = (kx \cdot y + c)^p$
Sigmoid Kernel	$K(x\ y) = \tanh(kx \cdot y + c)$
Radial Basis Kernel	$K(x\ y) = e^{-\|x-y\|2}$

The advantage of using the SVM in this study is the ability to classify protein sequences without depending on multiple alignments. There are only a few studies using SVMs in the classification of protein sequences. Karchin [Karchin, 2002] developed the SVM-Fisher method. Liao and Noble [Liao, 2002] on the other hand used the SVM-pairwise method. Wang [Wang, 2004] and Zhang [Zhang, 2003] also experimented with SVM on identifying Thioredoxin proteins, another example of protein family with low primary sequence similarity.

2.2.4 Classification based on Full Protein Sequence

Domain-based classifications are limited in the sense that many proteins have several domain appearances, and that some proteins do not have any domain (or atleast not a recorded domain). In addition, for small families (e.g., with 2 members) it is not possible to define domains. A different approach to classification is offered by several systems, which classify proteins based on the full sequence. This is typically achieved by sequence comparison. The basic principle of most full sequence based classifications is that of homology transitivity. The concept is that homology (the relation between two proteins which are evolved from the same protein) is a transitive relation, whereas similarity is not necessarily a transitive relation. The main caveat of all such classifications is chance similarities, which result in misclassification, and multidomain proteins, which may be related to several families.

2.2.4.1 ProtoMap

ProtoMap [Yona, 2000a] provides a fully automated hierarchical clustering of the protein space. ProtoMap takes a graph-based approach, where the sequence space is represented by a directed graph in which vertices are protein sequences and edges represent similarity between proteins. The weight associated with an edge measures the similarity, or the significance of the relationship. ProtoMap uses a combination of

Smith-Waterman, FASTA, and BLAST to determine similarity. The ProtoMap algorithm constructs a partitioning of the protein database at different levels of granularity. Initially, the most significant relationships only are considered. The subgraph induced by all edges with high similarity (e.g., E-value of 1e-100 or less) is used, and each connected component is considered as a single cluster. These clusters are then iteratively merged in agglomerative hierarchical clustering, using an average link where the average is geometric mean of sequence similarity scores. Hierarchical clustering, as the name suggests, is a clustering technique which generates a hierarchy of clusters, wherein at each level clusters are generated by merging clusters of a lower level. Accordingly, each cluster is a single entity at the bottom level. There are two fundamental paradigms for hierarchical clustering: (a) agglomerative (bottom-up) (b) divisive (top-down). Agglomerative starts from the bottom, from single entities, and repeatedly merge pairs of clusters into larger clusters. At each step, the pair of clusters with the highest similarity is merged. In the divisive approach, the whole set of data items are considered, and recursively split into smaller clusters. In practice, while using hierarchical clustering for proteins, agglomerative clustering is usually used. ProtoMap uses average link clustering, which means that the similarity of two clusters is defined as the average of pairs where one element is taken from one cluster and another element from other cluster. In ProtoMap, once the hierarchical clustering procedure is completed, the threshold used is increased. The same process is repeated for 1e-95, 1e-90, etc., upto 1e0. Each time the threshold is increased, strongly connected clusters are merged and hierarchical clustering is applied yet again.

ProtoMap offers a web-based interface, which allows browsing the clusters both textually and graphically. ProtoMap also maintains SwissProt annotation and keywords, as well as links into Bio-Space. The cluster page details each member protein, along with its SwissProt keywords. ProtoMap provides individual protein pages linking each protein to all clusters containing it. Each cluster also has a summary page with some additional information such as PROSITE families and taxonomy. ProtoMap provides a graphical display of cluster relationships, as well as a tree like representation.

2.2.4.2 ProtoNet

ProtoNet [Sasson, 2003] provides a fully automated hierarchical clustering of the SwissProt proteins. It implements an average link hierarchical agglomerative clustering algorithm. The novelty of ProtoNet is the use of several averaging methods. The averaging methods provided are Arithmetic mean, Geometric mean and Harmonic mean.

ProtoNet offers a variety of methods to traverse through clusters and analyze their contents. The hierarchical clustering is fully traversable, thus providing users with varying granularity, based on individual requirements. The ProtoNet web based interface provides detailed information in the cluster level. The top portion of the cluster details demonstrates a graphical representation of the sequence of merges taking place for the creation of this cluster and subsequent clusters. Each protein recorded in the ProtoNet database is associated with detailed taxonomical representation as well as a variety of keywords (including InterPro and PROSITE entries). ProtoNet also provides motif and domain information taken from PFAM, PRINTS, Prosite, and SMART.

ProtoNet provides a unique feature by which users may classify their own proteins. In contrast to other systems, users may classify only a single sequence at a time. Whereas ProtoNet stores sequences classified by users may be specifically classify multiple sequences concurrently. ProtoNet provide statistical measures for the purity and the sensitivity for each merging in the process as compared with other type of classifications.

2.2.5 Phylogenetic Classification

COGS [Tatusov, 2001] provide a clustering of proteins derived from 43 complete genomes. COGS apply single linkage hierarchical agglomerative clustering to get clustering of orthologous proteins or orthologous sets of paralogs. The former refers to genes from different species, which have evolved from the same protein, and the later to genes from the same genome, which are related by duplication. Each cluster consists of atleast three species. The clustering process starts by forming a minimal COG with three elements, and then proceeds by merging COGS sharing an edge. The COGS clustering algorithm has the capability of splitting COGS, which are incorrectly merged.

2.2.6 Classification based on Protein Structure

Full sequence analysis helps in classifying proteins with known sequences. However, the common perception is that protein function stems from protein structure and not much from sequence. This gives rise to the idea of classifying proteins based on structure. The drawback of structure based clustering is that the number of sequences for which the structure is available (or the structure is 'solved') is relatively small, due to the complexity of obtaining high-resolution structural information.

Structure based clustering takes into account the 3-D structure of a protein. Several different algorithms and software packages are available for measuring the similarity between two protein structures.

2.2.6.1 SCOP

SCOP [Lo Conte, 2002] is a structural classification of proteins into a four-level hierarchy.

- Family – Proteins with significant sequence similarity, and with clear evolutionary relationship.
- Superfamily – Proteins with low sequence similarity, but with structural and functional features suggesting a common evolutionary origin.
- Fold – Superfamilies with major structural similarity.
- Class – High-level classification (e.g., All-alpha, All-beta, Alpha/Beta, Alpha+Beta, Membrane proteins, etc.)

SCOP is organized as a tree, and on top of all classes there is the SCOP "root".

2.2.6.2 CATH

CATH [Orengo, 1999] is a structural classification into four-level hierarchy:

- Homologous superfamily (H-level) – Sequences with high similarity. Several conditions are defined combining sequence and structure.
- Topology (T-level) – Structure comparison is used to group together protein structures into fold families.
- Architecture (A-level) – Structures are grouped based on the overall shape of the domain structures.

- Class – Structures are determined according to secondary structure composition, similar to SCOP classes.

The name of the system is based on the level names (Class, Architecture, Topology and Homology). The assignment of structures to families and topologies based on similarity is automatic.

2.2.6.3 FSSP

FSSP [Holm, 1996] provides fold classification using structure-structure alignment of proteins. The classification is based on an exhaustive all-against-all structure comparison using Dali [Holm, 1998] structure comparison. FSSP provides a web-based interface, which facilitates 3-D superimposition and multiple alignments of structures.

2.3 LIMITATIONS OF THE EXISTING ALGORITHMS

Multiple sequence alignment, profiles, motifs, short sequences, etc., [Hulo, 2004; Attwood, 2003; Bateman 2003] are utilized in the existing classification algorithms. The amino acid composition of the protein has not been explored for diverged series [Ramadevi, 2004a]. The sequence consists of local and global information. Frequency occurrence, profiles of amino acids as a global characteristic and based on neighborhood as local information has not been considered.

In this thesis, some of problem stated above are addressed, focusing mainly on search space reduction. The proposed methodology in the thesis is hybridization of Set theory, Relations, Rough Sets, Decision Tree, and Concept Lattice and binarization processes. The following sections briefly gives foundations of the topics starting with set theory for completeness.

2.4 SET THEORY

A set is a collection of well-defined objects, called the elements of the set. The elements of the set are said to belong to the set. Sets are represented by capital letters. Lower case letters represents the elements. The universal set is represented as '∑', it consists of all the symbols/elements under consideration.

Set ‘A’ contains elements {1,2,3,4,5} is written as A = {1,2,3,4,5}, ‘5’ ∈ A indicates that element ‘5’ belongs to Set A. The order of the elements in the set is invariant, i.e., {1,2,3,4,5} is equivalent to {3,2,1,5,4}.

$\sum$ = {1,2,3,4,5,6,7,8,9,10}

A = {1,2,3,4,5}

B = {2,3,5}

2.4.1 Set Operations

The different set operations are given in Table 2.3.

Table 2.3: Set Operations

Set operation	Symbol	Remarks
Set B is a subset of set A	$B \subset A$	Set B elements are contained in Set A
The absolute complement of set A	$\overline{A}$	$\sum$ - A = {6,7,8,9,10}
The relative complement of set B with respect to set A	B - A	B – A = { }
The union of sets A and B	$A \cup B$	$A \cup B$ = {1,2,3,4,5}
The intersection of sets A and B	$A \cap B$	$A \cap B$ = {2,3,5}
The symmetrical difference of sets A and B	$A \Delta B$	$A \Delta B = (A-B) \cup (B-A)$ = {1,5}$\cup$ { } = {1,5}

The concept of Set theory is utilized in extracting information in Sequence Arithmetic module (Chapter III). A brief of Rough Set Theory is elucidated in the following section which useful in finding the Reduct (Chapter IV).

2.5 ROUGH SETS

The theory of Rough Sets provided by Pawlak is an important mathematical tool for computer technology. It is used in several decision-making problems like data mining, knowledge representation, knowledge acquisition and many real time applications [Ziarko W; Elsayed Radwan, 2004]. In 1981, Pawlak [Pawlak, 1981] laid the theoretical foundation for information systems. Later, this approach helped him in adopting theory of Rough Set [Pawlak, 1982], he derived rough dependency of attributes in information systems [Pawlak, 1985].

Subsequently, in 1989, Wiweger [Wiweger, 1989] viewed the theory of rough sets using topological spaces. In 1989, Wybraniec generalized the approximation space [Wybraniec, 1989]. Yao discussed the algebraic approach [Yao, 1996a] on rough sets and later generalized the theory of rough sets [Yao, 1996b; Yao, 1998a; Yao, 1998b]. Later the concepts were adapted to information system [Yao, 2001].

Selecting an efficient set of attributes (features) is a basic problem, for many practical applications of the rough sets [Mohua, 2006]. These attributes are necessary for the classification of objects in the universe considered. These knowledge reduction problems are highly involved in information systems Buszkowski [Buszkowski, 1986; Biswas, 2004]. In general, any information system consists of several attributes. In the process, it is tedious to recall each attribute every time. Therefore, it is necessary to avoid the redundant attributes as well as to pick up the minimal feature. This minimal feature is called a reduct, which can be computed using Rough Set. Discernibility matrices are used for describing the method of computing reducts [Skowron, 1991]. However, this method cannot list all the possible reducts of the information system. In order to obviate this, Starzyk [Starzyk, 1999] gave an algorithm to list all reducts of the given information system. In [Rao, 1999; Hari, 2006] the predominant attributes were found using 'val theory', which were equivalent to reducts.

A brief introduction to Rough Set is given in the following section. One can refer to applications involving Rough Set Theory in documents of Slezak [Slezak, 2008] and Bhanuprasad [Bhanuprasad, 2007].

2.5.1 Knowledge Base

In Rough Set theory, a decision table is denoted by T = (U, A, C, D), where U is universe of discourse, A is a set of primitive features, and C, D $\subset$ A are the two subsets of features that are called condition and decision features, respectively.

2.5.2 Indiscernibility Relation

Let a $\in$ A, P $\subseteq$ A. A binary relation IND (P), called the Indiscernibility relation, is defined as follows:

$$IND(P) = \{(x, y) \in U \times U: \text{for all } a \in P, a(x) = a(y)\}$$

Let U/ IND (P) denote the family of all equivalence classes of the relation IND(P). For simplicity, notation U/P is used instead of U/ IND (P).

Consider 'U' and the equivalence relation 'R' on 'U'. The equivalence classes of 'R' are referred as *categories* or *concepts* of 'R' or *granules* and $[x]_R$ denotes a category in 'R' containing an element $x \in U$.

Equivalence classes U/IND(C) and U/IND(D) are called condition and decision classes, respectively.

2.5.3 Lower Approximation

Let $R \subseteq C$ and $X \subseteq U$. The R-lower approximation set of X can be presented formally as $\underline{R}\, X = \cup\{Y \in U/R: Y \subseteq X\}$

2.5.4 Upper Approximation

Let $R \subseteq C$ and $X \subseteq U$. The R-upper approximation set of X is represented as $\overline{R}\, X = \cup\{Y \in U/R: Y \cap X \neq \Phi\}$

R-rough set is given by $(\underline{R}X, \overline{R}\, X)$. If X is R-definable then $\underline{R}X = \overline{R}\, X$ otherwise X is R-Rough.

2.5.5 Boundary

The boundary $BN_R(X)$ is defined as $BN_R(X) = \overline{R}\, X - \underline{R}X$. Hence, if X is R-definable, then $BN_R(X) = \Phi$.

2.5.6 Positive Region and Negative Region

$\underline{R}X$ is also called positive region. The negative region of R is written as $NEG_R(X)$.

$POS_R(X) = \underline{R}X$,

$BN_R(X) = \overline{R}\, X - \underline{R}X$,

$NEG_R(X) = U - \overline{R}\, X$.

Example 2.1:

Consider the universe of discourse U = {a, b, c, d, e, f} and R be any equivalence relation in IND (K) (where K is knowledge base consisting of U and R) which partitions U into {{a, b, d}, {c, f}, {e}}. Then for any subset X = {a, b, c, d} of

U, $\underline{R}X$ = {a, b, d} and $\overline{R}X$ = {a, b, c, d, f}. Hence, $BN_R(X)$ = {c, f}. Hence, $POS_R(X)$ is {a, b, d} and the $NEG_R(X)$ is {e}.

On the other hand, consider a subset Y = {c, e, f}. Here, $\underline{R}Y$={c, e, f} and $\overline{R}Y$ ={c, e, f}. Therefore, $BN_R(Y) = \Phi$. Hence, Y is said to be R-definable.

Example 2.2:

Table 2.4: Sunburn Dataset [Wroblewski, 1995]

ID	Hair	Height	Weight	Lotion	Sunburn
X1	Blonde	Average	Light	No	Yes
X2	Blonde	Tall	Average	Yes	No
X3	Brown	Short	Average	Yes	No
X4	Blonde	Short	Average	No	Yes
X5	Red	Average	Heavy	No	Yes
X6	Brown	Tall	Heavy	No	No
X7	Brown	Average	Heavy	No	No
X8	Blonde	Short	Light	Yes	No

Using Table 2.4, the concepts of the information system are described as

U = {X1, X2, X3, X4, X5, X6, X7, X8}

A = {Hair, Height, Weight, Lotion, Sunburn}

For R = {Lotion} $\subseteq$ A, U/IND(R) = {{X2, X3, X8}, {X1, X4, X5, X6, X7}}

Let X = {X1, X4, X5}. Then with reference to R = {Lotion} for objects with decision attribute Sunburn = yes, the lower and upper approximations are $\underline{R}X = \Phi$, and $\overline{R}X$ = {X1, X4, X5, X6, X7}, respectively.

2.5.7 Dispensable and Indispensable Features

Let c $\in$ C.

Feature '**c**' is dispensable in '**T**' (refer Section 2.5.1), if $POS_{(C-(c))}(D) = POS_C(D)$.

If $POS_{(C-(c))}(D) \neq POS_C(D)$ feature '**c**' is indispensable in 'T'. 'C' is independent if all c $\in$ C are indispensable.

2.5.8 Reduct and CORE

A set of features $R \subseteq C$ is called a reduct of C, if $T^1 = \{U, A, R, D\}$ is independent and POS_R (D). In other words, a reduct is the minimal feature subset preserving the above condition.

CORE (C) denotes the set of all features indispensable in C. We have

CORE (C) = $\cap$ RED(C), where RED(C) is the set of all reducts of C.

2.5.9 Search for Indispensable Features

All indispensable features should be contained in an optimal feature subset, because removing any one or more of them cause inconsistency in a decision table.

CORE is the set of all indispensable features. Hence, the process of searching indispensable features is that of finding CORE.

The discernibility matrix can be used for CORE searching. The discernibility matrix can be briefly presented as follows:

Let T =(U, A, C, D) be a decision table, with U = $\{x_1, x_2, x_3, . . , x_n\}$. Discernibility matrix of T denoted $M(T)_{n \times n}$ is defined as:

$m_{ij} = \{a \in C: a(x_i) \neq a(x_j) \wedge (\exists d \in D, d(x_i) \neq d(x_j))\}$, for i , j = 1,2,3,..,n

Thus entry m_{ij} is the set of all attributes that classify objects x_i and x_j into different decision classes in U/D.

The CORE can be defined now as the set of all single element entries of the discernibility matrix, that is,

CORE(C) = $\{a \in C: m_{ij} = \{a\}$ for some i, j$\}\}$

Example 2.3:

The discernibility matrix corresponding to the sample database in Table 2.5 with U =$\{x_1, x_2,..,x_7\}$ C =$\{a, b, c, d\}$, D = $\{E\}$ is shown in Table 2.6.

A feature selection method using rough set theory can be regarded as finding a reduct R with respect to the best classification. Thus R is used instead of C in a rule discovery algorithm.

Table 2.5: A sample database

	a	b	c	d	E
x1	1	0	2	1	1
x2	1	0	2	0	1
x3	1	2	0	0	2
x4	1	2	2	1	0
x5	2	1	0	0	2
x6	2	1	1	0	2
x7	2	1	2	1	1

Table 2.6: Discernibility matrix for data in Table 2.5

	x1	x2	x3	x4	x5	x6
x2	-	-	-	-	-	-
x3	b,c,d	b,c	-	-	-	-
x4	b	b,d	c,d	-	-	-
x5	a,b,c,d	a,b,c	-	a,b,c,d	-	-
x6	a,b,c,d	a,b,c	-	a,b,c,d	-	-
x7	-	-	a,b,c,d	a,b	c,d	c,d

Product of Sums is obtained (POS) for the entries in the discernibility matrix and they are converted to Sum of Products (SOP) form, which gives the reduct.

POS = (bVcVd) Λ b Λ (aVbVcVd) Λ (aVbVcVd) Λ(bVc) Λ(bVd) Λ (aVbVc)Λ (aVbVc)Λ (cVd)Λ(aVbVcVd) Λ (aVbVcVd) Λ (aVbVcVd) Λ(aVb) Λ(cVd) Λ(cVd)

= (bVcVd) Λ b Λ (a V b V c V d) Λ (b V c) Λ(bVd) Λ (aVbVc) Λ (aVb) Λ(cVd) Λ(cVd)

Simplifying the above POS form into SOP form results in (b Λ c) V (b Λ d)

Reducts = {{b, c}, {b, d}}

CORE = {b, c} ∩ {b, d} = {b}

The Selection of an optimal reduct R from all subsets of features is not an easy task. Considering the combinations among N features, the number of possible reducts can be 2^N. Hence, selecting the optimal reduct is NP-hard. For this reason, various

methods for finding approximate results have been proposed [Pawlak, 1991; Liu, 1998; Yao, 1999]. The features in CORE must be included in an optimal result and in an approximate result. It is obvious that if the accuracy of a decision table is not changed, all indispensable features in CORE cannot be deleted from 'C'. The feature in CORE must be a member of feature subsets. It must be noted that not all the features in an optimal feature subset are indispensable.

A problem of feature subset selection is the process to select the features from dispensable features for forming the best reduct with CORE. We use CORE as an initial feature subset. If CORE is not a reduct, some of the dispensable features must be selected and added to it to make a reduct.

Heuristic algorithm searching for reduct approximation has been developed [ZHong, 2001]. At the outset, CORE is chosen as the initial attribute subset, and then an attribute from dispensable attributes are selected one by one using the above mentioned strategies by adding to the attribute subset, until a sufficient reduct approximation is achieved. The reduct is same as the reduct generated by classical approach.

The reduct is used in generating the decision tree; therefore a brief introduction to decision tree construction is given in the following section.

2.6 DECISION TREE

Decision Tree algorithms follow the basic principle known as Concept Learning System (CLS)[Hunt, 1961]. It tries to mimic the human process of learning, i.e., a concept starting with examples from two classes and based on some conditional attributes; rules are induced to distinguish them. A reduction in dimension of data reduces the size of the hypothesis-space and allows algorithms to operate faster and more effectively. Quinlan's ID3 algorithm for discrete attributes and its successor C4.5 for numeric continuous attributes are widely used algorithms for constructing decision trees [Quinlan, 1993]. Both algorithms attempt to select attributes barring the nominal attributes that have been used at the parent nodes. They are retained for further computation in the splitting criteria. Irrelevant attributes are not filtered until the example reaches the leaf of the decision tree, thus leading to extra overheads in terms of memory and computational efforts. Prior selection of attributes increases the

performance of the algorithm. Major problems posed in real life on using decision trees are – presence of correlated attributes in the dataset, over-fitting of induced decision tree, attribute selection error in the process of decision tree induction [Murthy, 1998].

2.6.1 Decision Tree Construction

A Decision Tree is typically constructed recursively in a top-down manner [Friedman, 1977; Quinlan, 1986]. If a set of labeled instances is sufficiently pure, then the tree is a leaf, with the assigned label being that of the most frequently occurring class in that set. Otherwise, a test is constructed and placed into an internal node that constitutes the tree so far. The test defines a partition of the instances according to the outcome of the test as applied to each instance. A branch is created for each block of the partition, and for each block, a decision tree is constructed recursively. A decision tree can be pruned, i.e., restricting the growth of the tree before it occurs. If the test at the node is done based on just one variable, it is called univariate test, otherwise it is multivariate test.

The best test at the internal node is selected based on heuristic function, which include Information gain, Gain ratio, GINI and Kolmogorov-Smirnoff distance. It is quite possible that a tree will overfit the data, therefore postpruning methods are available that reduce the size of the tree after it has been grown.

Traversing the tree from root to different leaf nodes will generate different decision rules. The path from root to each leaf is one decision rule. The decision tree and decision rules generated for the weather dataset (Table 2.7) are as in Figure 2.2.

Example 2.4:

Consider the Weather dataset

Table 2.7: Weather dataset

Id	Outlook	Temp	Humid	Wind	Play
X1	Sunny	Hot	High	Weak	No
X2	Sunny	Hot	High	Strong	No
X3	Overcast	Hot	High	Weak	Yes
X4	Rain	Mild	High	Weak	Yes
X5	Rain	Cool	Normal	Weak	Yes

X6	Rain	Cool	Normal	Strong	No
X7	Overcast	Cool	Normal	Strong	Yes
X8	Sunny	Mild	High	Weak	No
X9	Sunny	Cool	Normal	Weak	Yes
X10	Rain	Mild	Normal	Weak	Yes
X11	Sunny	Mild	Normal	Strong	Yes
X12	Overcast	Mild	High	Strong	Yes
X13	Overcast	Hot	Normal	Weak	Yes
X14	Rain	Mild	High	Strong	No

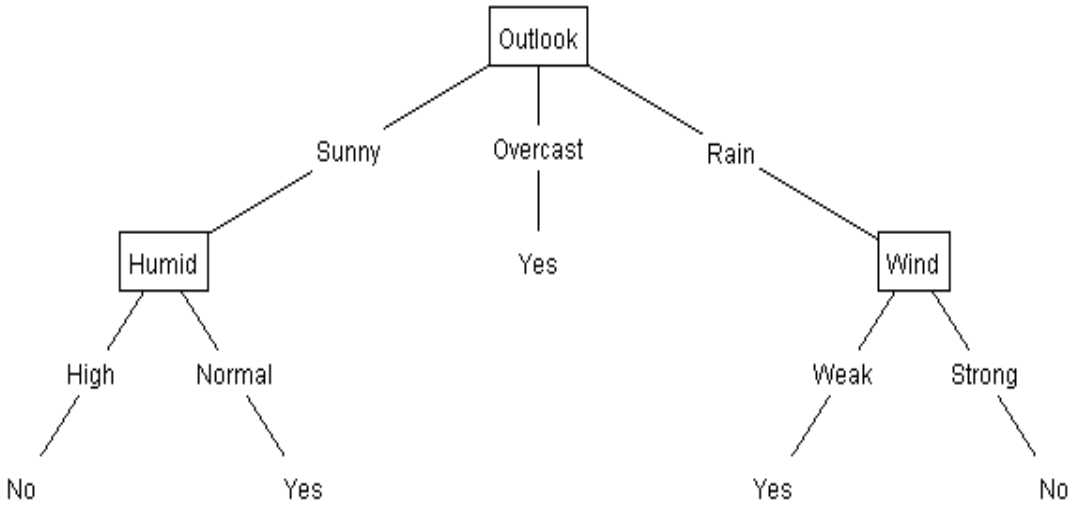

Figure 2.2: Decision Tree for Weather dataset

Rules Generated are of the form

1. If Outlook = Sunny and Humid = High then Play = No.
2. If Outlook = Sunny and Humid = Normal then Play = Yes.
3. If Outlook = Overcast then Play = Yes.
4. If Outlook = Rain and Wind = Weak then Play = Yes.
5. If Outlook = Rain and Wind = Strong then Play = No.

The relationship between the attributes and the instances in a classified decision table can be obtained by making use of concept lattice. Therefore, an introduction to concept lattice is given in the following section.

2.7 CONCEPT LATTICE

A concept lattice can be built from binary relation that has many applications. Each element in the lattice is a concept and corresponding graph (a Hasse diagram) is the generalization/specialization relationship between concepts and was proposed by Wille [Wille, 1982]. It represents a concept hierarchy, where each concept is a pair composed of an extension representing a subset of instances and an intension representing the common features for this set of instances.

Concept lattice can also be induced in an incremental manner, called incremental concept formation or simply concept formation [Fisher, 1991; Gennari, 1990] and in a big way; it is a fundamental process of human learning. Galois lattice is a concept lattice, and the construction of the lattice is shown in following subsection along with an illustration.

2.7.1 Galois Lattice

Given a set of instances E and a set of features E', and a binary relation R between these two sets (Table 2.8), there is a unique Galois lattice [Godin, 1995] corresponding to this relation (Figure 2.3). Each element of the lattice is a pair, (X, X'), composed of a set $X \in P(E)$ and $X' \in P(E')$ where P(A) is a power set of A.

A pair (X, X') from P(E) * P(E') is complete with respect to R if and only if the following properties are satisfied.

1. $X' = f(X)$ where $f(X) = \{x' \in E' \mid x \in X, x \, R \, x'\}$.
2. $X = f'(X')$ where $f'(X') = \{x \in E \mid x' \in X', x \, R \, x'\}$.

It may be noticed that $f(\Phi) = E'$ and $f'(\Phi) = E$. Only maximally extended pairs are kept in the hierarchy. If a subset X is not maximally extended, in the sense that there are other instances described by the features common to X, then the subset has to be extended to contain this instance. Symmetrically, if X' does not contain a feature that is common to every instance in X, it has to be extended to include this feature. The idea of maximally extending the sets is formalized by the mathematical notion of a closure in ordered sets.

A closure in an ordered set (E, =) is an application, h: E → E, satisfy the following properties:

i) $\forall\ x, \forall\ y,\ \ x = y \ \ \Rightarrow \ \ h(x) = h(y)$

ii) $\forall\ x,\ h(x) = x$

iii) $\forall\ x,\ h(h(x)) = h(x)$

Table 2.8: Binary matrix Representation

	E'(Attributes)									
E	R	a	b	c	d	e	f	g	h	i
	1	1	0	1	0	0	1	0	1	0
	2	1	0	1	0	0	0	1	0	1
	3	1	0	0	1	0	0	1	0	1
	4	0	1	1	0	0	1	0	1	0
	5	0	1	0	0	1	0	1	0	0

The image h(x) of x in E is called h-closure of x; if x = h(x), x is h-closed or a closed element for h. The applications h = fof ' and h' = f 'of are respectively closures in E and E'. Therefore the h-closed elements of E correspond to the maximally extended instance subsets and they form a lattice H isomorphic to the Galois lattice. The isomorphism is simply to put into correspondence the closed elements with the X sets of G. Symmetrically, the h'-closed elements of E ' are the maximally extended feature subsets and they correspond to the X ' sets of the G.

The couple of functions (f, f ') is a Galois connection between P(E) and P(E') and the Galois lattice G for the binary relation is the set of all complete pairs [Barbut, 1970] with the partial order

given C1 = (X1, X1') and C2 = (X2, X2 '), $C1 < C2 \Leftrightarrow X1' \subset X2'$.

There is dual relationship between the X and X' sets in the lattice, i.e.,

$$X1' \subset X2' \ \Leftrightarrow \ X2 \subset X1$$

and therefore, $C1 < C2 \ \Leftrightarrow X2 \subset X1$

The partial order is used to generate the lattice graph in the following way: There is an edge (C1, C2) if C1< C2 and there is no element C3 in the lattice such that C1 < C3 < C2. C1 is called the parent of C2 and C2 is child of C1. The graph is a Hasse diagram, where the edge direction is shown as either downward or upward, representing the generalization / specialization relationship between the concepts. It

corresponds to the subset relationship between the extension or intension of the concepts.

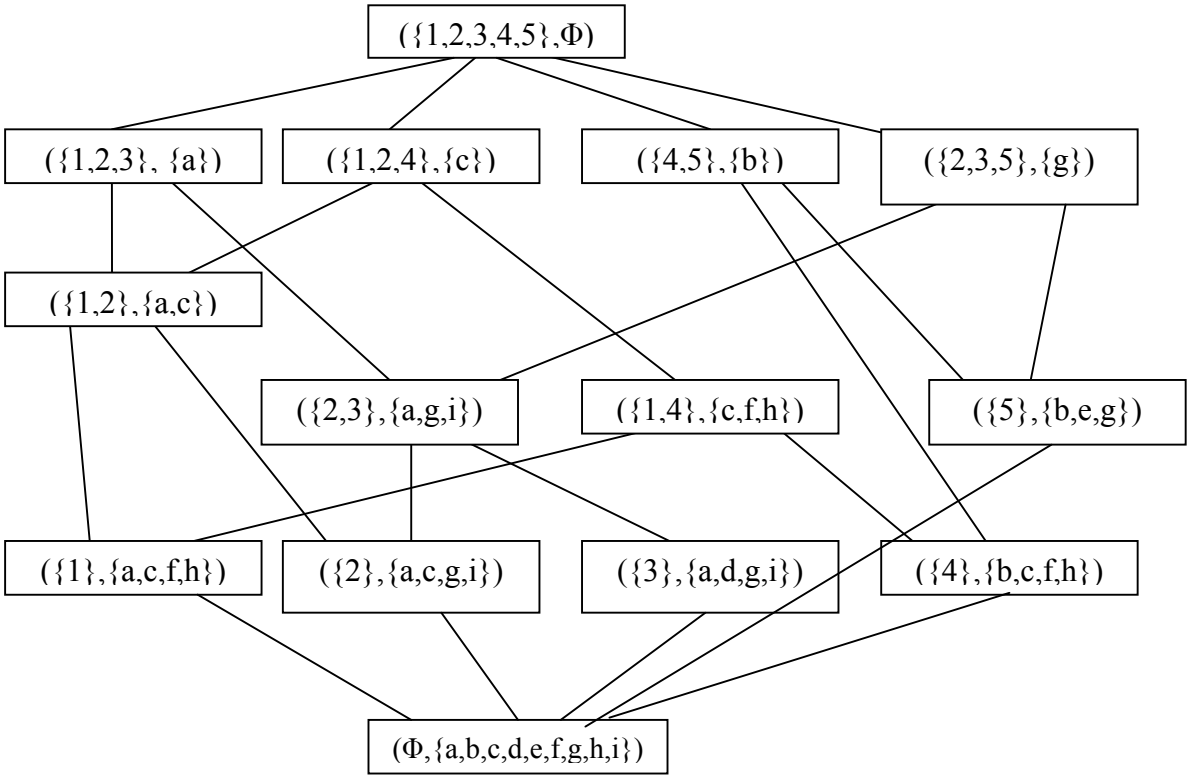

Figure 2.3: Galois Lattice for the binary data of Table 2.8

Given, C, a set of elements from G, inf(C) and sup(C) will denote the greatest lower bound and least upper bound of the elements in C respectively. A fundamental property of G is that it is a complete lattice since for any subset of G there exists a unique greatest lower bound and a unique least upper bound [Davey, 1992]. The construction of the lattice is incremental so that new instances can be added or deleted.

Each node in the Figure 2.3 shows the relationship between the objects and attributes, for example, a node with information ({1,2},{a,c}) indicates that object 1 and object 2 have attributes {a,c} with the same value. The node with value ({1,2,3,4,5}, Φ) depicts that objects 1,2,3,4 and 5 do not have any attributes with same value.

Different types of data continuous or discrete are discretized into binary format using different discretization algorithms [Han, 2001; Son, 1997; Witten, 2000;

Hoa, 1996; Ohrn, 1999]. The binary relation matrix that is used for the construction of the lattice is obtained from the spatial analysis of the sequences, which is discussed with illustrations in Chapter V.

2.8 PROTEIN DATASET

G-Protein Coupled Receptors and Myoglobin proteins are used for experimentation.

2.8.1 G-Protein Coupled Receptors (GPCR)

G-Protein Coupled Receptors are a superfamily of cell membrane proteins. They are characterized by seven water-insoluble (hydrophobic) regions believed to represent those that pass through the cell membrane, or transmembrane regions, as shown in Figure 2.4. Each GPCR has an amino terminal (NH_2 or N-terminal) region outside of the cell (extra cellular), followed by three sets of alternate intracellular (inside of the cell) and extra cellular loops, which connect the seven transmembrane regions, and a final intracellular carboxyl terminal (COOH- or C-terminal) region [Liao, 2002].

GPCRs are involved in signal transmission from outside to the interior of the cell through interaction with heterotrimeric G-proteins, or proteins that bind to guanine (G) nucleotides. The receptor is activated when a ligand that carries an environmental signal binds to a part of its cell surface component. A wide range of molecules is used as the ligands including peptide hormones, neurotransmitters, pancrine mediators, etc., which may be in many forms: e.g., ions, amino acids, lipid messenger's proteases.

The heterotrimeric G-proteins have three subunits, namely, 'α', 'β' and 'γ'. The G-protein activity is regulated by the 'α' subunit, which binds guanine (G) nucleotides. In an inactive state, 'α' is bound to a GDP (guanine diphosphate), which together are bound to subunits 'β' and 'γ'. A ligand binding at the extra cellular domain of the receptor induces a conformational change in the receptor, which causes the G-proteins to bind to the intracellular domain of the receptor. This stimulates the exchange of the GDP with a GTP (guanine triphosphate) in the binding site of the 'α' subunit.

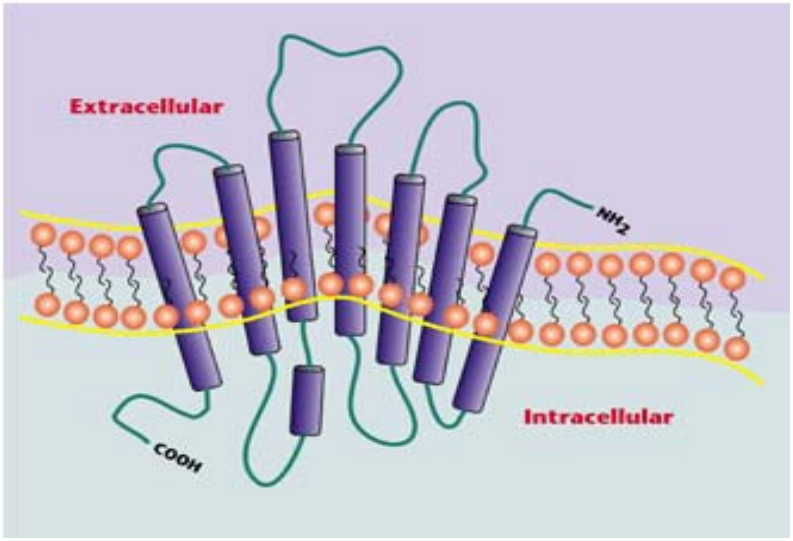

Figure 2.4: G-Protein Coupled Receptor protein

The activated GTP-bound 'α' subunit then dissociates from the 'β' and 'γ' subunits. The 'β' and 'γ' subunits remain bound to each other and function as the 'βγ' complex. The 'βγ' complex and the GTP-bound 'α' subunit interact with their targets, for example, an enzyme or an ion channel, to transmit the signal. The bound GTP becomes a GDP due to hydrolysis after the transmission of the signal. The GDP-bound 'α' subunit re-associates with the 'α' complex to form a heterotrimeric G-protein, which is ready for another cycle of transmission of a signal through a GPCR [Cooper, 2000].

GPCR are involved, for example, as light sensing molecules in the eye (Rhodopsin), odorant receptors in the olfactory system, and as taste receptors on the tongue [Foreman, 2003]. They are found in a wide range of eukaryotic organisms. The GPCRDB, a database system for GPCR [Horn, 1998], divides the GPCR superfamily into five major classes based on the ligand types, functions, and sequence similarities, as shown in Table 2.9. The sequences of different GPCR classes are highly diverged from each other, except that they share one common structural feature, i.e., they all have seven hydrophobic transmembrane regions. GPCRs within a class, share common functions with more sequence similarities. Class A, the Rhodopsin like class, is by far the most populated GPCR class with more than 3,500 members in the database. Each class is further divided into subclasses, sub-subclasses, and so forth; depending upon the common agents they bind to and sequence similarities.

Table 2.9: Major GPCR classification based on GPCRDB (as of Nov' 2003)

Class	Examples	Number of entries
A: Rhodopsin like	Rhodopsin and adrenergic receptors	3,519
B: Secretin like	Calcitonin receptors	217
C: Metabotropic glutamate/ pheromone	Metabotropic receptors	131
D: Fungal pheromone	pheremone receptors	24
E: cAMP receptors (Dictyostelium)	cAMP receptors	5

Identification of the function of GPCR sequences is important in biomedical and pharmaceutical research, since GPCRs play key roles in many biologically important functions and are related to many diseases (e.g., neurological, cardiovascular diseases, depression, obesity, pain and viral infections [Predix, 2000]). However, identifying and classifying this membrane protein family is a difficult task due to the high levels of divergence observed among the GPCR family members. Therefore, it becomes important that there be a way to accurately and efficiently identify any new GPCRs from genomic data. This would benefit the pharmaceutical research and give a better understanding of GPCR functions.

2.8.2 Myoglobin

Myoglobin is a single chain globular protein of about the size of 153 amino acids, containing a heme(iron-containing porphyrin) prosthetic group in the center around which the remaining apoprotein folds. It has a molecular weight of 16,700 daltons, and is the primary oxygen-carrying pigment of muscle tissues [George, 2004]. Unlike the blood-borne hemoglobin, to which it is structurally related [Harvey, 2000], this protein does not exhibit cooperative binding of oxygen, since positive cooperativity is a property of multimeric / oligomeric proteins only. Instead, the binding of oxygen by Myoglobin is unaffected by the oxygen pressure in the surrounding tissue. Myoglobin is often cited as having an instant binding tenacity to oxygen given its hyperbolic oxygen dissociation curve. High concentrations of Myoglobin in muscle cells allow organisms to hold their breaths longer. In 1958,

Myoglobin structure was determined by X-ray crystallography by John Kendrew [Kendrew, 1958].

The methods developed in this thesis will also be applicable to other proteins. GPCRs and Myoglobin are used in the study due to their scientific importance.

The Set theory concepts are utilized in Sequence Arithmetic module (Chapter III). Rough Set Theory and decision tree is used in the construction of Reduct based Decision Tree (Chapter IV). The relationships between attributes and instances in a binary decision table can be obtained from node information of Concept Lattice (Chapter VI)

CHAPTER III
SEQUENCE ARITHMETIC

Proteins are classified by various methods; one technique is based on motifs, the functional or structural aspect of a protein. Motif is a consecutive sequence of characters that occur frequently in a protein. The occurrence of a specific motif is one of the significant features [Falquet, 2002].

Motif based analysis needs experts intervention in protein classification. Since the main objective of the study is to develop machine-learning tool with minimal human interference and the power of motifs motivated to contemplate the analysis based on occurrence of characters in the present work.

Protein is a sequence of 22 characters (20 labels for amino acids and two characters (B, Z) for substitution of amino acids). A protein essentially need not contain all amino acids. The absence of amino acids can be deliberated as a feature of that protein. This chapter expounds a systematic way of dealing with sequences by introducing various types of sets based on the occurrence of amino acids in a given protein (or) set of proteins.

Sets, operations and representation aspects of the sets and inferences associated with them have been referred as Sequence Arithmetic (SA) in the present work.

Sequence definitions along with their characteristics, the generation of different sets have been discussed in Section 3.1. Creation of knowledge base has been described in Section 3.2. Demonstration of Sequence Arithmetic has been illustrated in Section 3.3. The knowledge extraction process is presented in Section 3.4. Design and implementation issues of Sequence Arithmetic have been discussed in Section 3.5. Demonstration has been carried out on G-Protein Coupled Receptors and Myoglobin datasets. The reduction in search space has been elucidated in Section 3.6. The complexity analysis of the module is discussed in Section 3.7, followed by conclusions in Section 3.8.

3.1 SEQUENCE ARITHMETIC

Sequence types and basic string definitions are discussed in this section.

3.1.1 Sequence

Sequence has been defined in many ways depending upon the instance, location and domain. Some of the definitions of sequences with examples are given as follows:

- A serial arrangement in which things follow in logical order or a recurrent pattern.

  →The sequence of names was alphabetical.
- Following of things one after another in time.

  → The doctor saw patients in a sequence.
- Film consisting of a succession of related shots that develop a given subject in a movie.
- Succession: the action of following in a particular order.

  → He played the trumps in sequence.
- In mathematics, a sequence is a list of objects (or events) arranged in a 'linear fashion', such that the order of the members is well defined and significant.
- The linear arrangement of building blocks in biological macromolecules like DNA, RNA, protein and polysaccharides. DNA and RNA macromolecules are linear polymers of nucleotides. Proteins are linear polymers of amino acids. Polysaccharides are linear and branched polymers of monosaccharides (sugars).

In the present research, protein is a sequence of varying length with permutations of 22 characters (20 amino acids and 2 substitution characters).

3.1.2 String

Let Σ be any finite set of alphabets. A finite sequence of zero or more elements chosen from Σ is a string over Σ. The length of the string '*n*' is denoted by |n|. The set of strings of length '*k*' is denoted by Σ^k.

The well-formed formulae representation of a string as defined [Tremblay, 1987] [Mott, 1999] are:

1. $\Sigma^0 = \{\Lambda\}$; it is null string.
2. $\Sigma^{k+1} = \{wa \mid w \in \Sigma^k \text{ and } a \in \Sigma\}$ for $k \geq 0$.
3. $\Sigma^* = \bigcup_{k=0}^{n} \Sigma^k$, denoting the set of all strings over Σ.
4. $\Sigma^+ = \bigcup_{k=1}^{n} \Sigma^k$, denoting the set of all non-null strings over Σ.

Protein is a sequence consisting of 22 characters (20 amino acids and 2 substitution characters).

Σ={A, C, D, E, F, G, H, I, K, L, M, N, P,Q, R, S, T,V, W, Y}∪{B, Z} (Table 2.1).

Each and every sequence obtained from the definition may not be a protein though characters confine to the set of character 'Σ'. The following characteristics of proteins have been obtained:

i. Length of sequence.
ii. Location / Space preserving information.
iii. Frequency tabulation of characters (pattern representation).
iv. Statistical information based on Run length encoding.

Frequency tabulation of a character ascertains the information of occurrences of a character in the sequence. The characters, which have non-trivial frequency in a frequency table of a given sequence 's' constitute A-Set of the sequence and is denoted as A(s). Similarly the character that has trivial frequency (i.e., '0') in a frequency table is called D-Set of the sequence and written as D(s).

$$D_\Sigma(s) = \Sigma - A(s).$$

Similarly A-Set, D-Set for a given set of sequences 'S' are denoted by a

1. $A(S) = \bigcup_{s \in S} A(s)$.
2. $D(S) = \Sigma - A(S)$.

In addition to these two sets, Prime set (denoted as P(S)) has been defined which is intersection of A-Set of each sequence of S.

$$P(S) = \bigcap_{s \in S} A(s)$$

By considering a set of proteins the above concepts have been demonstrated in the following section.

3.1.3 Protein Sequences

Proteins are long chains of twenty different amino acids and two substitutions characters i.e., Σ = {A, C, D, E, F, G, H, I, K, L, M, N, P,Q,R,S,T,V,W,Y} $\cup$ {B,Z} (Table 2.1), that serve as building blocks for proteins. Each amino acid has a specific chemical structure, which contains a carbon backbone similar to all other amino acids, and a residue. The residue varies from one amino acid to another amino acid. The length of a protein chain can range from 50 to 1000-5000 amino acids (350 on average). Proteins are the molecules with diverse chemical properties.

The proteins [Mount, 2001]

- form functional and structural machinery of each cell in every organism.
- are involved for giving biological signals.
- act as catalyst for metabolic activities of the body.
- have many levels of structure *viz.*,
    i. Primary structure, which is the 1-D sequence.
    ii. Secondary structure (2-D), which is a composition of the regular substructures that the protein polymer forms due to streric and hydrogen bond interactions.
    iii. Tertiary structure (3-D), which is the most complex protein structure, composed of multiple chains.

Proteins are essentially grouped according to their secondary structure but not by functional families. The focus in protein classification has been on finding proteins that have similar chemical architectures. The significant property of a protein is the length and composition of amino acid chain. The series can be obtained automatically from the gene that encodes for the protein. The amino acids composition of a protein determines the tertiary structure, which is unique [Mount, 2001].

A protein sequence is a sequential arrangement of the amino acids (with permutation). Sequence Arithmetic for protein classification aims at finding the information with respect to essential amino acids, which occurs in a family. This amino acids information has been used to map with the sequences that are already present in the protein database-Brookhaven National Laboratory [Brookhaven NL].

Various Sequence Arithmetic operations have been used to reduce the search space, find the nearest neighbors within a family and to which family the sequence belongs.

3.1.4 Mathematical Representation of Set

Let 's' be any sequence and 'S' be set of 'n' sequences, then the mathematical representations are as given below:

$A(s) = \{x / x \text{ is valid character in the sequence 's'}\}$

$A(S) = \bigcup_{s \in S} A(s)$

$A(s) \subseteq A(S)$

$D(S) = \Sigma - A(S)$

$P(S) = \bigcap_{s \in S} A(s)$

The knowledge base has been built by finding the Σ, A-Set, D-Set, and P-Set for each family of sequences collected from the sample data of sequence Information System. They have been used to determine the family of an unknown sequence using expert system techniques namely forward chaining and backward chaining [Ramadevi, 2004b].

3.1.5 Alphabet Set (A-Set)

A-Set of a sequence: A(s): It consists of all the symbols, which appear in the sequence.

A-Set of a class: A(C): It is the union of A-Set of all the sequences, which belong to a particular class.

A-Set of a family: A (F): It is the union of A-Set of all the class that belongs to a particular family. In some cases, it is equal to Universal Set (Σ).

Example 3.1:

Consider the sequences of a class belonging to Myoglobin family

Sequence 1(s_1)

GLSDGEWELVLKTWGKVEADIPGHGEFVLVRLFTGHPETLEKFDKFKHLKTE
GEMKASEDLKKQGVTVLTALGGILKKKGHHEAEIQPLAQSHATKHKIPIKLEF
ISDAIIHVLQSKHPAEFGADAQGAMKKALELFRNDIAAKKELGFQG

$A(s_1) = \{A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W\}$

Sequence 2(s_2)

GLSDGEWQLVLHVWGKVEADLAGHGQEVLIRLFKGHPETLEKFNKFKHIKSE
DEMKASEDLKKHGVTVLTALGGVLKKKGHHEAEIKPLAQSHATKHKIPIKLE
FISEAIIHVLQSKHPGFGADAGAMNKALELFRKDIAYAKKELGFQG

$A(s_2)$ = {A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

Let C = $\{s_1, s_2\}$

A-Set for the class is the union of the A-Set of all the sequences $\mathbf{s_i}$ present in the class. The accuracy of the set depends on the number of samples taken.

$A(C) = A(s_1) \cup A(s_2)$

={{A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W}

$\cup$ {A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}}

= {A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

A-Set for a family: It is the union of A-Set of all the classes that belong to a particular family. A-Set for Myoglobin family is as follows:

A(Myoglobin) = {A,B,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y,Z} (Appendix A)

3.1.6 Difference Set (D-Set)

D-Set for a sequence: It consists of all the symbols that are not present in the sequence.

D-Set for the Sequence 1 and Sequence 2 are as follows:

$D(s_1)$ = {B,C,Y,Z}

$D(s_2)$ = {B,C,Z }

D-Set for a class: It is the intersection of D-Set of all the sequences, which belong to a particular class.

D(C) = {{B,C,Y,Z} $\cap$ {B,C,Z}}

= {B,C,Z}

D-Set for a family: It is the intersection of D-Set of all the classes that belong to a particular family.

D(Myoglobin) = { }(Appendix A)

3.1.7 Prime Set (P-Set)

P-Set for a class-P(C): It is the intersection of A-Set of all the sequences, which belong to a particular class.

P-Set for a family- P(F): It is the intersection of A-Set of all the sequences, which belong to a particular family.

Example 3.2:

Consider the Sequence 1 and Sequence 2

$$P(C) = \{\{A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W\} \cap \{A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y\}\}$$
$$= \{A,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W\}$$

P(Myoglobin) = { } /* it is null set */

The A-Set, D-Set and P-Set for the Myoglobin family and G-Protein Coupled Receptors ha been given Appendix A

3.2 SEQUENCE ARITHMETIC KNOWLEDGE DATABASE

The classification pattern of proteins as discussed in Chapter II comprises:

- Proteins are classified into families.
- Families into classes.
- Classes into subclasses and so on.

A unique identification number (Fid) has been given to a family. Attaching a Family ID(Fid) to the sequence helps to acquire the knowledge for classification of a new sequence. The information regarding A-Set, D-Set and P-Set of the family have been generated and stored in the Sequence Arithmetic Knowledge Database.

These sets have been utilized in framing the Sequence Arithmetic Rules, which are discussed in Section 3.5. These generated rules have been stored in Sequence Arithmetic Knowledge Database and are used for the identification of unknown sequence. It has been possible to identify an effective inference engine model that works on this knowledge base for the classification of unknown sequences.

3.3 DEMONSTRATION OF SEQUENCE ARITHMETIC

The input to the Sequence Arithmetic module is the unidentified sequence 'y'. A(y) and D(y) of the unknown sequence are computed. Knowledge has been extracted by mapping these sets with the information present in the Sequence Arithmetic Knowledge Database by following the rules (Section 3.4.1). The output of the Sequence Arithmetic module is the family to which the sequence belongs. The family information < F > consists of A-Set, P-Set and D-Set for the corresponding family along with the Fid. The output of the Sequence Arithmetic module will be the A(y), D(y), A(F), D(F) and P(F). The Sequence Arithmetic module is shown in Figure 3.1.

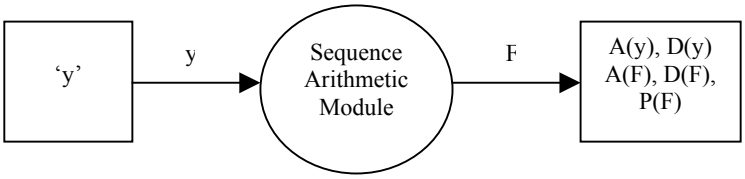

Figure 3.1: Sequence Arithmetic module

The internal processes are shown in Figure 3.2. Sequence Arithmetic Knowledge Database consists of data related to various families. When unknown sequence 'y' is given as input to Sequence Arithmetic, the map process is used to map the unknown sequence information with the different families' information in the Sequence Arithmetic Knowledge Database.

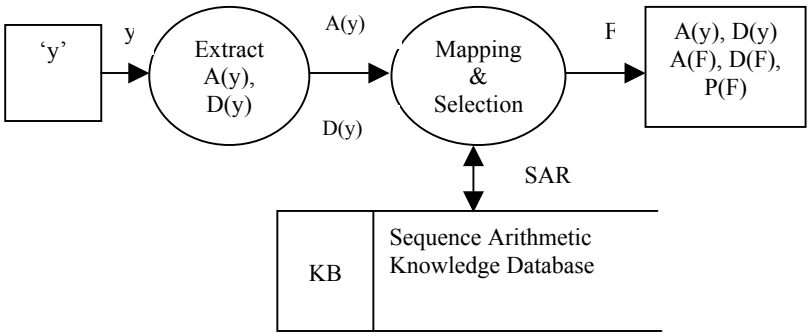

Figure 3.2: Internal processes of the Sequence Arithmetic module

3.3.1 Sequence Arithmetic Datasets

For a given set of sequences, which belong to a class, sets have been generated as discussed in Section 3.1.4. These sets play an important role in the classification / identification of a novel sequence. G-Protein Coupled Receptors dataset and Myoglobin dataset have been used for generation of different sets. A sample of 1041 out of 3896 G-Protein Coupled Receptors family consisting of five classes (Class A, B, C, D and E) has been taken for demonstration [Horn, 1998]. Sequence Arithmetic has also been demonstrated on sample set of Myoglobin sequences consisting of 399 out of 727 available sequences. These sequences are categorized into 36 classes.

3.4 KNOWLEDGE EXTRACTION FROM SEQUENCE ARITHMETIC REPRESENTATION FOR PROTEIN FAMILIES

The input for this process is sequence information. The following information about the sequence is required for the knowledge extraction:

1. Sequence.
2. Sequence Family ID.

The necessary steps in the process of extraction of knowledge base for the sequences are as follows:

1. Extract the distinct family IDs from the sequence samples.
2. Compute A-Set for each sequence.
3. For each sequence ‘s’ within a class and each class ‘C’ within a family ‘F’ apply the following rules:

   i) $A(C) = \bigcup_{s \in C} A(s)$.

   ii) $A(F) = \bigcup_{C \in F} A(C)$.

4. Compute D-Set based on rules below:

   i) $D(C) = \Sigma - A(C)$.

   ii) $D(F) = \Sigma - A(F)$.

5. Compute the P-Set of the family.

The intersection of A-Set of all the sequence of a family generates the P-Set of a family.

i) $P(C) = \bigcap_{s \in C} A(s)$.

ii) $P(F) = \bigcap_{C \in F} A(C)$.

3.4.1 Inference Engine for Identification of Unknown Sequence

The knowledge base has been built by finding the Σ, A-Set, D-Set, and P-Set for each family of sequences collecting the sample data from sequence Information System. They have been used to determine the family of unknown sequences using expert system techniques namely forward chaining and backward chaining.

Lemma 3.1: If $s \in F$, then $D(F) \subseteq D(s)$.

Proof: Let $s \in F$, then $A(s) \subseteq A(F)$.

$D_\Sigma(s) = \Sigma - A(s)$ and

$D_\Sigma(F) = \Sigma - A(F)$.

If $A(s) \subseteq A(F)$ then $D(F) \subseteq D(s)$ (Proved).

Lemma 3.2: If $s \in F$, then $P(F) \subseteq A(s)$.

Proof: Let $s \in F$, then $A(s) \subseteq A(F)$.

$P(F) = \cap A(s)$

If $A(s) \subseteq A(F)$ then $P(F) \subseteq A(s)$ (Proved).

Corollary 1: If $D(F) \not\subset D(s)$ then s will not be a sequence in family F.

Corollary 2 If $P(F) \not\subset A(s)$ then s will not be possible sequence in family F.

The above said lemmas and corollary helps in developing algorithms and improving the efficiency of the algorithms for classifying a new sequence. The following rules are used for reducing complexity of classification of unidentified sequences into one of the available family.

Rule 1: If $D(F) \not\subset D(s)$, then do not search in family F.

Rule 2: If $P(F) \not\subset A(s)$, then do not search in family F.

Rule 3: If $D(F) \subseteq D(s)$, F is possible search space.

Rule 4: If $P(F) \subseteq A(s)$, F may be potential family where ‘s’ can be.

The rules for not searching definitely are Rule 1 and Rule 2, whereas Rule 3 and Rule 4 still have ambiguity. In order to reduce the ambiguity, exhaustive knowledge has to be extracted to arrive at better decision rules. The same are subject of interest in Chapters IV, V and VI.

The inference engine follows Algorithm FIdentifier for the classification of unknown protein sequence '**y'**.

Algorithm 3.1

Algorithm FIdentifier (y)

Input: Sequence 'y'.

Output: Family F.

1. Compute the A(y).
2. Use the following conditions for deciding whether the sequence belong to a family or not using Sequence Arithmetic Knowledge Database. (Appendix A)
   a. $A(y) - A(F) = \Phi$
   b. $P(F) - A(y) = \Phi$
   c. $D(F) \cap A(y) = \Phi$

   If any of the above conditions are not satisfied, it implies that the sequence does not belong to that family.
3. Step two will give a set of family / families closely related to the sequence 'y'.

Algorithm 3.1 will give a set of protein families, which are very close to the unknown sequence 'y'. Minute operations can be performed for further classification for finding exact family. The algorithm may return more than one family for some sequences, therefore, it is advised to rank these identified families by taking the closeness of A(s) to P(F) and D(s) to D(F). The methods developed in the following chapters need to be followed for all identified families in rank order.

3.5 SEQUENCE ARITHMETIC MODULE

The architecture of the model expert system and different methods used for the knowledge extraction are discussed in this section.

3.5.1 Architecture of Model Expert System

The architecture of a Model Expert System is shown in Figure 3.3. All the sequences have been stored in the Sequence Database, in the format as in Table 3.1

using a VB module. Here the Sequence Database can be updated whenever new sequence is identified. The different interface methods generated in this Framework are shown in Table 3.2 and the output screens are as in Figures 3.4,3.5, 3.6 and 3.7.

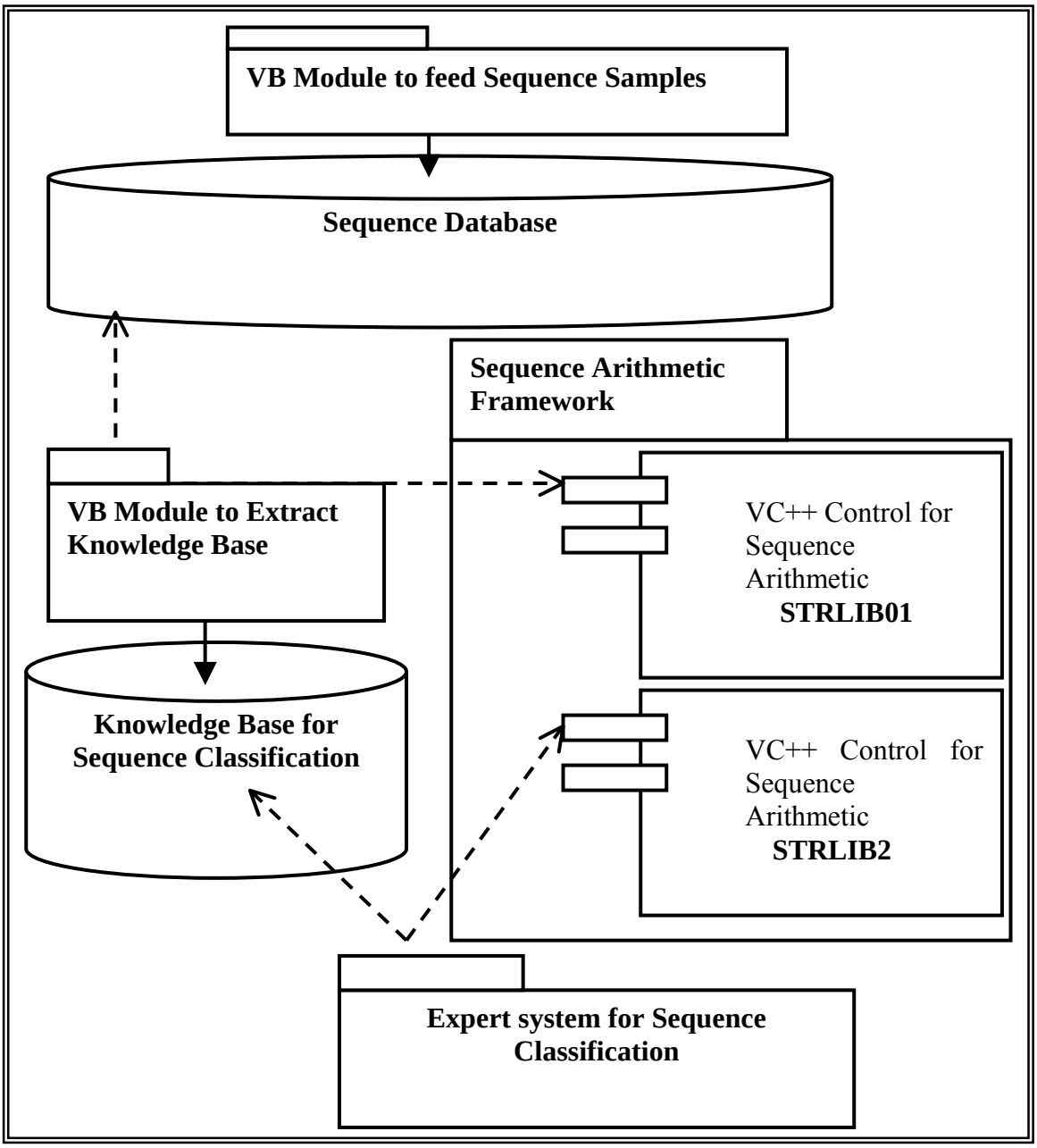

Figure. 3.3: Architecture of Expert System

Table 3.1: Format for sequence stored in database

S.No	Sequence ID	Fid	Sequence

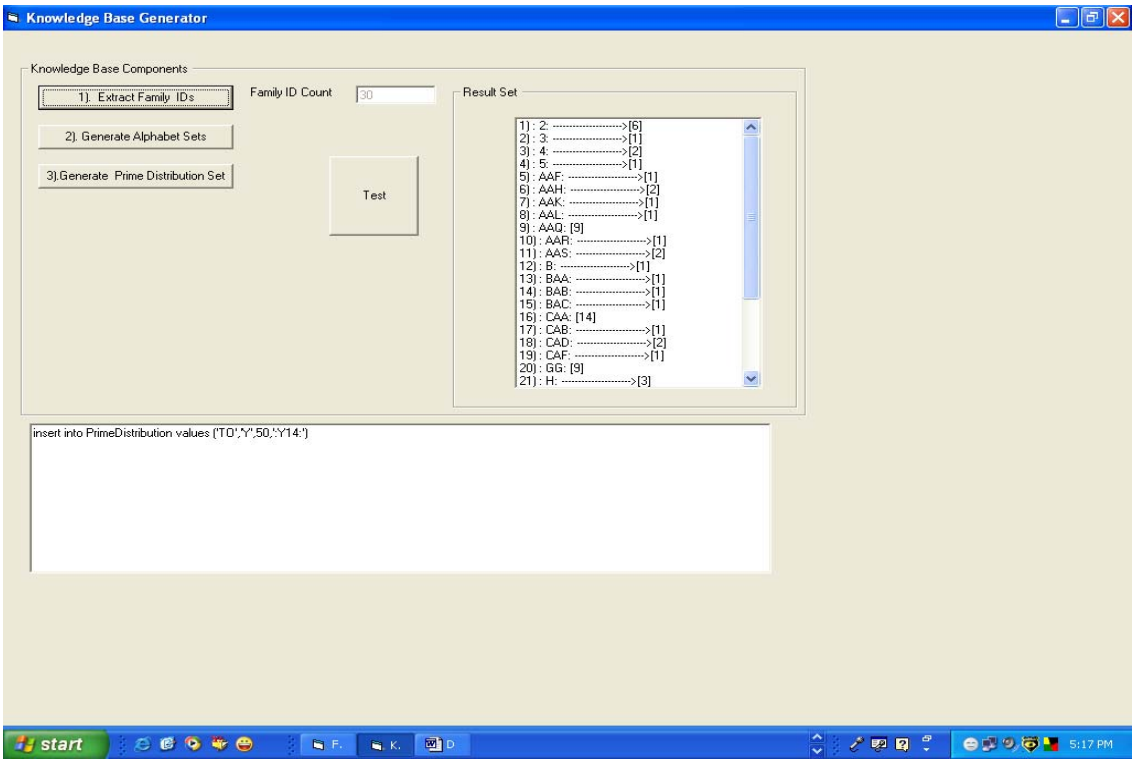

Figure 3.4: Extraction of sequence –minimum number of sequences are considered.

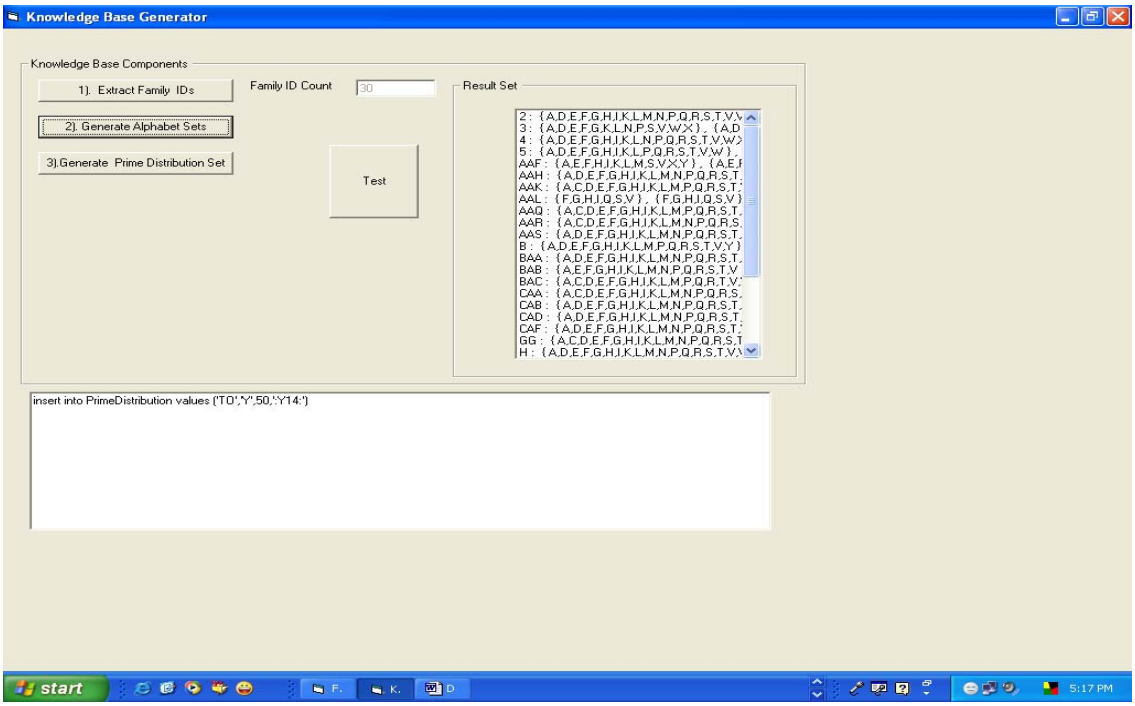

Figure 3.5: Generation of Alphabet Sets for all sequences in the database

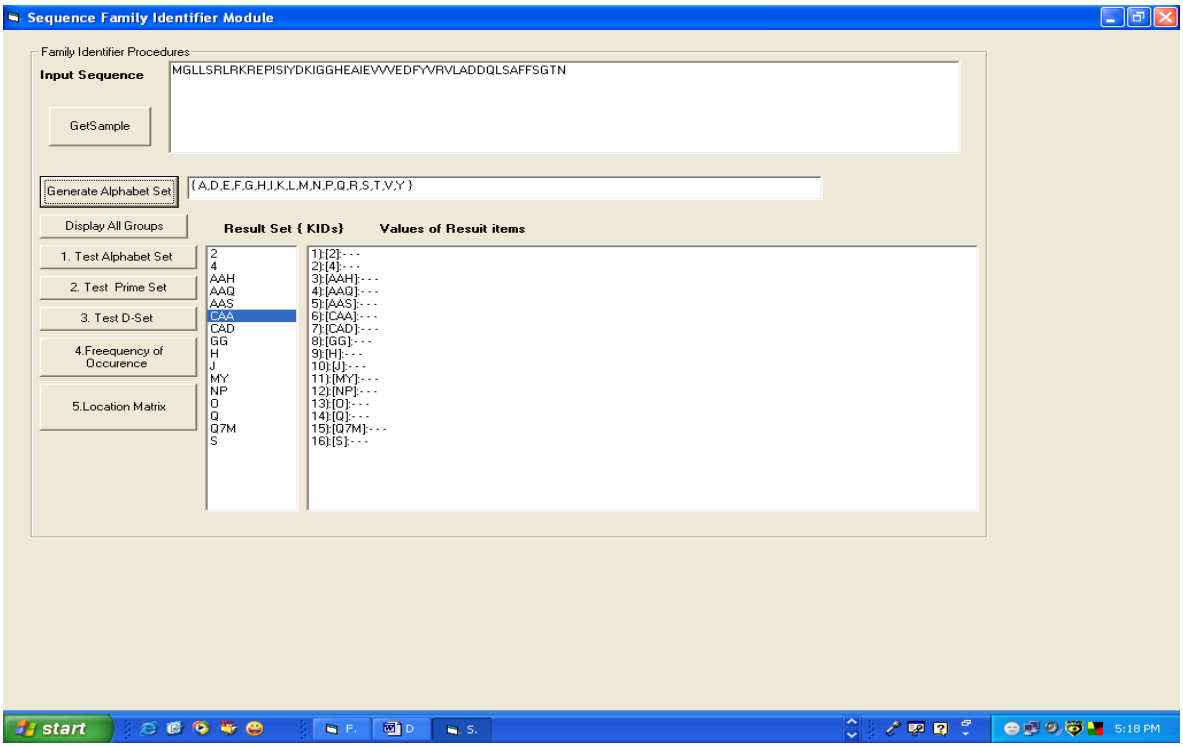

Figure 3.6: Generation of A-Sets for a given sequence

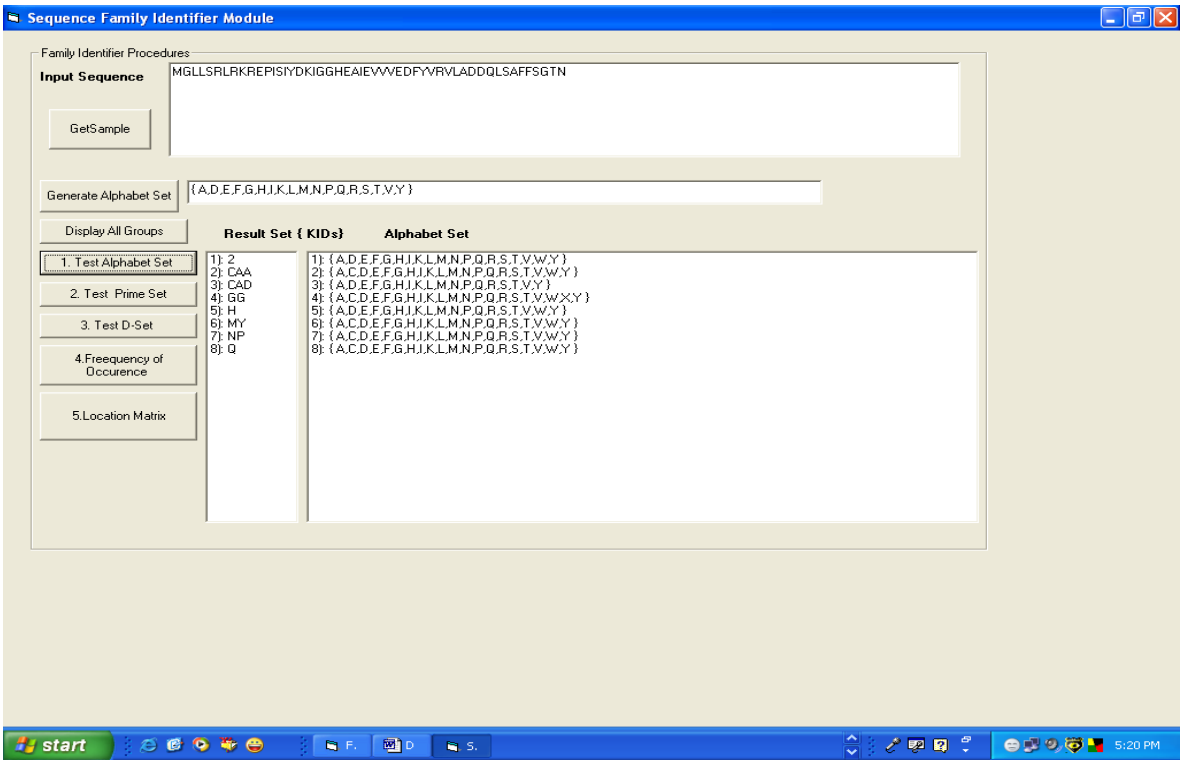

Figure 3.7: Testing of A-Set for a given sequence and displaying the families that are closer to it

Table 3.2: STRLIB01 interface methods

S.No	Function Prototype	Remarks /Samples
01	BSTR GetSet (LPCTSTR str)	Returns Alphabet Set for given sequence
02	BSTR GetUnionset (LPCTSTR str1, LPCTSTR str2)	Gets Union Set
03	BSTR GetSetInterSection (LPCTSTR str1, LPCTSTR str2)	Returns Intersection set
04	BSTR GetDiffSet (LPCTSTR str1, LPCTSTR str2)	Returns Difference set
05	Short IsEmpty(LPCTSTR Str1)	Returns whether a set is empty or not
06	Short IsSubSet (LPCTSTR Master, LPCTSTR Prime)	Returns whether a set is subset of other or not
07	Short GetSLength (LPCTSTR Str)	Returns length of a sequence
08	BSTR GetLocationMatrix (LPCTSTR Str, LPCTSTR bit)	Returns location matrix of a string
09	BSTR InterLocMat (LPCTSTR Str1, LPCTSTR Str2, LPCTSTR Bit, short L)	Finds Intersection of two location matrices up to given length of sequence
10	Short ProcessString (LPCTSTR Str, LPCTSTR bit)	To find number of occurrence of a given string
11	BSTR TrimYRDString (LPCTSTR Str)	Removes spaces

Sequence Arithmetic has been used to identify a family, which has been found to be faster than other methods. The arithmetic is totally dependent on known protein samples for the classification. The accuracy of this method depends on the size of the protein sequence information system and accuracy of the data in the sampling system. The limitation of the method is that it cannot give assurance for finding exact protein family. But it gives probable set of family / families to which the protein sequence belongs, depending on the behavior of the sample protein sequence of the family.

3.5.2 Sets Generation for G-Protein Coupled Receptors Family

The G-Protein Coupled Receptors family has been taken as a sample database; they are divided into 5 classes (class A, B, C, D and E). By considering the sequences, which belong to Class A of G-Protein Coupled Receptors dataset and the following sets, are generated:

A(A) = {A,C,D,E,F,G,H,I,J,K,L,M,N,P,Q,R,S,T,V,W,Y}

D(A) = {B,Z}

P(A) ={ A,C,D,E,F,G,H,I,J,K,L,M,N,P,Q,R,S,T,V,W,Y}

Similarly the outputs for the remaining classes is as shown in Table 3.3

Table 3.3: A-Set, P-Set and D-Set for GPCR database

Class	A-Set	D-Set	P-Set
A	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
B	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
C	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
D	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
E	{ A,C,D,E,F,G,H,I,J,K,L M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L M,N,P,Q,R,S,T,V,W,Y}

Given an unknown sequence **'y'**, generate the A(y), D(y) and then apply Rules 1- 4 as given in Section 3.4.1.These A(y) and D(y) have been mapped with the rules in Sequence Arithmetic Knowledge Database and family information has been obtained.

If the D(y) is {B,Z} then the sequence **'y'** belongs to GPCR family.

If A(y) = {A,C,D,E,F,G,H,I,J,K,L,M,N,P,Q,R,S,T,V,W,Y} $\supset$ P (GPCR) then it belongs to GPCR family [Ramadevi, 2006a]**.**

3.5.3 Testing and Result Analysis

Consider the unknown Sequence 'y'

GLSWELVLKTWGKVEADIPGHGEFVLVRLFTGHPETLEKFDKFKHLKTEGEM
KASEDLKKQGVTVLTALGGILKKKGHHEAEIQPLAQSHATKHKIPIKYLEFISD
AIIHVLQSKHPAEFGADAQGAMFRNDIAAKYKELGFQG

A(y) = { A,C,D,E,F,G,H,I,J,K,L,M,N,P,Q,R,S,T,V,W,Y}

D(y) = {B,Z}

A(y) $\supset$ P (GPCR) and A(y) $\supset$ P(Myoglobin).

Therefore 'y' belongs to two families.

Now compute A(y) - P(GPCR) ={ } and

A(y) - P(Myoglobin) ={A,C,D,E,F,G,H,I,J,K,L,M,N,P,Q,R,S,T,V,W,Y}

The identified families are ranked in increasing order of number of elements in the resultant set. The families are searched in rank order, so the sequence 'y' is searched in GPCR first and it belongs to GPCR family (Appendix A). Thus the search is confined to only GPCR family.

3.6 REDUCTION OF SEARCH SPACE

If an unknown sequence 'y' is given then BLAST finds similar sequences by comparing 'y' with all the sequences in the database. In Sequence Arithmetic module the family information has been obtained by computing A(y), D(y) and subsequently applying the Sequence Arithmetic Rules present in the Sequence Arithmetic Knowledge Database. The first level of horizontal fragmentation of domain search space is achieved.

A sample dataset of Myoglobin family and GPCR family containing 1440 sequences have been considered for demonstration of SA module. A random selection of 60 sequences has been made and used for testing. Sequence Arithmetic has been applied on them. 32 sequences were classified into one family (Myoglobin) and 28 sequences has multi-valued output. If a new sequence is given, it is classified into one of the two families, thus on an average 60% of the database has to be searched. Thus the reduction of domain search is about 40% by applying Sequence Arithmetic module (Appendix D).

3.7 COMPLEXITY

The complexity of the Sequence Arithmetic module depends on A-Set, D-Set and P-Set and storing the data in Sequence Arithmetic Knowledge Database.

Let T_A be the complexity for finding the A-Set. For initial 'n' sequences, a single scan for A-Set is to be performed. T_A = O(nm), where 'm' is maximum length of any sequence. If the number of classes within a family is 'C' then $T_{A(C)}$ = O(22n)(as number of characters is 22).
For a family T_{SA} =O(nm)+O(3n*22) = O(nm).

Let 'b' be the complexity of searching for a sequence in a database with 'n' sequences without using Sequence Arithmetic.

Family information is obtained by applying Sequence Arithmetic module. If n_1 sequences are present in a family, then $n_1 < n$.

With Sequence Arithmetic it will be $T_{SA} + O(n_1 b)$ which will be less than or equal to O(nb). Therefore the complexity of Sequence Arithmetic is $O(n_1 b) + O(nm)$.

3.8 SUMMARY

The search space has been considerably reduced both for Myoglobin and G-Protein Coupled Receptors. The horizontal fragmentation of the dataset has been achieved by using Sequence Arithmetic, thus reducing the complexity of the search. The generation of A-Set and D-Set for the unknown sequences, and applying the rules reduced the domain search space to either of the families. The application of the Sequence Arithmetic Rules to Myoglobin and GPCR families has shown about 40% reduction in the search space (Appendix D).

The result of applying Sequence Arithmetic is that family information is generated and at present the search is confined only to it. The inability to identify the class within the family is conquered by proceeding to the Reduct based Decision Tree module (Chapter IV) and Neighborhood Associations module (Chapter V).

CHAPTER IV
REDUCT BASED DECISION TREE

The Sequence Arithmetic has been developed and illustrated in Chapter III. It uses Set theory concepts and generates three types of sets. These sets have been used to extract family information of proteins. Once the family is obtained, the class within the family is to be identified. It was experiential that the Sequence Arithmetic is unable to identify the class of a protein. This chapter extends a Rough Set based tool named as Reduct based Decision Tree for identifying the class. The output data generated from Sequence Arithmetic has been compiled as A-Table, D-Table and P-Table and given as input to the Reduct based Decision Tree module. Rules extraction process from Reduct based Decision Tree is presented in this chapter.

A novel approach of Rough Set Theory and decision tree has been developed in the present study. Predominant attributes (approximate reduct) have been generated in Rank order and used in the same order in the construction of the decision tree called Reduct based Decision Tree, and the rules generated are called Reduct based Decision Tree Rules. These rules have been stored in a database called Reduct based Decision Tree Rules Database. Demonstration details of the methodology along with illustrations are discussed in Section 4.3.

A critical analysis of this novel RDT algorithm has been performed by considering A-Table, D-Table and P-Table. Its performance is compared with various decision trees by using WEKA tool [David, 2007] for standard datasets.

4.1 GENERATION OF A-TABLE, D-TABLE AND P-TABLE

A protein family ‘f’ has ‘k’ classes $\{C_1, C_2, C_3, C_4,\ldots, C_k\}$. The corresponding A-Sets of the classes be $\{A_{C1}, A_{C2}, A_{C3}, A_{C4},.. , A_{Ck}\}$. Let R be number of distinct A-Sets among $\{A_{C1}, A_{C2}, A_{C3}, A_{C4},.., A_{Ck}\}$ which are labeled as $A_1, A_2,..,A_R$.
Let $\delta_{ij} = 1$ if A_j is the A-Set of C_i, i.e., $A_j = A_{Ci}$.

An array representation of δ_{ij} is a table, referred as A-Table. D-Table and P-Table are the corresponding tables constructed by following the above description by considering D-Sets and P-Sets. The classID is concatenated as one more column to the above table resulting in decision tables and has been demonstrated on the datasets (Myoglobin and G-Protein Coupled Receptors) in the sections to follow.

4.1.1 Myoglobin Dataset

The Myoglobin family constitutes 36 classes. It has been observed that there are 15 distinct types of A-Sets and D-Sets, 17 distinct types of P-Sets (Appendix A).

The available 36 classes of Myoglobin family have been uniquely numbered from 1 to 36 (Appendix A). They form the decision attribute, which precedes the attribute list of corresponding tables (Tables 4.1a, 4.1b and 4.1c).

The entries of the A-Table are computed as follows:

A-Table (class, A_i) = '1', if the A_i is the A-Set of the class.
= '0' otherwise. (Table 4.1a).

Example 4.1:
A1 is the A-Set for class 1 and class 26, therefore the decision table entry is '1' for the rows 1 and 26 with A1 (column value).

The entries of the D-Table are computed as follows:

D-Table (class, D_i) = '1', if the D_i is the D-Set of the class.
= '0' otherwise. (Table 4.1b).

Example 4.2:
D1 is the D-Set for class 1 and class 26, therefore the decision table entry is '1' for the rows 1 and 26 with D1(column value).

Similarly, the entries of the P-Table are computed as follows:

P-Table (class, P_i) = '1', if the P_i is the P-Set of the class.
= '0' otherwise. (Table 4.1c).

Example 4.3:
P1 is the P-Set for class 1 and class 14, therefore the decision table entry is '1' for the rows 1 and 14 with P1(column value).

It has been observed that A-Table and D-Table are similar, therefore it is sufficient to consider one table.

Table 4.1a: A-Table for Myoglobin Dataset

Class	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
5	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
15	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
19	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
21	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
25	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
26	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
31	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
33	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 4.1b: D-Table for Myoglobin Dataset

Class	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
5	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
15	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
19	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
21	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
25	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
26	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
31	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
33	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 4.1c: P-Table for Myoglobin Dataset

Class	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
5	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
14	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
20	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
23	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
26	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
30	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
33	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

The A-Table and D-Table are similar. The knowledge derived from A-Table will be the same as that derived from D-Table; hence one table is sufficient to obtain knowledge. However, number of distinct P-Sets is more than the number of distinct A-Sets, so the present study is confined to P-Sets.

4.1.2 GPCR Dataset

GPCR family is divided into 5 classes. The A-Set, D-Set and P-Set values of classes A, B, C, D and E are same (Table 3.3). The number of distinct A-Sets, D-Sets and P-Sets are one, thus the corresponding A-Table, D-Table and P-Table has one value only as in Tables 4.2a, 4.2b and 4.2c.

Table 4.2a: A-Table for GPCR

Class	A1
A	1
B	1
C	1
D	1
E	1

Table 4.2b: D-Table for GPCR

Class	D1
A	1
B	1
C	1
D	1
E	1

Table 4.2c: P-Table for GPCR

Class	P1
A	1
B	1
C	1
D	1
E	1

The rows of Tables 4.1a, 4.1b and 4.1c are not identical. Any one of the decision tables can identify the Myoglobin class/classes with the acquired knowledge. Whereas the Tables 4.2a, 4.2b and 4.2c with identical rows indicate that the derived data based on Sequence Arithmetic is not having enough potential to categorize the classes of GPCR.

The following section develops Reduct based Decision Tree that is used for acquiring knowledge for discriminating classes based on any one of the Tables of 4.1 (Myoglobin).

4.2 REDUCT BASED DECISION TREE (RDT)

Decision table is an Information System and core knowledge can be extracted from it. Two algorithms have been developed to calculate core attributes and reducts for feature selection [Hu, 2004]. These algorithms are applied to a wide range of real life applications with very large data sets. A Quick reduct algorithm to compute a minimal reduct without exhaustively generating all possible subsets has been explored

[Jenson, 2004]. Rough Sets with Heuristics (RSH) and Rough Sets with Boolean Reasoning (RSBR), have been used for attribute selection and discretization of real-valued attributes [Zhong, 2001]. Ant Colony optimization and Improved Quick reduct technique is combined for feature selection [Jaganathan, 2007].

A new method has been introduced in the present work, for extracting attributes, which is an approximate reduct. The attributes are extracted one by one, preserved in Rank order and are called predominant attributes. A decision tree construction method has been proposed based on the set of predominant attributes [Ramadevi, 2006b]. The set of attributes used in the tree has been found to be minimal by 'val' theory [Rao, 1999].

A threshold based discretization method has been proposed and adopted in this thesis [Ramadevi, 2006b]. Continuous data can be discretized using any available discretization algorithms like BROrthogonal Scalar [Han, 2001; Son, 1997; Witten, 2000], Boolean reasoning [Hoa, 1996; Ohrn, 1999] etc. A hybridized approach uses Rough Set theory for dimensional reduction [Minz, 2005]. C4.5 decision tree algorithm is applied to derive a reduced decision tree; it is a loosely coupled hybridization and uses data discretization method specified in Rosetta Software. In the present work, the proposed Reduct based Decision Tree is tightly coupled with Rough Set Theory [Ramadevi, 2007a]. Reduct Computation Algorithm has been used for finding the predominant attributes and use these attributes in Rank order for the construction of decision tree.

4.2.1 The RDT Algorithm

Reduct based Decision Tree algorithm mainly consists of two important steps: (i) Reduct Computation (ii) Decision Tree Construction.

4.2.1.1 Reduct Computation Algorithm (RCA)

The Reduct Computation Algorithm generates a set of Predominant Attributes (PA) as described in Algorithm 4.1.

Algorithm 4.1:

Algorithm RCA (Decision Table)

Input: decision table

Output: Predominant Attributes (Approximate Reduct)

1. Read the decision table with the first column as decision column.
2. Sort the rows in the ascending order of decision attribute.
3. Initialize the set of PA to NULL.
4. Construct a Boolean Table (BT), as explained in step 5, by generating a row for each pair of rows having different values for the decision attribute.
5. Construct a row with '1' and '0'. Assign a '1' to an element if the corresponding independent attribute values are different, otherwise assign '0'.
6. The columns of BT are then summed up. The column with maximum sum is picked up as a PA and appended to set of PA.
7. Take the result BT and remove those rows from it with '1' in the column corresponding to the PA selected in step 6, resulting in a reduced BT.
8. Steps 6 and 7 are repeated until

   The sum of columns of the reduced BT is '0' s, which means to say, that sufficient information is not available regarding the system to achieve crisp discrimination.

   OR

   The reduced BT is a null matrix, meaning that the set of PA obtained produces crisp rule set for discrimination.
9. These set of PA are grouped together and referred to as 'reduct'.

Most of the time the set of PA is a reduct or close to the reduct. The order in which the Predominant Attributes have been picked is maintained in predominant attribute set and referred as Rank order..

4.2.1.2 RDT Construction

Decision Rules are generated based on the decision tree constructed using PA from Algorithm 4.1.

Algorithm 4.2:

Algorithm RDT

Input: Training Dataset T1

Output: Decision Rules

1. Input the training dataset T1.
2. Discretize the continuous attributes, if any, and label the modified dataset as T2.

3. Compute PA of dataset T2 by using RCA (Algorithm 4.1).
4. Reduce dataset T2 based on PA and label reduced dataset as T3.
5. Construct decision tree on dataset T3 with PA, taking an attribute with i^{th} Rank order for splitting the i^{th} level current node (where i = 1 to number of PA).
6. Generate the decision rules by traversing all the paths from the root to the leaf node in the decision tree. These rules are called Reduct based Decision Tree Rules and they are stored in Reduct based Decision Tree Rules Database.

4.2.1.3 Complexity of RDT

Reduct based Decision Tree consists of two steps mainly Reduct Computation and decision tree construction. Complexity of the Reduct based Decision Tree depends on the complexity of Reduct Computation Algorithm and decision tree construction algorithm.

The computational cost of preprocessing the training data with 'n' instances and 'm' attributes using Reduct Computation Algorithm is $O(mn^2)$. The complexity of a decision tree depends upon the number of predominant attributes 'k' and maximum number of distinct values 'v' it can take. The complexity of decision tree is O(mv) (i.e., when k=m). In the experimentation 'v' value is '2' (predominant attribute values are considered to be binary). Therefore, the complexity of Reduct based Decision Tree algorithm is $O(mn^2+2m)$, which is equal to $O(mn^2)$.

4.3 IMPLEMENTATION OF RCA

The dataset (Table 2.5) introduced by Pawlak [Pawlak, 1982] considered benchmark by several authors for demonstrating Rough Sets concepts has been used for implementation of Reduct Computation Algorithm.

Step 1: Consider the dataset presented in Table 2.5, there are 7 objects with {a,b,c,d} as attributes and {E} as decision attribute.

Step 2 of RCA: Table 2.5 is sorted based on E(decision attribute) and the result is Table 4.3.

Table 4.3: Sorted data of Table 2.5 based on E

	E	a	b	c	d
X4	0	1	2	2	1
X1	1	1	0	2	1
X2	1	1	0	2	0
X7	1	2	1	2	1
X3	2	1	2	0	0
X5	2	2	1	0	0
X6	2	2	1	1	0

Step 3 and Step 4 of RCA results in Table 4.4

Table 4.4: Generation of Boolean Table

	a	b	c	d
4,1	0	1	0	0
4,2	0	1	0	1
4,7	1	1	0	0
1,3	0	1	1	1
1,5	1	1	1	1
1,6	1	1	1	1
2,3	0	1	1	0
2,5	1	1	1	0
2,6	1	1	1	0
7,3	1	1	1	1
7,5	0	0	1	1
7,6	0	0	1	1
Sum	6	**10**	9	7

Attribute ‘b’ has maximum value. Hence, it is the first predominant attribute to be selected. Delete the rows in which ‘b’ has value ‘1’ and repeat the process. The results are shown in Table 4.5.

Table 4.5: Final iteration of Reduct Computation Algorithm

	a	**b**	**c**	**d**
7,5	0	0	1	1
7,6	0	0	1	1
Sum	0	0	**2**	**2**

Next ‘c’ and ‘d’ have the maximum value; therefore next predominant attribute can be ‘c’ or ‘d’. For next iteration, the matrix will contain all zeros. Therefore the predominant attributes are {b,c} or {b,d}

4.3.1 Implementation of RDT

Step 1 & 2: Table 2.5 is the discretized dataset considered.

Step 3: The PA are {b,c} or {b,d}

Step 4: The reduced matrices based on the predominant attributes are as in Table 4.6 and Table 4.7 respectively.

Table 4.6: Reduced table with predominant attributes {b, c} only

	E	**b**	**c**
X1	1	0	2
X2	1	0	2
X3	2	2	0
X4	0	2	2
X5	2	1	0
X6	2	1	1
X7	1	1	2

Step 5: Decision trees are drawn taking PA in Rank order for splitting and corresponding decision trees are as in Figure 4.1 and Figure 4.2.

Step 6: Reduct based Decision Tree Rules are generated by traversing all the paths of the tree from the root to the leaf node and stored in Reduct based Decision Tree Rules Database.

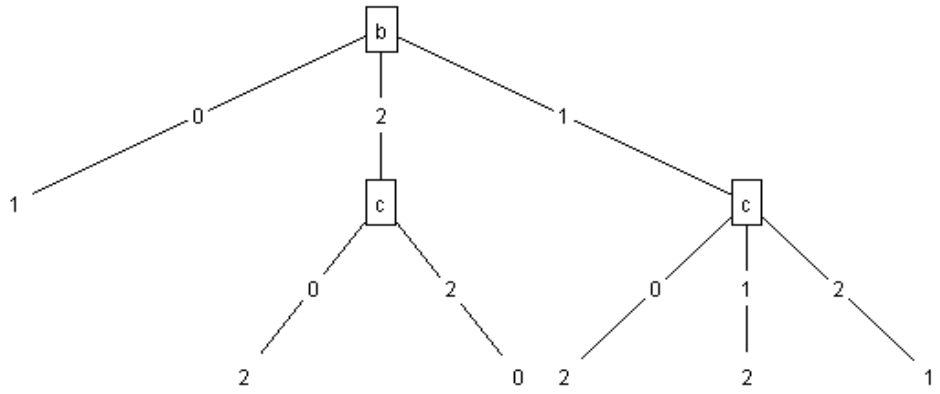

Figure 4.1: Decision Tree for Table 4.6

Table 4.7: Reduced table with predominant attributes {b, d} only

	E	b	d
X1	1	0	1
X2	1	0	0
X3	2	2	0
X4	0	2	1
X5	2	1	0
X6	2	1	0
X7	1	1	1

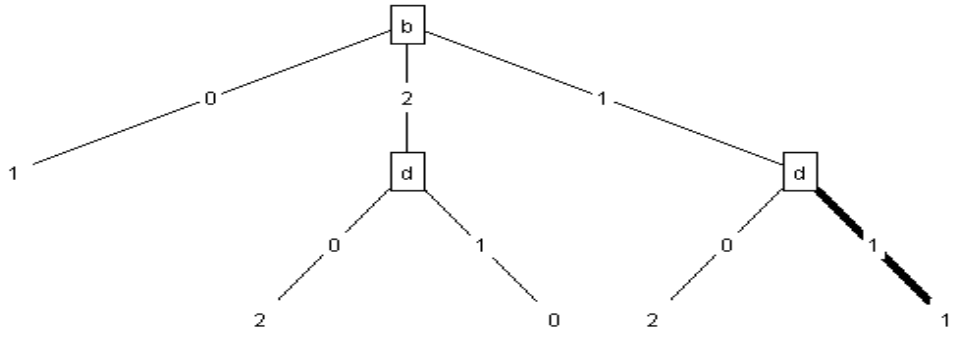

Figure 4.2: Decision Tree for Table 4.7

Reduct based Decision Tree Rules for decision tree in Figure 4.1

1. If ‘b’ is 0 then class =‘1’.
2. If ‘b’ is 1 and ‘d’ is 0 then class = ‘2’.
3. If ‘b’ is 1 and ‘d’ is 1 then class = ‘2’.
4. If ‘b’ is 1 and ‘d’ is 2 then class = ‘0’.
5. If ‘b’ is 2 and ‘d’ is 0 then class = ‘2’.
6. If ‘b’ is 2 and ‘d’ is 2 then class = ‘0’.

Reduct based Decision Tree Rules for decision tree in Figure 4.2

1. If ‘b’ is 0 then class = ‘1’.
2. If ‘b’ is 1 and ‘d’ is 0 then class = ‘2’.
3. If ‘b’ is 1 and ‘d’ is 1 then class = ‘1’.
4. If ‘b’ is 2 and ‘d’ is 0 then class = ‘2’.
5. If ‘b’ is 2 and ‘d’ is 1 then class = ‘0’.

It is observed that the performance of decision tree arrived based on a reduct is invariant of the selection of reducts. The efficiency of the algorithm with multiple reducts has been found to be same in terms of Kappa statistics.

4.4 CREATION OF REDUCT BASED DECISION TREE RULES DATABASE (RDTRD)

Reduct based Decision Tree algorithm has been applied to A-Table, D-Table and P-Table separately. Decision rules have been generated by traversing all paths from the root to the leaf node, they are called Reduct based Decision Tree Rules. These rules have been stored in a database called Reduct based Decision Tree Rules Database (Appendix B) along with class and family information.

4.5 ILLUSTRATION OF RDT

Algorithm CIdentifier (Algorithm 4.3) is used to obtain class information within a family for a given unknown sequence ‘y’. From the Sequence Arithmetic module, A(y) is generated and it is used to obtain class information based on Rules stored in Reduct based Decision Tree Rules Database (Appendix B).

Algorithm 4.3:

Algorithm CIdentifier

Input: A(y) // A-Set for the unknown Sequence

Output: A class or set of classes

1. Construct a Boolean vector: for each predominant P-Set assign '1' if input A-Set is superset of predominant P-Set, otherwise assign '0'.
2. Construct a query using this Boolean vector and administrate this query on Reduct based Decision Tree Rules Database.
3. Return class/classes with the family of the protein.

Similarly decision tree for D-Set can also be obtained and used for extracting class information.

Example 4.3:

Consider an unknown sequence 'y'

GLSDGEWQLVLHVWGKVEADLAGHGQEVLRLFKGHPETLEKFNKFKHKSE
DEMKASEDLKKHGVTVLTALGGVLKKKGHHEAEPLAQSHATKHKPKLEFSE
AHVLQSKHPGFGADAGAMNKALELFRKDAAKKELGFQG

A(y) = {A,D,E,F,G,H, K,L,M,N,P,Q,R,S,T,V,W }

P(Myoglobin) = { }

P(GPCR) = {A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T ,V,W,Y}

Since P(Myoglobin) $\subseteq$ A(y) , 'y' belongs to Myoglobin family(Chapter III). CIdentifier algorithm with A(y) as input, and gives 'y' belongs to Class 2 as output (Appendix B1).

4.6 TESTING AND ANALYSIS

The performance evaluation of the Reduct based Decision Tree algorithm with the existing classification techniques (WEKA tool) on benchmark databases has been performed and the results are shown in Table 4.8. It has been observed that RDT performs better than other methods in terms of Kappa statistics [Ramadevi, 2008a].

Table 4.8: Comparison of RDT with other Classification Techniques

Classification Technique	Kappa Statistics	
	Sunburn	**Weather**
Bayes Network	0	0
ComplementNaiveBayes	0.142	0
NaiveBayesMultinomial	0	0
Logistic	0	0.588
RBFNetwork	0	0.256
SimpleLogistic	0	0
SMO	0	0
BFTree	0	0
J48	0	0.143
J48graft	0	0.143
LMT	0	0
NBTree	0	0
RandomForest	0	0.429
RandomTree	0	0.378
SimpleCart	0	0
RDT	0.5	0.58

Reduct based Decision Tree Rules have been generated for the known sequences from the decision trees. They have been stored in the Reduct based Decision Tree Rules Database.

4.7 REDUCT BASED DECISION TREE MODULE

Given an unknown sequence ‘y’, family information < F > is obtained from Sequence Arithmetic module (Section 3.4.1).

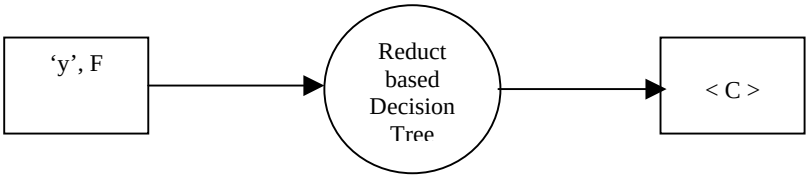

Figure 4.3: Reduct based Decision Tree Module

A(y) and < F > are given as inputs to the RDT module (Figure 4.3) which gives the set of class/classes < C > within the family to which 'y' belongs as output.

The sub processes of the Reduct based Decision Tree is as in Figure 4.4. The family information has been obtained for queried sequence 'y'. A(y) is mapped for superset with the distinct P-Sets and vector is obtained. This vector has been mapped with the rules present in the database and class information < C > is given.

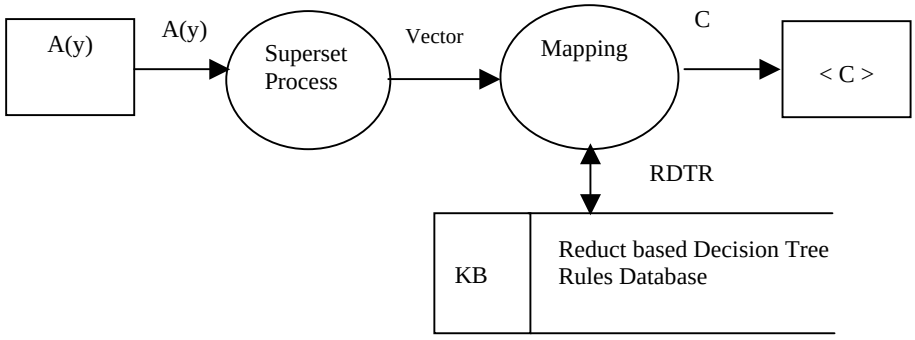

Figure 4.4: Internal sub process of Reduct based Decision Tree Module

The limitation of the Reduct based Decision Tree module is that decision tables shown in Tables 4.2a, 4.2b and 4.2c cannot be given as input to Reduct based Decision Tree as the discriminating power is lost due to same row values. Therefore localized information has to be extracted for further sub-classification. A new technique, neighborhood analysis has been developed in the present work, for the extraction of localized information in Chapter V.

4.8 SUMMARY

A predominant attributes set extraction method has been developed for a given decision table. Based on PA set, a decision tree has been constructed. The rules have been generated from the decision tree and organized in a database namely Reduct based Decision Tree Rules Database. These rules are used for class identification of the given protein using CIdentifier Algorithm (Algorithm 4.3).

The complexity of the Reduct Computation Algorithm is $O(mn^2)$ and the complexity of generating the decision tree is O(2m). Therefore, the overall complexity of the RDT is $O(mn^2)$.

The Reduct based Decision Tree has been derived for various reducts and the performance evaluation has been carried out. It is observed that the performance of decision tree arrived based on a reduct is invariant of the selection of reducts. The efficiency of the algorithm with multiple reducts is found to be same in terms of Kappa statistics.

As the search is confined to a class within a family, the search space has been reduced to 36% (Appendix D).

The infirmity of this method is that rules cannot be generated when the decision table has same value for all its entries. Hence, the need for neighborhood association arises, which has been discussed in the Chapter to follow.

CHAPTER V
NEIGHBORHOOD ASSOCIATIONS

A sequence has rich information both globally as well as locally. Chapters III and IV were elaborated based on global information. It has been observed that this global information is inadequate for critical analysis. This chapter emphasizes on building a type of local information known as neighborhood associations and explores the potential characteristics of it for protein classification.

The classification of a given protein into set of class / classes has not been possible for families like GPCR by confining to Sequence Arithmetic as discussed in earlier chapters. This chapter introduces a new concept called 'Neighborhood Associations' / 'Neighborhood Analysis', which solicits a type of spatial information of the sequence. A Binary Association Matrix is developed for adapting Rough Set Theory using these neighborhood associations [Ramadevi, 2007b].

Neighborhood analysis is introduced in Section 5.1. Creation of database is presented in Section 5.2. Demonstration details are given in Sections 5.3 and 5.4.

5.1 NEIGHBORHOOD ANALYSIS

Neighborhood Associations, Neighborhood Matrix and Binary Association Matrix computation with implementation are introduced in this section.

5.1.1 Neighborhood Association

Proteins are sequences made up of amino acids (treated as characters). Observations have been made by calculating the number of occurrences of a character Y in the surroundings of the character X within a distance 'd', which has been used in the generation of neighborhood matrix. If X is positioned at the extreme ends of the sequence, the number of neighbors will be restricted to 'd' only. If X is at the middle, it can have atmost '2d' neighbors (both side 'd' neighbors), thus X's neighbors are varying from 'd' to '2d'.

The number of times Y occurs in the neighbors varies from X to Y when X is not equal to Y.

5.1.2 Neighborhood Matrix (NM)

Let X and Y be two characters whose association has to be found, X may occur at different positions in the sequence and given ‘d’ the distance, there will be at least ‘d’ neighbors and utmost ‘2d’ neighbors to X in which Y occurs.

The search for the occurrence of Y in the neighborhood of X at a distance less than or equal to ‘d’ in a sequence ‘s’ is computed as in Algorithm 5.1.

Let ‘n’ be size of the string and ‘i’ take the values from 1 to n.

$\delta_x^i(s)$ = 1 i^{th} character is ‘x’ in the String ‘s’.

= 0 Otherwise. (5.1)

$$T(X) = \sum_{i=1}^{n} \delta_x^i(s) \quad (5.2)$$

$$C_i(X,Y,d) = \sum_{\substack{j=-d \\ j\neq 0}}^{d} \delta_X^i(s)\,\delta_Y^{i+j}(s) \quad (5.3)$$

$$L_i(X, d) = M(i) - m(i) \quad (5.4)$$

Where M(i) = Min(i+d, n)

m(i) = Max(i-d,1)

$$N_i(X, Y, d) = C_i(X,Y,d) \;/\; L_i(X, d) \quad (5.5)$$

$$N(X, Y, d) = \frac{\sum_{i=1}^{T(x)} N_i(X, Y, d))}{T(X)} \quad \forall \text{ i.} \quad (5.6)$$

Algorithm 5.1:

Algorithm NM(s,d)

Input: Distance ‘d’, Sequence ‘s’

Output: NM

1. For each centre character X
2. Repeat Step 3 and Step 4 for each character Y
3. Compute $N_i(X, Y, d)$ for i = 1 to n (size of ‘s’)
4. Compute N (X, Y, d)

Computation of Neighborhood Associations is shown below:

Example 5.1:

Consider the sequence‘s’ (belongs to class B)

MRFTFTRQFLAFFILISNPASILPRSENLTFPTFEPEPYLYSVGRKKLVDAQYRC
YDRMQQLPPYEGEGPYCNRTWDGWMCWDDTPAGVLSVQLCPDYFPDFDPT
EKVTKYCDESGVWFKHPENNRTWSNYTLCNAFTPEKLQNAYVLYYLAIVGH
SMSIITLVVSLGIFVYFRSLGCQRVTLHKNMFLTYILNSMIIIIHLVEVVPNGEL
VRKDPVSCKILFFHQYMMACNYFWMLCEGIYLHTLIVVSVFNEAKHLRWYY
LLGWGFPLVPTTIHAITRALYFNDNCWISVDTHLLYIIHGPVMVALVVNFFFLL
NIVRVLVTKMRETHEAESYMYLKAVKATMILVPLLGIQFVVFPWRPSNKVLG
KIYDYFMHSLIHFQGFFVATIYCFCNNEVQTTLKRQWAQFKIQWNQRWGTRP
SNRSAAARAAAAAAEAGGDNIPVYICHQEPRNDPPNNQGEEGAEMIVLNIIEK
ESS

The frequency of occurrence of character in the sequence ‘s’ is given in Table 5.1 and ‘W’ has been observed to be Least Frequent occurring Character.

Table 5.1: Frequency Table for the given Sequence‘s’

Character	Frequency	Character	Frequency
A	28	M	15
C	15	N	27
D	15	P	27
E	24	Q	17
F	31	R	21
G	22	S	22
H	14	T	26
I	32	V	38
K	18	**W**	**13**
L	44	Y	27

Let A be taken as center, then occurrence of A in its neighborhood at distance d =1 is N (A, A, 1) = 0.2413793.

Similarly for different characters, the N values are as follows (Table 5.2):

Table 5.2: Neighborhood Associations for different characters with A as center and distance 'd'

N (A, A, 1) = 0.2413793	N (A, M, 1)= 0.01724138
N (A, C, 1) = 0.01724138	N (A, N, 1) = 0.03448276
N (A, D, 1) = 0.01724138	N (A, P, 1) = 0.03448276
N (A, E, 1) = 0.10344828	N (A, Q, 1) = 0.03448276
N (A, F, 1) = 0.03448276	N (A, R, 1) = 0.05172414
N (A, G, 1) = 0.05172414	N (A, S, 1) = 0.06896552
N (A, H, 1) = 0.01724138	N (A, T, 1) = 0.03448276
N (A, I, 1) = 0.03448276	N (A, V, 1) = 0.05172414
N (A, K, 1) = 0.05172414	N (A, W, 1) = 0.01724138
N (A, L, 1) = 0.06896552	N (A, Y, 1) = 0.01724138

In the similar manner the N (X, Y, d) are computed where

$X \in \{A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y\}$,

$Y \in \{A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y\}$ and 'd' is less than 1,2,3,4,5 and 6 .

N(X, Y, 1) for different X and Y values for the above sequence which belongs to class B is shown in Table 5.3

Table 5.3: Neighborhood Matrix (NM) for distance 'd'=1

	A	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**
A	.24	.02	.02	.1	.03	.05	.02	.03	.05	.07	.02	.03	.03	.03	.05	.07	.03	.05	.02	.02
C	.04	0	.04	.04	.03	.03	.03	.05	.03	.06	.04	.07	.06	.05	.04	.04	.06	.06	.05	.1
D	.03	.03	.07	.03	.07	.06	0	0	.03	0	0	.13	.17	0	.03	0	.07	.07	.07	.13
E	.12	.02	.02	.04	.07	.07	0	0	.08	.02	.02	.08	.13	.02	.02	.08	.04	.06	0	.02
F	.03	.03	.03	.02	.17	.03	.03	.03	.03	.06	.03	.05	.06	.06	.03	0	.09	.08	.03	.08
G	.07	.02	.05	.15	.05	.05	.05	.07	.02	.11	0	.02	.04	.04	.02	.02	.02	.09	.09	0
H	.04	.04	0	.04	.07	.07	0	.14	.1	.1	.03	0	.03	.07	0	.07	.01	0	0	0
I	.03	.01	0	.02	.03	.05	.06	.19	.04	.12	.04	.04	.01	.03	0	.06	.06	.06	.01	.09
K	.08	.03	.03	.11	.06	.03	.08	.08	.05	.11	.02	.05	0	0	.08	0	.05	.08	0	.02
L	.05	.03	0	.01	.05	.06	.06	.09	.04	.09	.01	.04	.04	.03	.01	.04	.07	.15	0	.10
M	.03	.03	0	.03	.07	0	.03	0.1	.03	.03	.06	.03	0	.03	.13	.1	.03	.06	.06	.1
N	.04	.09	.07	.07	.06	.02	0	.06	.03	.07	.02	.11	.05	.05	.07	.09	0	.02	.02	.03
P	.04	.02	.09	.11	.07	.04	.02	.02	0	.07	0	.05	.07	0	.07	.04	.09	.11	.02	.05
Q	.06	.03	0	.03	.11	.06	.06	.06	0	.09	.03	.09	0	.06	.12	0	.03	.06	.06	.06

R	.07	.02	.02	.02	.04	.02	0	0	.06	.02	.06	.09	.09	.09	0	.06	.11	.09	.06	.02
S	.07	.02	0	.09	0	.02	.05	.09	0	.09	.06	.11	.05	0	.06	.04	0	.15	.02	.05
T	.04	0	.04	.04	.11	.02	.06	0	.04	.13	.02	0	.09	.02	.09	0	.07	.05	.04	.04
V	.04	0	.03	.04	.07	.05	0	0	.04	.17	.01	.01	.08	.03	.05	.09	.04	.13	.01	.04
W	.04	.08	.08	0	.08	.15	0	0	0	0	.07	.04	.04	.07	.11	.04	.07	.03	0	.4
Y	.02	.07	.07	.02	.09	0	0	0	.02	.17	.05	.04	.05	.04	.02	.04	.04	.05	.02	.09

The Neighborhood matrix all for characters with centers S, M (at extreme ends) and W (Least Frequent occurring Character) at a distance ‘d’ = 2 is shown in Tables 5.4a, 5.4b, 5.4c.

Table 5.4a: Neighborhood Associations with centers at S(Last character).

N(S,A,2)	0.045	N(S,M,2)	0.068
N(S,C,2)	0.011	N(S,N,2)	0.068
N(S,D,2)	0.023	N(S,P,2)	0.057
N(S,E,2)	0.083	N(S,Q,2)	0.011
N(S,F,2)	0.034	N(S,R,2)	0.045
N(S,G,2)	0.057	N(S,S,2)	0.061
N(S,H,2)	0.023	N(S,T,2)	0.011
N(S,I,2)	0.068	N(S,V,2)	0.114
N(S,K,2)	0.038	N(S,W,2)	0.023
N(S,L,2)	0.125	N(S,Y,2)	0.034

Table 5.4b: Neighborhood Associations with centers at M(First character).

N(M,A,2)	0.083	N(M,M,2)	0.033
N(M,C,2)	0.05	N(M,N,2)	0.033
N(M,D,2)	0.017	N(M,P,2)	0.017
N(M,E,2)	0.033	N(M,Q,2)	0.05
N(M,F,2)	0.083	N(M,R,2)	0.067
N(M,G,2)	0.017	N(M,S,2)	0.083
N(M,H,2)	0.033	N(M,T,2)	0.033
N(M,I,2)	0.083	N(M,V,2)	0.05
N(M,K,2)	0.033	N(M,W,2)	0.05
N(M,L,2)	0.067	N(M,Y,2)	0.083

`Table 5.4c: Neighborhood Associations with centers at W(LFC).

N(W,A,2)	0.00	N(W,M,2)	0.06
N(W,C,2)	0.06	N(W,N,2)	0.06
N(W,D,2)	0.08	N(W,P,2)	0.06
N(W,E,2)	0.00	N(W,Q,2)	0.06
N(W,F,2)	0.10	N(W,R,2)	0.13
N(W,G,2)	0.12	N(W,S,2)	0.04
N(W,H,2)	0.00	N(W,T,2)	0.06
N(W,I,2)	0.04	N(W,V,2)	0.02
N(W,K,2)	0.02	N(W,W,2)	0.00
N(W,L,2)	0.06	N(W,Y,2)	0.06

5.1.3 Binary Association Matrix

The Binary Association Matrix has been obtained by applying a threshold value 'T' on the Neighborhood Matrix, thus binarizing it to obtain a binary matrix. Algorithm 5.2 describes the process of conversion of Neighborhood Matrix to Binary Association Matrix with distance 'd' (BAMD) and threshold 'T'.

Algorithm 5.2:

Algorithm BAMD (NM,d,T)

Input : NM, Threshold T, distance 'd'

Output : BAMD

1. Obtain the Neighborhood Matrix
2. for each entry in the NM,

   if NM(X, Y, d) $\geq$T then A (X, Y, d) $\leftarrow$ 1

   else A (X,Y, d) $\leftarrow$ 0
3. BAMD $\leftarrow$ A

5.1.4 Threshold

The binarization of Neighborhood Matrix to obtain Binary Association Matrix with distance 'd' is sensitive with respect to threshold 'T'. Choosing 'T' is important, as it should be neither too large nor too small. The process of selection of 'T' is discussed in Section 5.5.

The Neighborhood Matrix (Table 5.2) is used and Binary Association Matrix has been obtained by taking different Threshold as shown in Table 5.5**.**

Table 5.5: BAM for Distance d=1 for different Threshold

S.No	NM	BAM(T=0.05)	BAM (T=0.06)
1	N (A, A,1) = 0.24	1	1
2	N (A, C,1) = 0.02	0	0
3	N (A, D,1)= 0.02	0	0
4	N (A, E,1)= 0.10	1	1
5	N (A, F,1)= 0.03	0	0
6	N (A, G,1)= 0.05	1	0
7	N (A, H,1)= 0.02	0	0
8	N (A, I,1)= 0.03	0	0
9	N (A, K,1)= 0.05	1	0
10	N (A, L,1)= 0.07	1	1
11	N (A, M,1)= 0.02	0	0
12	N (A, N,1)= 0.03	0	0
13	N (A, P,1)= 0.03	0	0
14	N (A, Q,1)= 0.03	0	0
15	N (A, R,1)= 0.05	1	0
16	N (A, S,1)= 0.07	1	1
17	N (A, T,1)= 0.03	0	0
18	N (A, V,1)= 0.05	1	0
19	N (A, W,1)= 0.02	0	0
20	N (A, Y,1)= 0.02	0	0

5.1.5 Generation of Binary Association Matrix

Binary Association Matrix for the Neighborhood Matrix with threshold ‘T’ and distance ‘d’ ≤ 5 is constructed by disjunction of the BAMD (NM, d, T).

BAM = $\vee$ (BAMD (NM, d, T)) for d=1, 2, 3, 4and 5.

5.1.6 Generation of Predominant Attributes (PA)

The Predominant Attributes are generated by considering the Binary Association Matrix developed in Section 5.1.5 and taking X as Least Frequent occurring Character as in Algorithm 5.3

Algorithm 5.3:

Algorithm RDT (BAM, X)

Input: Set of sequences, X, T

Output: PA(X), RDT(X)

1. BAM(X) ← BAM.
2. Concatenate classID to respective rows in BAM(X) as 1st column.
3. Perform RCA, Obtain PA(X).
4. Construct reduced Decision Table(X).
5. Obtain RDT(X) by applying RDT for the obtained in Step 8.

5.2 CREATION OF NEIGHBORHOOD ASSOCIATION RULES DATABASE

Decision tables have been constructed by concatenating class information to the Binary Association Matrix. Reduct based Decision Tree is constructed by implementing Algorithm 5.3 on the decision table obtained. Decision rules are generated by traversing the decision tree from root to all the leaves. The following Algorithm 5.4 has been designed for generating rules based on neighborhood analysis named as Neighborhood Association Rules and stored in database known as Neighborhood Association Rules Database (NARD). The rules are similar to RDT rules (Appendix B2).

Algorithm 5.4:

Algorithm NARD (D)

Input: Database D, Threshold T

Output: NARD

1. Initialize NARD ← NULL.
2. Repeat Step 3 to Step 8 for each selected center character X.
3. Repeat Step 4 to Step 8 for each sequence ‘s’ of D.
4. Initialize distance ‘d’ ←1.
5. Initialize BAM ← NULL.
6. while d ≤ 5 do

{ NM ← NM(s, d)

BAM ← ∨ (BAM , BAMD (NM, d, T)).

d ← d+1

}

7. NMRules ← RDT (BAM, X).
8. Append (NMRules, NARD).

5.3 IDENTIFICATION OF CLASS

The methodology developed in the earlier sections will be used for the purpose of extraction of class information of a given protein whose family is already known. Algorithm 5.5 discusses the process of class identification with Least Frequent occurring Character as center.

Algorithm 5.5:

Algorithm SFIdentifier(y,F)

Input: Unknown sequence 'y', Family <F>

Output: Class information C.

1. Center ← Least Frequent occurring Character in the sequence 'y'.
2. Generate Binary Association Matrix for the 'y' with center obtained in Step 1.
3. Extract data from the Binary Association Matrix pertaining to Predominant Attributes for this family with center as in Step 1.
4. Identify 'C' by submitting the data obtained in Step 3 to the corresponding tree.

5.4 DEMONSTRATION AND RESULTS

A-Set, D-Set and P-set have been found to be same for each of the 5 classes in the GPCR dataset (section 4.1.2), and moreover A-Set and P-Set are equal to Σ = {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}. The decision table generated for all the 5 classes with the sets has equal entries, which cannot be given as input to the Reduct based Decision Tree module. Therefore, spatial information has been extracted for the dataset and then Binary Association Module with Least Frequent occurring Character has been subjected to Reduct based Decision Tree to get class information.

5.4.1 GPCR Analysis

Information extraction and class identification process is illustrated in the following section.

5.4.1.1 Neighborhood Matrix (NM)

Table 5.6 illustrates the generation of Neighborhood Matrix for a sample dataset of GPCR and Table 5.7 gives binarized matrix of Table 5.6.

Table 5.6: Training set with character W as center (LFC)

A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
0.00	0.19	0.00	0.00	0.06	0.13	0.00	0.13	0.00	0.19	0.06	0.00	0.00	0.00	0.13	0.00	0.06	0.06	0.00	0.00
0.05	0.25	0.00	0.00	0.10	0.00	0.00	0.10	0.00	0.10	0.05	0.05	0.00	0.00	0.10	0.00	0.05	0.15	0.00	0.00
0.10	0.25	0.00	0.00	0.10	0.05	0.00	0.10	0.00	0.10	0.05	0.05	0.00	0.00	0.10	0.00	0.05	0.05	0.00	0.00
0.00	0.28	0.00	0.00	0.00	0.06	0.00	0.17	0.00	0.11	0.06	0.06	0.00	0.00	0.06	0.00	0.11	0.06	0.00	0.06
0.07	0.07	0.00	0.00	0.00	0.00	0.00	0.14	0.00	0.00	0.07	0.07	0.14	0.07	0.00	0.00	0.00	0.36	0.00	0.00
0.00	0.00	0.06	0.00	0.00	0.11	0.00	0.06	0.06	0.11	0.00	0.00	0.06	0.11	0.06	0.11	0.11	0.11	0.00	0.06
0.13	0.06	0.00	0.00	0.00	0.00	0.06	0.06	0.00	0.19	0.13	0.06	0.06	0.00	0.00	0.06	0.06	0.13	0.00	0.00
0.21	0.07	0.00	0.07	0.07	0.07	0.00	0.00	0.07	0.14	0.00	0.00	0.00	0.00	0.07	0.00	0.00	0.14	0.00	0.07
0.07	0.03	0.04	0.11	0.00	0.09	0.01	0.03	0.05	0.08	0.01	0.03	0.05	0.04	0.05	0.18	0.04	0.07	0.00	0.01
0.11	0.02	0.06	0.07	0.02	0.15	0.04	0.06	0.02	0.09	0.00	0.04	0.06	0.02	0.04	0.09	0.04	0.00	0.07	0.02
0.04	0.08	0.08	0.00	0.08	0.15	0.00	0.04	0.00	0.00	0.08	0.04	0.04	0.08	0.12	0.04	0.08	0.04	0.00	0.04
0.04	0.07	0.07	0.00	0.07	0.14	0.00	0.00	0.04	0.07	0.04	0.07	0.04	0.07	0.11	0.04	0.07	0.04	0.00	0.04
0.08	0.00	0.08	0.08	0.04	0.12	0.00	0.12	0.00	0.12	0.00	0.04	0.08	0.00	0.15	0.00	0.04	0.08	0.00	0.00
0.04	0.00	0.04	0.04	0.04	0.12	0.00	0.12	0.04	0.12	0.00	0.04	0.08	0.00	0.15	0.04	0.04	0.12	0.00	0.00
0.08	0.00	0.08	0.08	0.04	0.12	0.00	0.12	0.00	0.12	0.00	0.04	0.08	0.00	0.15	0.00	0.04	0.08	0.00	0.00
0.07	0.00	0.03	0.00	0.10	0.03	0.03	0.10	0.03	0.03	0.03	0.10	0.03	0.07	0.03	0.07	0.07	0.17	0.00	0.00
0.05	0.05	0.10	0.00	0.00	0.05	0.05	0.10	0.00	0.15	0.10	0.00	0.00	0.00	0.05	0.10	0.10	0.10	0.00	0.00
0.04	0.04	0.04	0.00	0.00	0.04	0.04	0.21	0.04	0.13	0.08	0.04	0.00	0.00	0.04	0.13	0.13	0.00	0.00	0.00
0.04	0.04	0.04	0.00	0.00	0.04	0.04	0.21	0.04	0.13	0.08	0.04	0.00	0.00	0.04	0.13	0.13	0.00	0.00	0.00
0.04	0.04	0.04	0.00	0.00	0.04	0.04	0.21	0.04	0.13	0.08	0.04	0.00	0.00	0.04	0.13	0.13	0.00	0.00	0.00

Table 5.7: Binarized values of Table 5.6 with Threshold T=0.05

CLASS	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
1	0	1	0	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	0	0
1	0	1	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0
1	1	1	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	1	0	1	0	1	1	1	0	0	1	0	1	1	0	1
2	1	1	0	0	0	0	0	1	0	0	1	1	1	1	0	0	0	1	0	0
2	0	0	1	0	0	1	0	1	1	1	0	0	1	1	1	1	1	1	0	1
2	1	1	0	0	0	0	1	1	0	1	1	1	1	0	0	1	1	1	0	0
2	1	1	0	1	1	1	0	0	1	1	0	0	0	0	1	0	0	1	0	1
3	1	0	0	1	0	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0
3	1	0	1	1	0	1	0	1	0	1	0	0	1	0	0	1	0	0	1	0
3	0	1	1	0	1	1	0	0	0	0	1	0	0	1	1	0	1	0	0	0
3	0	1	1	0	1	1	0	0	0	1	0	1	0	1	1	0	1	0	0	0

4	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	0	1	0	0
4	0	0	0	0	0	1	0	1	0	1	0	0	1	0	1	0	0	1	0	0
4	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	0	1	0	0
4	1	0	0	0	1	0	0	1	0	0	0	1	0	1	0	1	1	1	0	0
5	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0	1	1	1	0	0
5	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	0	0	0
5	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	0	0	0
5	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	1	0	0	0

5.4.1.2 Reduct Computation

The data table (Table 5.7) has been given as input to the Reduct Computation Algorithm (section 4.2.1.1) and [P, W, F, C] was the resultant Predominant Attributes (section 4.2.1.2). Table 5.7 has been reduced to Table 5.8 with only [P, W, F, C] columns.

Table 5.8: Reduced table containing PA

CLASS	P	W	F	C
1	0	0	1	1
1	0	0	1	1
1	0	0	1	1
1	0	0	0	1
2	1	0	0	1
2	1	0	0	0
2	1	0	0	1
2	0	0	1	1
3	1	0	0	0
3	1	1	0	0
3	0	0	1	1
3	0	0	1	1
4	1	0	0	0
4	1	0	0	0
4	1	0	0	0
4	0	0	1	0
5	0	0	0	0
5	0	0	0	0
5	0	0	0	0
5	0	0	0	0

5.4.1.3 RDT Construction

The Reduct based Decision Tree constructed for Table 5.8 is as in Figure 5.1. The splitting attribute at the internal node is Predominant Attributes taken in Rank order. The leaf node indicates the value of the decision attribute. The presence of two or more classes at the leaf is indicated by a symbol '?'. In the conventional

approach local attributes are used for splitting the nodes, but in RDT Algorithm global attributes in Rank order has been used for splitting the nodes.

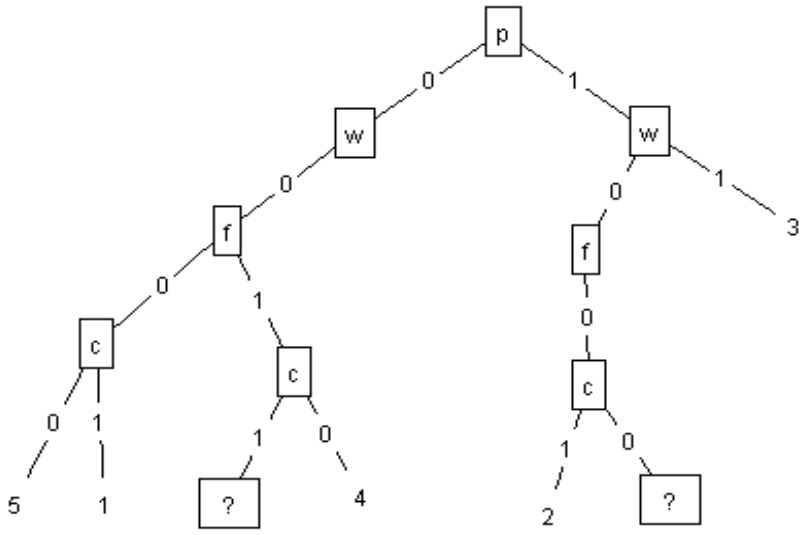

Figure 5.1: Decision Tree generated by RDT Algorithm for data in Table 5.8.

5.4.1.4 Decision Rules

Decision Rules are generated by traversing all paths of the decision tree from root to leaf.

1. If P = 0, W = 0, F = 0, C = 0 then Class = '5'.
2. If P = 0, W = 0, F = 0, C = 1 then Class = '1'.
3. If P = 0, W = 0, F = 1, C = 0 then Class = '4'.
4. If P = 1, W = 0, F = 0, C = 1 then Class = '2'.
5. If P = 1, W = 1, F = 0, C = 0 then Class = '3'.
6. If P = 0, W = 0, F = 1, C = 1 then Class = '1'or '2' or '3'.
7. If P = 1, W = 0, F = 0, C = 0 then Class = '2'or '3' or '4'.

Global attributes have been used for splitting nodes at each level. The leaf node with no decision class indicates that the information is not sufficient for taking any kind of decision i.e., class cannot be identified crisply. It has a multivalued output.

5.4.1.5 Experimental Results for the GPCR Dataset

Example 5.2:

Consider the unknown sequence 'y'

MVFLSGNASDSSNCTQPPAPVNIPKAILLGVILGVLILFGVPGNILVILSVACHR
HLHSVTHYYIVNLAVADLLLTSTVLPFSAIFEILGYWAFGRVFCNIWAAVDRL
CCTASIMSLCIISIDRYIGVSYPLRYPTIVTQRRGLRALLCLWWLTIVISIGPLFG
ARQIARQDETICQINEDPSYVLFSALGSFYVPLAIILVMYCRVYVVAKRESRGL
TSGLKTDKSDSEQVTLRIHRKNAPLGGSGVASSKNKTHFSVRLLKFSREKKAA
KTLGIVVGCFVLCWLPFFLVMPIGSFFPDFKPSETVFKIVFKLGYLNSCINPIIYP
CSSQEFKKAFQNVLKIQCLRRKQSSKHALGYTLHPPSQAVEGQHKDMVRIPV
GSRETFYKISKTDGVCEWWFFRSMPRGSARITVPKDQSACTTARVRSKSFLQV
MMCVGPSTPNPHQV

A(y) = {A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,A,T,V,W,Y}

If P(GPCR) $\subset$ A(y) , then it belongs to GPCR family (Section 3.4.1).

As it belongs to GPCR family, neighborhood association has to be found.

The frequency of all Amino Acids is given below

Centre	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	**W**	Y
frequency	27	17	12	11	25	27	10	35	24	45	8	13	26	15	26	41	21	41	**7**	14

From the frequencies, it has been observed that 'W' is LFC and it is considered as center for computing the neighborhood associations. If more than one LFC occurs, then the selection of the least frequent character has been done based on the observation for various classes and family, and priorities are assigned by polling.

BAM for sequence 'y' and threshold 'T' = 0.05

Centre	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
W	1	1	0	0	1	1	0	1	0	1	0	0	0	0	1	0	0	1	1	0

Based on the BAM values and RDTR for the GPCR dataset, the outcome of the rule is that the unknown sequence 'y' belongs to class E (class 5). (Appendix B1)

5.5 ANALYSIS

The process of fixing the values for threshold ‘T’ and distance ‘d’ along with the performance analysis is discussed in this section.

5.5.1 Threshold Fixation

For a given center, for a fixed distance ‘d’, for varying thresholds ‘T’ (0.01 to 1.0 in steps of 0.01) has been used for binarization. The distribution of ‘0’ and ‘1’ among all the alphabets is derived. The entropy of this distribution is obtained. For a particular distance d=1 with center A the entropy variation is shown below (Figure 5.2):

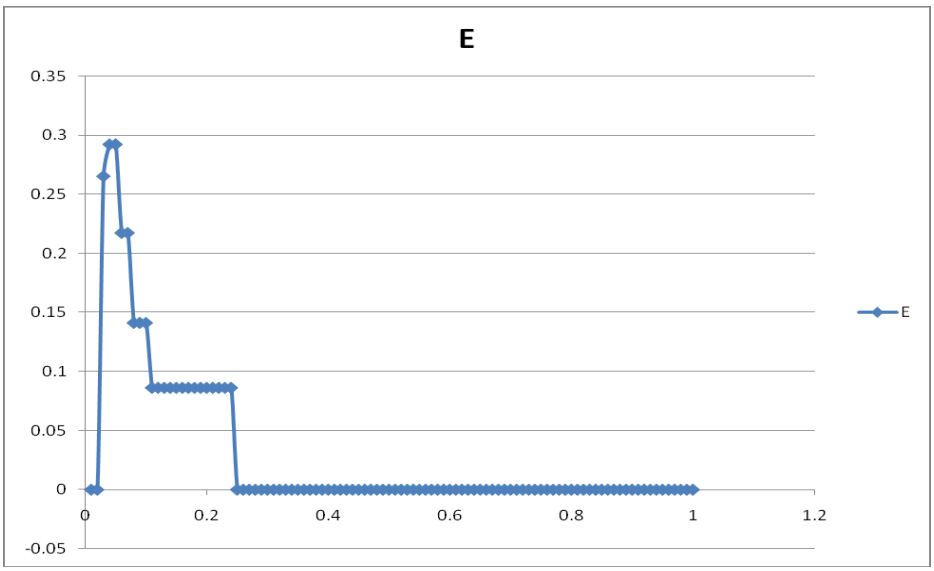

Figure 5.2: Entropy variation for distance d=1 and center is ‘W’

The entropy has been monotonically increasing and then decreasing. It has been observed that the maximum entropy is achieved when ‘T’ values are 0.04 and 0.05. Comparing the entropies for various centers at these threshold values, the maximum entropy is obtained at T = 0.05. Entropy obtained for threshold values 0.04 and 0.05 at distance ‘d’ = 1 is depicted in Figure 5.3. It is observed that for least frequent occurring characters (in ascending order) {H,M,Q,W}, the entropy for threshold 0.05 is dominating. Hence, threshold is frozen at 0.05 for binarization. Similarly for different centers, entropy variations have been observed.

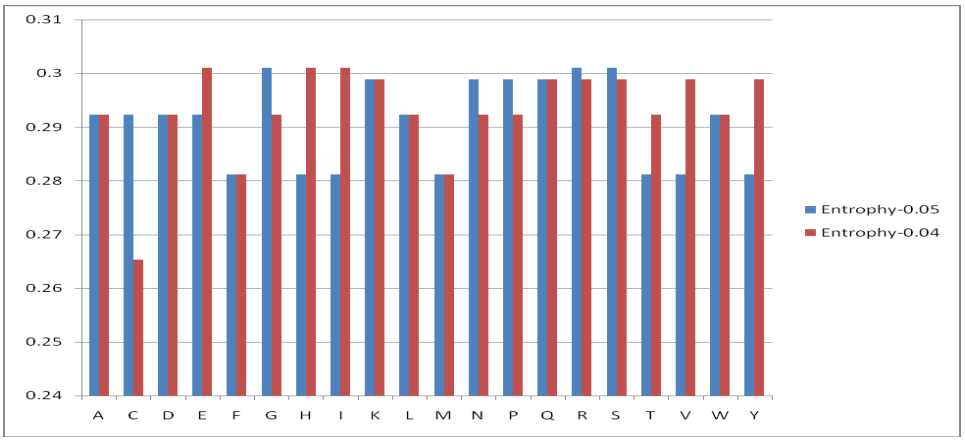

Figure 5.3: Entropy distribution for all characters at distance d=1

5.5.2 Impact of Neighborhood Distance

The neighborhood associations have been carried out for distance d = 1, 2, 3, 4, 5 and 6. The corresponding Binary Association Matrices have been constructed. Rules are obtained for Binary Association Matrices with different centers for each 'd'. It has been observed that the rules for a fixed center are monotonic in 'd'. The number of rules with different centers is depicted in the Figure 5.4 for class A of GPCR.

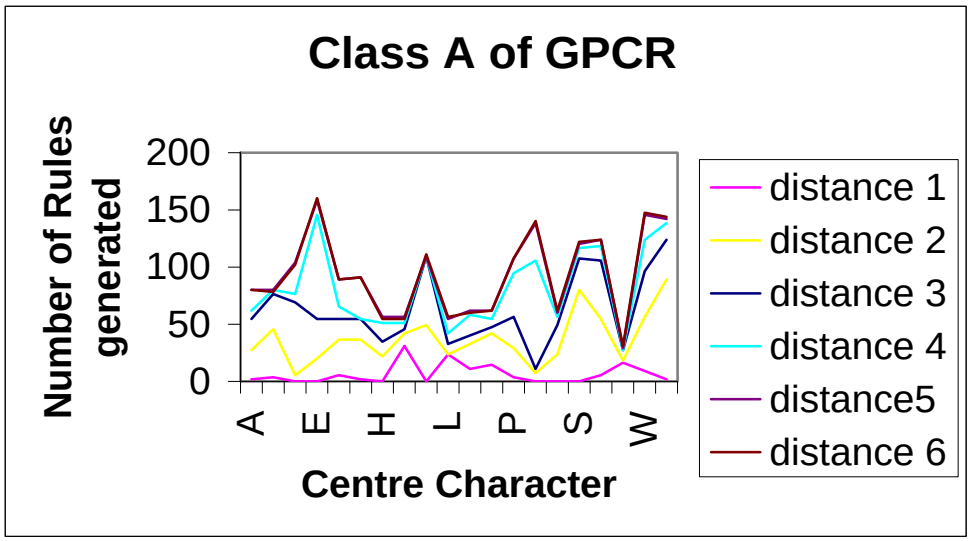

Figure 5.4: Number of Rules generated for Class A of GPCR

It has been noticed that the line indicating d = 5 and d = 6 are very close and more or less same in case of Least Frequent occurring Character. Hence 'd' is frozen as five in the present study for extracting the rules and preparing the rule database.

5.5.3 Five-Fold Test

Five-fold test has been conducted on the dataset and the results are furnished in Table 5.9. The dataset is divided into 5 sets, one set is taken as test set and the remaining four sets are combined to form a training set. LFC indicates the Least Frequent occurring Character and Reduct is the set of predominant attributes obtained by using Reduct Computation Algorithm.

Table 5.9: Reduct for the LFC for different training sets

SET	LFC	Reduct
Set 1	H	E K Y G N C D R Q M F H P T V I A S
	M	E N Y D R K P G C Q I M W H F T S A V
	Q	Q K D N E R F P Y G I M C V T H A
	W	D R E P N C K M F H A Y V I Q W T S G
Set 2	H	E C Y R Q N G D K M H V P T F A S I
	M	E R K N D Y Q G H M C W F P V I T S
	Q	Q D Y N M K I P G E F V R C W A H T
	W	D E N R C P K W M Q H F A T I Y G S
Set 3	H	E Q C G N D R K P M F H I V Y S A T W
	M	E N K R Y D P M W C Q G H F I T V S
	Q	D N Y Q K P F E R H I M V C W A T G
	W	D R E C N K P W Q M I H A F T V Y S G
Set 4	H	E C Q Y K D N G H M I P F R V T A W
	M	E N K R D Y P Q C W H I F G V T M S
	Q	Q N Y K D P C I G E F V M R A T W H
	W	E D C R P N K W M H Q A F I T Y S G
Set 5	H	E Y C Q D K N R G M H V P I F S A T W
	M	E K R P D Y N Q M C W F G H I T S V
	Q	D Q N Y K C P F R G I E A V W H M T
	W	D E R C N P K W M H Q I T Y A V F S

Five-fold test has been performed on the GPCR. The available 3896 sequences of GPCR has been divided into 5 sets (Set 1, 2, 3, 4, 5). Any one set is taken as test set and the remaining four sets have been combined to form training set. The result of

Five-fold test is given in Table 5.10. It shows that on an average 81% of the sequences are classified accurately.

Table 5.10: Results of Five-fold test of GPCR dataset

Set	Correctly classified	Misclassified
Set 1	80%	20%
Set 2	80%	20%
Set 3	82%	18%
Set 4	84%	16%
Set 5	82%	18%

5.5.4 Compactness of Reduct based Decision Tree

Number of rules generated with Reduct based Decision Tree and Decision Tree (ID3) [Arun, 2001] for different training set is as in Figure. 5.5.

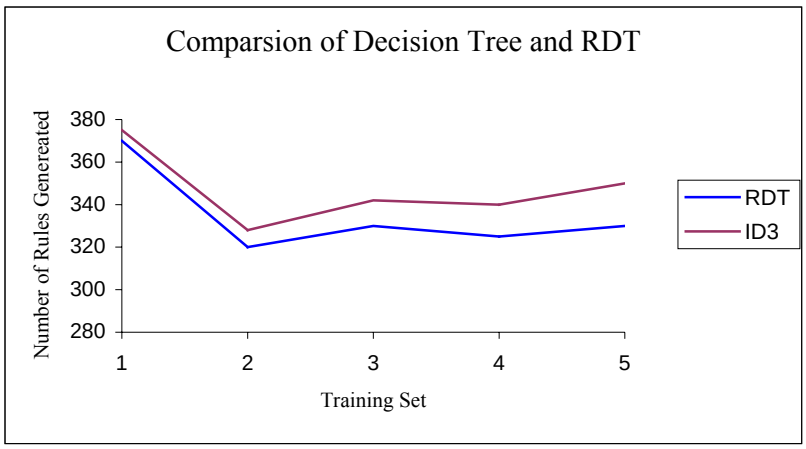

Figure 5.5: Comparison of Rules generated in ID3 and RDT

Though the number of rules by RDT is less, without losing any amount of accuracy RDT performed better than decision tree. The comparison result of Five-fold test on GPCR dataset is shown in Table 5.11. The results depicts that about 80% of the GPCR data has been classified correctly.

Table 5.11: Comparison of Five-fold test on GPCR dataset

	Correctly Classified		Misclassified	
Set	RDT	ID3	RDT	ID3
Set 1	80%	78%	20%	22%
Set 2	80%	81%	20%	19%
Set 3	82%	80%	18%	20%
Set 4	84%	84%	16%	16%
Set 5	82%	80%	18%	20%

Rules obtained for set 1 with centers 'H', 'M', 'Q', 'W' are given in Appendix B respectively as sample [Ramadevi, 2008a].

5.6 NEIGHBORHOOD ASSOCIATIONS MODULE

The family information has been obtained from the Sequence Arithmetic module. The inability to identify the class information has been conquered by performing Neighborhood Analysis. The input to the Neighborhood Association module is family information and unidentified sequence 'y'. The output will be class information as in Figure 5.6.

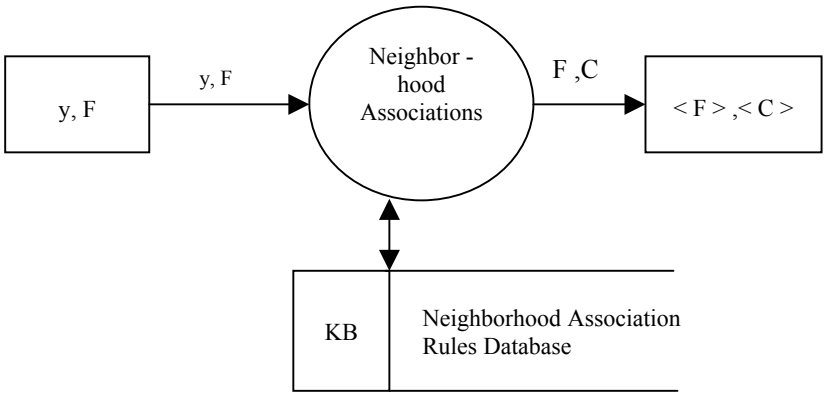

Figure 5.6: Neighborhood Associations Module

5.7 RESULTS AND DISCUSSIONS

The proposed Reduct based Decision Tree has been demonstrated on Sunburn, Weather, and GPCR datasets. Reduct based Decision Tree is compared with other classification tools (WEKA tool is used for experimentation of other methods and

Reduct based Decision Tree has been constructed according to the procedure). Kappa Statistics has been used for measuring the efficiency.

The efficiency of Reduct based Decision Tree over other algorithms is shown in Table 5.12. In case of large dataset, RDT and RandomForest are equally efficient.

Table 5.12 Comparison of classification techniques using Kappa Statistics

	Kappa Statistics		
Classification Technique	**Sunburn**	**Weather**	**GPCR**
Bayes Network	0	0	0.306
ComplementNaiveBayes	0.142	0	0.29
NaiveBayesMultinomial	0	0	0.013
Logistic	0	0.588	0.362
RBFNetwork	0	0.256	0.416
SimpleLogistic	0	0	0.359
SMO	0	0	0.338
BFTree	0	0	0.574
J48	0	0.143	0.582
J48graft	0	0.143	0.585
LMT	0	0	0.556
NBTree	0	0	0.445
RandomForest	0	0.429	0.652
RandomTree	0	0.378	0.595
SimpleCart	0	0	0.572
RDT	0.5	0.58	0.62

5.8 SUMMARY

Neighborhood Association concept has been introduced, binarization threshold has been recommended as 0.05 and neighborhood distance ‘d’ has been fixed as 5. Based on these parameters the construction of BAM for a given center has been developed.

The performance evaluation of the hybridized Neighborhood Associations module and Reduct based Decision Tree reduced the search space by 67%, thus confining the search to 37% (Appendix D).

By attributing class information to the BAM and treating that as decision table, Predominant Attributes set has been derived as described in RCA (Chapter IV). Using Predominant Attributes a decision tree has been constructed for identifying class. BAM has been further used for constructing concept lattice and hence, map the unknown protein to a set of known proteins. The process of construction of concept lattice and generation of rules is discussed in the following chapter.

The derived information and rules are organized in respective databases (Appendix B) for quick process in identifying a new protein.

CHAPTER VI
CONCEPT LATTICE

The set of class / classes information has been obtained from the knowledge built about the proteins with the modules developed in Chapters III, IV and V. The number of proteins to which the queried protein needs to be compared is relatively small by this stage. To further reduce the complexity of comparisons, and the search space, Concept Lattice [Godin, 1995] is introduced in this Chapter.

A brief account of Association Rules is presented in Section 6.1. Section 6.2 develops an algorithm for constructing a Concept Lattice. Concept lattice Association Rules have been generated from the node information and are organized in database called Concept lattice Association Rules Database. Section 6.3 discusses about subclass identification. The Concept Lattice Module is presented in Section 6.4. Section 6.5 summarizes the chapter.

6.1 ASSOCIATION RULES

Knowledge is extracted from large databases using Association Rules [Agarwal, 1993]. Given a database of transactions, the problem of mining association rules consists of generating all association rules that have certain user specified minimum 'support' and 'confidence'. An association rule is an implication of the form $X \Rightarrow Y$, where X and Y are subsets of A, also called "itemsets", and $X \cap Y = \Phi$.

6.1.1 Support

Let $A = \{l_1, l_2, .., l_m\}$ be a set of items, and 'T' be a transaction database which is a set of transactions 't', where each transaction 't' is a set of items [Arun, 2001] .

A transaction 't' is said to support an item l_i, if l_i is present in 't'. 't' is said to support a subset of items $X \subset A$, if 't' supports each item l_i in X. An itemset $X \subset A$ has a support 's' in T, denoted as s(X), if s% of transactions in 'T' support 's'.

The support of a rule $X \Rightarrow Y$ is defined as sup $(X \cup Y)$ while its confidence is computed as the ratio sup$(X \cup Y)$ / sup(X).

6.1.2 Confidence

An Association Rule X $\Rightarrow$ Y holds with confidence Ť, if Ť% of transactions in 'D' that supports X also supports Y.

6.2 CONCEPT LATTICE ASSOCIATION RULES AND DATABASE

The construction of the concept lattice is based on Binary Association Matrix. The process of Concept lattice Association Rules generation and storing in the database is discussed in this section.

6.2.1 Concept lattice Association Rules Generation

Concept lattice Association Rules have been generated for each class 'C' of family 'F' with center 'x' based on corresponding BAM. The node information of the lattice furnishes the association between the attributes and objects. The node information is stored in the database [Ramadevi, 2005]. The generation of the rules is discussed in Algorithm 6.1

Algorithm 6.1:

Algorithm CAR Generator (BAM, F, C, x)

Input: BAM; Family F; Class C; Center x;

Output: CARules.

1. Construct Concept Lattice[1] based on Binary Association Matrix, for the sequences in a class 'C' (within family F) for a specified center 'x'.
2. Generate the Concept lattice Association Rules based on the node information of Concept Lattice.
3. Append Fid (family id), Cid (class id) and center information to the generated Concept lattice Association Rules.

6.2.2 Concept lattice Association Rules Database

The Concept lattice Association Rules generated in Algorithm 6.1 have been stored in the database known as Concept lattice Association Rules Database. This derived database has been used for mapping of the Association Rules generated for a new sequence.

[1] Galicia – Galois lattice builder tool –University of Montreal, DIRO, Montreal, Quebec, Canada.

The record structure of the Concept lattice Association Rules Database is
(Fid, Cid, Center, R-Ante, R-Cons).
Where Fid- Family Identifier,

Cid- Class Identifier,

Center-center /LFC,

R-Ante - rule antecedent,

R-Cons - rule consequent.

Example 6.1:

Consider a record structure (MYO, 2, W, {A,E,G,H,K,L,Q,S,V},{1,3,8})

Fid → MYO indicates family is Myoglobin,

Cid → 2 indicates class is '2',

Center → W indicates the center considered is W,

R-Ante → {A,E,G,H,K,L,Q,S,V},

R-Cons → {1,3,8}

If the values of all attributes in the R-Ante set {A,E,G,H,K,L,Q,S,V} is one then the search is limited to the sequences in the R-Cons set {1,3,8}.

CARDGenerator (Algorithm 6.2) describes the steps involved in the creation of Concept lattice Association Rules for each class 'C' of each family 'F' with different centers and store them in Concept lattice Association Rules Database [Ramadevi, 2006c; Ramadevi, 2006d].

Algorithm 6.2:

Algorithm CARDGenerator (D)

Input: Database 'D', Threshold 'T'

Output: CARD

1. Initialize CARD ← NULL.
2. Repeat Step 3 to Step 9 for each family 'F'.
3. Repeat Step 4 to Step 9 for each class 'C'.
4. Repeat Step 5 to Step 9 for each selected center 'x'.
5. Repeat Step 6 to Step 8 for each sequence 's' of D.
6. Initialize BAM ← NULL
7. Initialize distance d ←1
8. While d ≤ 5 do

{NM ← NM(s, d)

BAM ← BAM ∪ BAMD(NM,T,R).

d ← d+1

}

9. BAM ← disjunction of BAMD
10. CARulesFCx ← F, C, x, CAR(BAM, F, C, x)
11. CARules ← CARulesFCx
12. Append (CARules, CARD)

6.2.3 Concept Lattice construction

Consider the class E sequences of the G-Protein Coupled Receptor family. Class E has five sequences; neighborhood analysis has been performed with 'W' as center and distance 'd' as 5. Threshold has been taken as 0.05 and the corresponding Binary Association Matrix has been generated as shown in Table 6.1. The corresponding Concept Lattice is as in Figure 6.1

Table 6.1: BAM for Class E sequences of GPCR family

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
1	1	1	0	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	1	0
2	1	1	0	0	1	1	0	1	0	1	0	1	0	0	1	0	0	1	1	0
3	1	1	0	0	1	1	0	1	0	1	1	1	0	0	1	1	1	1	1	1
4	1	1	0	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1
5	1	1	0	0	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	0

The node is in the form of ({A}, {S}), where {S} is the set of objects/ proteins/ sequences and {A} is set of attributes/ items. The node ({A}, {S}) is interpreted as the set of attributes {A} have value '1' for objects {S}.

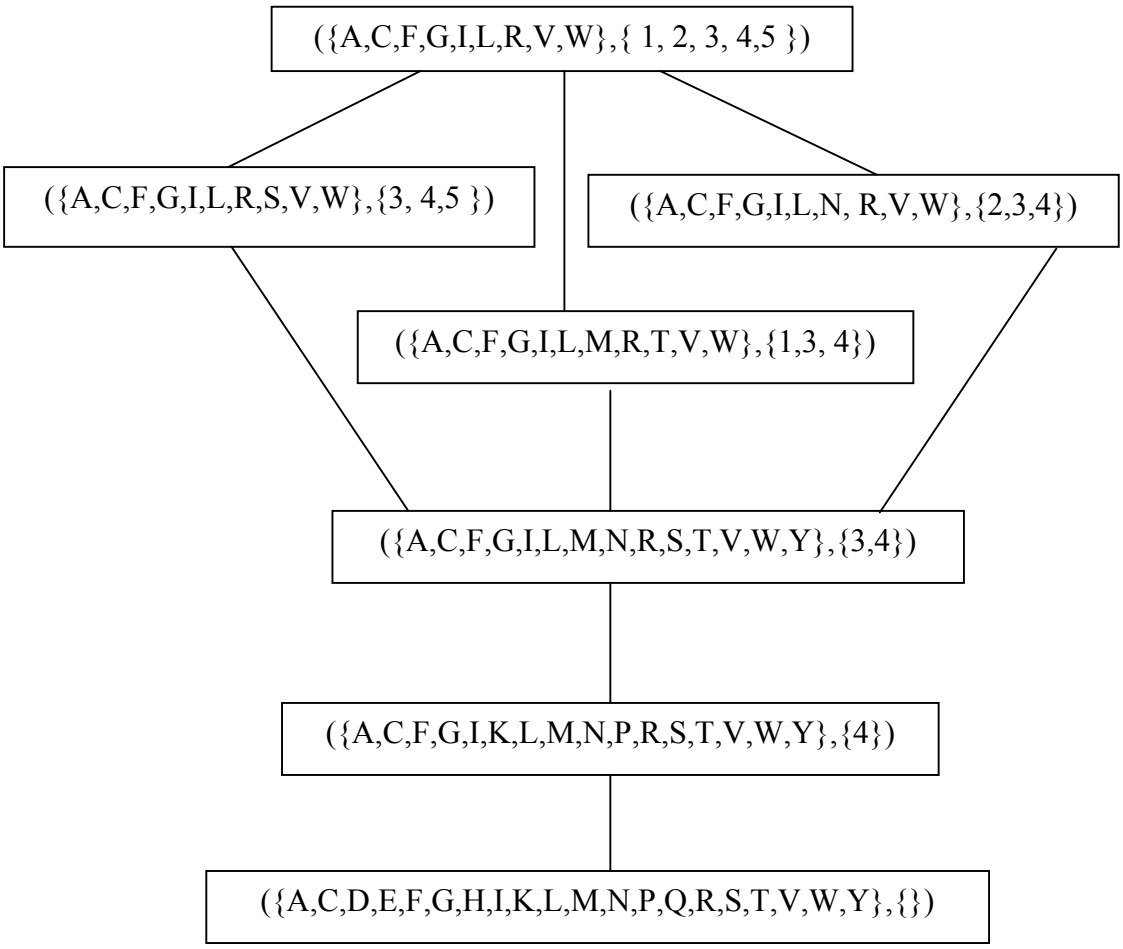

Figure 6.1: Concept Lattice for Class E of GPCR

Example 6.2:

Considering the node ({A,C,F,G,I,L,N,R,V,W},{2,3,4}) of Figure 6.1 the sequences 2, 3 and 4 of Class E have value '1' for A,C,F,G,I,L,N,R,V and W attributes among the Predominant Attributes.

The Concept lattice Association Rules Database entry associated with this node is

(Fid, Cid, Center, R-Ante, R-Cons).

(GPCR, E, W, {A,C,F,G,I,L,N,R,V, W}, {2,3,4})

and is interpreted as family is GPCR, Class is 'E', Center is 'W' , R-Ante is {A,C,F,G,I,L,N,R,V, W} and R-Cons is {2,3,4}.

The Concept lattice Association Rules generated from the above Concept lattice (Figure 6.1) are as follows:

(Family, Class, Center, R-Ante, R-Cons)

(GPCR, E, W, { R,W,C,L,G,V,I,A,F}, { 3,2,1,5,4})

(GPCR, E, W, { R,W,C,L,G,V,I,A,F,N}, { 3,2,4})

(GPCR, E, W, { S,R,W,C,L,G,V,I,A,F}, { 3,5,4})

(GPCR, E, W, { R,W,C,L,G,V,I,M,A,F,T}, { 3,1,4})

(GPCR, E, W, { S,R,W,C,L,G,V,I,M,A,F,Y,T,N}, { 3,4})

(GPCR, E, W, { S,K,R,W,C,P,L,G,V,I,M,A,F,Y,T,N},{ 4})

(GPCR, E, W, { S,D,K,R,W,H,C,P,L,Q,G,V,I,M,A,F,Y,T,N,E},{ })

6.3 SUBCLASS IDENTIFICATION

The process involved in the identification of family and class of unknown sequences is explained in Chapters III, IV and V. This section discusses the process of subclass identification of unknown sequence 'y'. For 'y' with Least Frequent occurring Character as center, the neighborhood analysis has to be carried out. The Predominant Attributes of the corresponding Least Frequent occurring Character will be used for projecting this neighborhood information. The Predominant Attributes with value '1' are considered as a set {A}. Set {A} is mapped to the antecedent of the rules in Concept lattice Association Rules Database for the given family and class information. The consequent {S} of mapped rule {A} is declared as the subclass of 'y'.

Example 6.3:

Consider the unknown sequence 'y'

MVFLSGNASDSSNCTQPPAPVNIPKAILLGVILGVLILFGVPGNILVILSVACHR
HLHSVTHYYIVNLAVADLLLTSTVLPFSAIFEILGYWAFGRVFCNIWAAVDRL
CCTASIMSLCIISIDRYIGVSYPLRYPTIVTQRRGLRALLCLWWLTIVISIGPLFG
ARQIARQDETICQINEDPSYVLFSALGSFYVPLAIILVMYCRVYVVAKRESRGL
TSGLKTDKSDSEQVTLRIHRKNAPLGGSGVASSKNKTHFSVRLLKFSREKKAA
KTLGIVVGCFVLCWLPFFLVMPIGSFFPDFKPSETVFKIVFKLGYLNSCINPIIYP
CSSQEFKKAFQNVLKIQCLRRKQSSKHALGYTLHPPSQAVEGQHKDMVRIPV
GSRETFYKISKTDGVCEWWFFRSMPRGSARITVPKDQSACTTARVRSKSFLQV
MMCVGPSTPNPHQV

The Alphabet set of sequence 'y' is

A(y) = {A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

As A(y) ⊃ P(GPCR), ‘y’ will be identified as string in the Family GPCR (section 3.4.1).

The Least Frequent occurring Character has been noticed as ‘W’ from the following frequency distribution (Table 6.2) of the characters of sequence ‘y’

Table 6.2: Frequency distribution table of string ‘y’

Char	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	**W**	Y
Freq	27	17	12	11	25	27	10	35	24	45	8	13	26	15	26	41	22	41	**7**	14

The corresponding neighborhood vector for ‘y’ with center ‘W’ is as shown in Table 6.3 and the corresponding binarized vector using threshold ‘T’ =0.05 is shown in Table 6.4.

Table 6.3: Neighborhood analysis of sequence ‘y’ at distance ‘d’ =5 with center ‘W’

N(W,A,5)	0.057	N(W,M,5)	0.029
N(W,C,5)	0.1	N(W,N,5)	0
N(W,D,5)	0.029	N(W,P,5)	0.014
N(W,E,5)	0.043	N(W,Q,5)	0
N(W,F,5)	0.129	N(W,R,5)	0.057
N(W,G,5)	0.057	N(W,S,5)	0.029
N(W,H,5)	0	N(W,T,5)	0.029
N(W,I,5)	0.071	N(W,V,5)	0.114
N(W,K,5)	0	N(W,W,5)	0.057
N(W,L,5)	0.171	N(W,Y,5)	0.014

Table 6.4: Binarized vector of Table 6.3 data with Threshold ‘T’ = 0.05

Centre	**A**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**
W	1	1	0	0	1	1	0	1	0	1	0	0	0	0	1	0	0	1	1	0

The reduct for identification of the class is {D, E , R, P, N, C, K, W, M, H, Q, I, Y, T, A, F, V, S, G} from the Appendix B under the GPCR family with center ‘W’.

Based on the Binary Association Matrix values and Reduct based Decision Tree Rules for the complete GPCR dataset, the outcome of the rule is that the unknown sequence 'y' belongs to class E (class 5)(Appendix B).

To identify the subclass or set of sequences to which 'y' is closer, the Concept lattice Association Rule Database (Appendix C) has to be searched. The query for search is of the form (GPCR, E, W, {R,W,C,L,G,V,I,A,F}, {?}). The database is searched for a match and when match is found the Rule consequent of the matched rule is given. The given query matches with the rule (GPCR, E, W, {R,W,C,L,G,V,I,A,F}, {3,2,1,5,4}), therefore {3,2,1,5,4} is the output.

The query can also be answered by interpreting the concept lattice diagram as in Figure 6.1.

For the unknown sequence 'y', {A, C, F, G, I, L, R, V, W} has value as '1' (Table 6.4). The search proceeds from bottom to top, finding for match in attribute values. The attributes {A, C, F, G, I, L, R, V, W} are matched with the attributes of the node ({A,C,F,G,I,K,L,M,N,P,R,S,T,V,W,Y},{5}). Since 'K' is not '1' in the 'y', proceed in upward direction until a match ({A,C,F,G,I,L,R,V,W},{ 1, 2, 3, 4,5 }) is found. At this point, it is concluded that the unknown sequence 'y' is closer to sequences {1,2,3,4,5}. Hence 'y' is closer to sequence 1, 2, 3, 4, & 5 of class E of GPCR family.

Example 6.4: Myoglobin

Consider the sequence s1

VLSEGEWQLVLHVWAKVEADVMAGHGQDLRLFKSHPETLEKFDRFKHLKT
EAEMKASEDLKKHGVTNVLTALGALKKKGHHEAELKPLAQSHATKHKPKY
LEFSEAHVLHSRHPGDFGNNADAQGAMNKALELFRKDAAKYKELGYQG

A(s1) ={A,D,E,F,G,H,K,L,M,N,P,Q,R,S,T,V,W,Y}

P(GPCR)={A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

P(Myoglobin) = {}

P(Myoglobin) $\subset$ A(s1). Therefore the sequence s1 belongs to Myoglobin family (Appendix A).

A(s1) $\supset$ P2.(Table 6.5)

Table 6.5: P-Table for s1

p-set	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Therefore the sequence s1 belongs to class 2 of the Myoglobin family

The Least Frequent occurring Character is noticed as 'W' from the following frequency distribution (Table 6.6) of the characters of sequence 's1'.

Table 6.6: Frequency distribution of sequence s1

Char	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
Freq	17	0	7	14	6	11	12	0	19	18	3	4	4	5	4	6	5	8	2	3

Select 'W' as least frequent occurring character and Binary Association Matrix is constructed with threshold 'T' = 0.05 and distance 'd' =5 (Table 6.7)

Table 6.7: Binary Association Matrix with 'W' as center

Char	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
Freq	1	0	0	1	0	1	1	0	1	1	0	0	0	1	0	1	0	1	0	0

The query is (MYO, 2, W, {A,E,G,H,K,L,Q,S,V,},{?}) .The search in the Concept lattice Association Rules Database for the family Myoglobin and class 2 for {A,E,G,H,K,L,Q,S,V,} gives the rule consequent as {1,3,8}. Therefore the queried sequence 's1' is closer to sequences {1,3,8}.

6.4 CONCEPT LATTICE MODULE

The Concept Lattice module is as shown in Figure 6.2 where the input to the module is set of classes < C >, family < F >, BAM(y)and an unknown string 'y'. The output is set of sequences < S > to which 'y' is close.

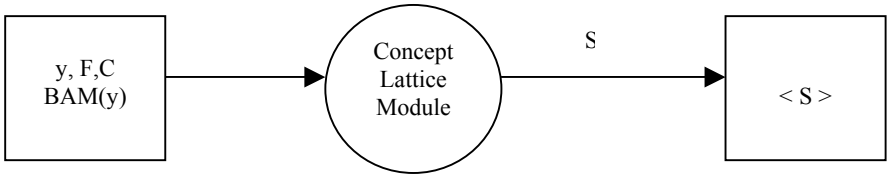

Figure 6.2: Concept Lattice Module

The generation of Concept lattice Association Rules and its storage are shown in Figure 6.3

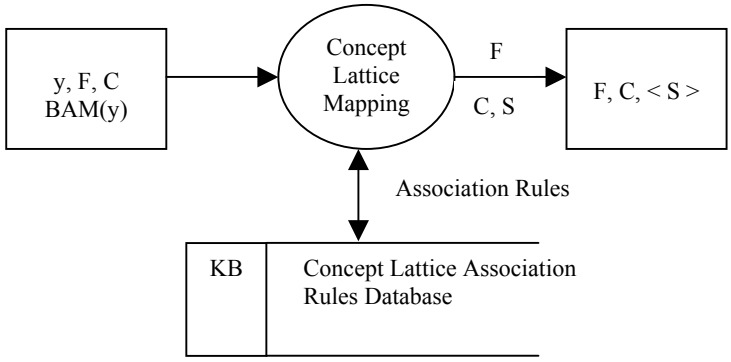

Figure 6.3: Internal process of the concept lattice module

6.5 SUMMARY

Concept lattice Association Rules (Appendix C) for the two families GPCR and Myoglobin have been framed based on the statistics obtained by neighbor's information. These rules have been used for the generation of subclass features and local features. They have been stored in database called Concept lattice Association Rules Database (record structure is given in Appendix C). The search reduced to around twenty sequences, on an average for an unknown sequence 'y' with family and class information.

It is observed that the further search will confine to 5 to 20 protein sequences once the family and class information of the unknown sequence 'y' is provided.

The empirical study reveals that the hybridized concept lattice module reduced the search space to about 9%.

CHAPTER VII

DESIGN AND ANALYSIS OF ROUGH SET PROTEIN CLASSIFIER

The Rough Set Protein Classifier (RSPC) consists of four modules, which aids in the classification of the protein. The four modules are:

i. Sequence Arithmetic (SA) Module.
ii. Reduct based Decision Tree (RDT) Module.
iii. Neighborhood Associations (NA) Module.
iv. Concept Lattice (CL) Module.

7.1 MODULES OF ROUGH SET PROTEIN CLASSIFIER

The different modules of Rough Set Protein Classifier are explained in this section.

7.1.1 Sequence Arithmetic Module

Three characteristics sets such as A-Set, D-Set and P-Set have been defined for protein sequences. They have been generated for classes of proteins and family of proteins. This information has been stored in the Sequence Arithmetic Knowledge Database (Appendix A). Given an unknown sequence 'y', A(y) and D(y) are generated (Chapter III). Family information has been extracted by mapping these sets to the information present in the Sequence Arithmetic Knowledge Database. The output of the Sequence Arithmetic Module is family information < F >. Therefore, the search space has been confined to only < F >, thus reduction in the domain search space is inversely proportional to **'f'** (number of families under study). The present study has been confined to two families; hence expected reduction is about 50%.

7.1.2 Reduct based Decision Tree Module

A-Table, D-Table and P-Table have been generated to obtain class information within a family. A new approach has been proposed to obtain the Rank ordered Predominant Attributes from the decision table (P-Set or A-Set or D-Set with a ClassID (decision attribute) concatenated as the first column) and a decision tree is created based on these Predominant Attributes. Rules have been generated from the decision tree and stored in the Reduct based Decision Tree Rules Database.

Databases have been generated using A-Table, D-Table or P-Table. However the present study is confined to P-Table (P-Sets). The decision tree for corresponding family will help in classifying the queried protein into a class or set of classes respectively.

7.1.3 Neighborhood Associations Module

Localized information has been extracted through neighborhood analysis in this module. Neighborhood associations concept has been introduced and binarization threshold 'T' has been recommended as 0.05 and neighborhood distance 'd' has been fixed at 5. Based on these parameters, the construction of Binary Association Matrix for a given center has been developed.

Predominant Attribute set has been derived as described in Reduct based Decision Tree module by attributing class information to the Binary Association Matrix and treating it as a decision table. Predominant Attributes are used to construct decision tree for identifying subclass. The decision tree based on the Binary Association Matrix for a family or a class will classify the questioned protein into a subclass. The size of the subclass may not be of manageable size. Thus, Binary Association Matrix has been further used for constructing concept lattice and hence, attaching the given unknown protein to a set of proteins.

7.1.4 Concept Lattice Module

The number of proteins to which the questioned protein needs to be compared will not be small by this stage. To further reduce the complexity of comparisons sequence by sequence, concept lattice [Godin, 1995] module was developed.

The concept lattice has been constructed based on the Binary Association Matrix (Chapter VI). The Concept lattice Association Rules have been generated from the nodes information of the concept lattice and stored in the database, Concept lattice Association Rules Database along with familyID, ClassID and Center. The discovered family/ class information for the queried protein has been compared with few similar proteins given by the set of proteins in the Concept lattice Association Rules Database.

The derived information of the questioned protein has been processed through the above modules to classify the protein, resulting in the reduction of search space. String comparisons are avoided for classification of family/ class.

7.2 DESIGN OF ROUGH SET PROTEIN CLASSIFIER

The process of different modules of the Rough Set Protein Classifier are shown in Figures 7.1(a), 7.1(b), 7.1(c) and 7.1(d). The unknown sequence ‘y’ is given as input to the Rough Set Protein Classifier.

In the Sequence Arithmetic module, A(y) and D(y) are generated. The generated A(y) is matched with the rules in Sequence Arithmetic Knowledge Database (SAKD). If A(y) is the superset of any of the families, then along the families information are given as output from the Sequence Arithmetic module, otherwise an error message is given(Figure 7.1(a)).

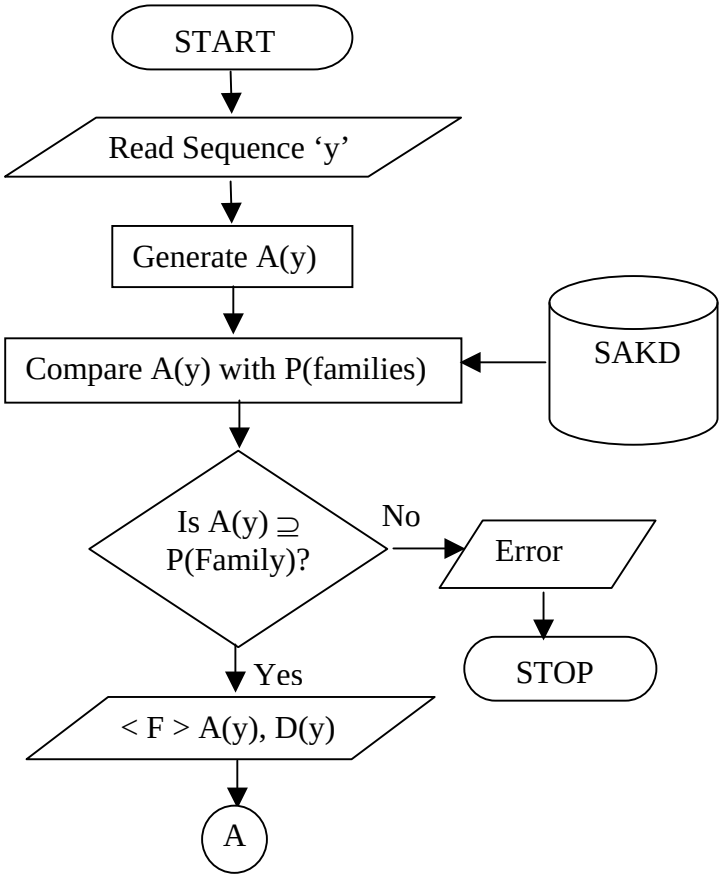

Figure 7.1(a): Sequence Arithmetic Module

The Reduct based Decision Tree module or Neighborhood Associations module is selected based on the family information. If RDT module is selected, P-Vector is generated and class information is obtained from rules present in RDTRD as in Figure 7.1(b).

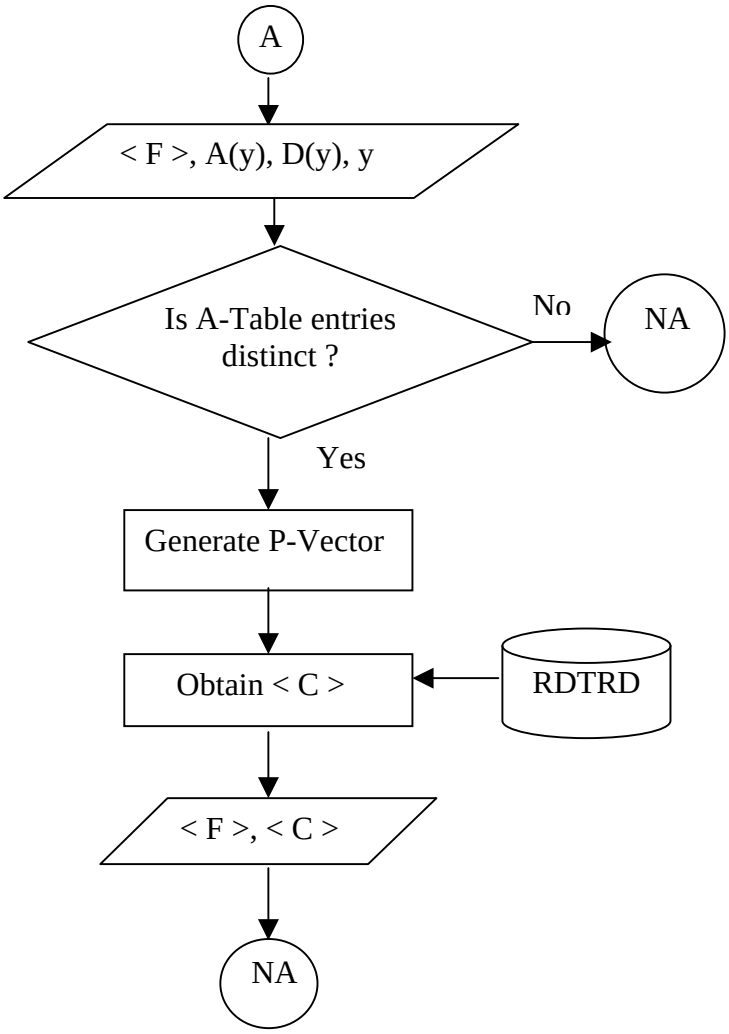

Figure 7.1(b): Reduct based Decision Tree Module

In Neighborhood Associations module, Binary Association Matrix (BAM) has been generated with Least Frequent occurring Character as center. The family and the predominant Attributes information have been utilized to obtain class information with the help of rules in NARD as in Figure 7.1(c).

In Concept Lattice module the generated BAM is utilized to map the rules present in the database (CARD) and generate the set of sequences < S > to which the given unknown sequence ‘y’ matches as in Figure 7.1(d).

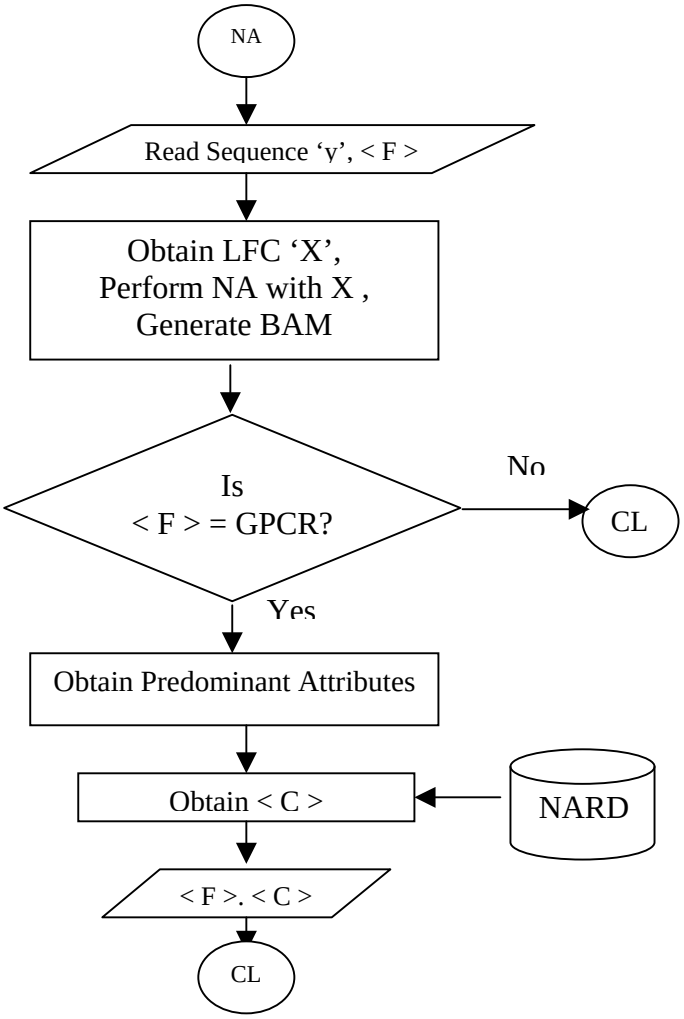

Figure 7.1(c): Neighborhood Associations Module

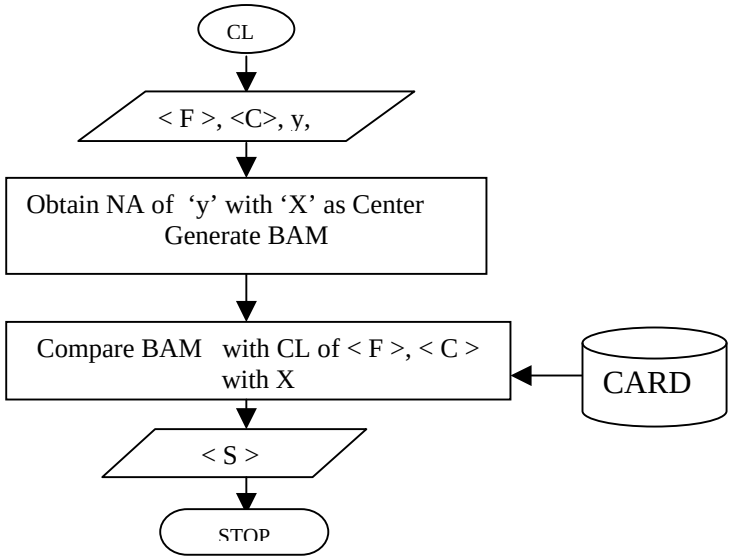

Figure 7.1(d): Concept Lattice Module

7.3 ILLUSTRATION OF ROUGH SET PROTEIN CLASSIFIER

Given an unknown sequence ‘y’ to the Rough Set Protein Classifier the output is family information, class information and subclass information (set of sequences).

Example 7.1:

Consider the unknown sequence ‘y’

MVFLSGNASDSSNCTQPPAPVNIPKAILLGVILGVLILFGVPGNILVILSVACHR
HLHSVTHYYIVNLAVADLLLTSTVLPFSAIFEILGYWAFGRVFCNIWAAVDRL
CCTASIMSLCIISIDRYIGVSYPLRYPTIVTQRRGLRALLCLWWLTIVISIGPLFG
ARQIARQDETICQINEDPSYVLFSALGSFYVPLAIILVMYCRVYVVAKRESRGL
TSGLKTDKSDSEQVTLRIHRKNAPLGGSGVASSKNKTHFSVRLLKFSREKKAA
KTLGIVVGCFVLCWLPFFLVMPIGSFFPDFKPSETVFKIVFKLGYLNSCINPIIYP
CSSQEFKKAFQNVLKIQCLRRKQSSKHALGYTLHPPSQAVEGQHKDMVRIPV
GSRETFYKISKTDGVCEWWFFRSMPRGSARITVPKDQSACTTARVRSKSFLQV
MMCVGPSTPNPHQV

The Alphabet set of sequence ‘y’ is

A(y) = {A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

As A(y) $\supset$ P(GPCR), ‘y’ is identified as sequence in the family GPCR. (Appendix A)

Example 7.2:

Consider an unknown sequence ‘y’

GLSDGEWQLVLHVWGKVEADLAGHGQEVLRLFKGHPETLEKFNKFKHKSE
DEMKASEDLKKHGVTVLTALGGVLKKKGHHEAEPLAQSHATKHKPKLEFSE
AHVLQSKHPGFGADAGAMNKALELFRKDAAKKELGFQG

A(y) = {A,D,E,F,G,H, K,L,M,N,P,Q,R,S,T,V,W}

P(Myoglobin) = { }

P(GPCR) ={A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

Since P(Myoglobin) $\subseteq$ A(y) , ‘y’ belongs to Myoglobin family (Chapter III). CIdentifier algorithm is applied with A(y) as input, and output is that ‘y’ belongs to Class 2 (Appendix B1).

Example 7.3:

Consider the sequence s2

VLSEGEWQLVLHVWAKVEADVMAGHGQDLRLFKSHPETLEKFDRFKHLKT
EAEMKASEDLKKHGVTNVLTALGALKKKGHHEAELKPLAQSHATKHKPKY
LEFSEAHVLHSRHPGDFGNNADAQGAMNKALELFRKDAAKYKELGYQG

A(s2) ={A,D,E,F,G,H,K,L,M,N,P,Q,R,S,T,V,W,Y}

P(GPCR)={A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

P(Myoglobin) = {}

P(Myoglobin) $\subset$ A(s2).

Therefore the sequence s2 belongs to Myoglobin family (Appendix A).

A(s2) is superset for P2 (Table 7.1).

Table 7.1: P-Table for s2

p-set	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Therefore the sequence s2 belongs to class 2 of the Myoglobin family

The Least Frequent occurring Characters can be noticed as ‘W’ (other than with ‘0’), from the following frequency distribution (Table 7.2) of the characters of sequence ‘s2’.

Table 7.2: Frequency distribution for sequence s2

Char	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	**W**	Y
Freq	17	0	7	14	6	11	12	0	19	18	3	4	4	5	4	6	5	8	**2**	3

The Least Frequent occurring Character is selected as ‘W’ and Binary Association Matrix is constructed (Table 7.3).

Table 7.3: BAM for sequence s2

Char	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
Freq	1	0	0	1	0	1	1	0	1	1	0	0	0	1	0	1	0	1	0	0

The search for a match where R-Ante is {A,E,G,H,K,L,Q,S,V} in the Concept lattice Association Rules Database belonging to the family Myoglobin and class 2 is the R-Cons {1,3,8}

Sequence Arithmetic module takes 'y' as input and computes A(y) and D(y). This information is mapped to the data present in the Sequence Arithmetic Knowledge Database and family information is obtained.

7.4 RESULTS AND ANALYSIS

Performance evaluation of the developed modules is discussed in this section. Compositions and frequency occurrence of Amino Acids has been studied while creating databases. Horizontal fragmentation of the domain search space (in levels) has been achieved while traversing from one module to another following protein hierarchy. Spatial information has been obtained by performing neighborhood analysis.

7.4.1 Creation of Different Databases

Sequence Arithmetic Knowledge Database has been created based on the A-Set, D-Set and P-Set generated for the datasets. The Sequence Arithmetic rules have been derived and stored in the Sequence Arithmetic Knowledge Database (Appendix A). The datasets considered for experimentation were GPCR and Myoglobin. Table 7.4 shows the different sets generated for the datasets. Table 7.5 depicts the different characteristics of the datasets.

Table 7.4: Sets generated by Sequence Arithmetic Module

Family	A-Set	D-Set	P-Set
Myoglobin	{A,B,C,D,E,F,G,H,I,K,L, M,N,P,Q,R,S,T,V,W,Y,Z}	{}	{}
GPCR	{A,C,D,E,F,G,H,I,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{A,C,D,E,F,G,H,I,K,L,M, N,P,Q,R,S,T,V,W,Y}

Table 7.5: Characteristics of datasets

Family	Number of sequences	Number of classes	Number of distinct A-Sets	Number of distinct D-sets	Number of distinct P-sets
Myoglobin	399	36	15	15	17
GPCR	3896	5	1	1	1

Reduct based Decision Tree Rules have been generated for different classes of the datasets and stored in Reduct based Decision Tree Rules Database (Appendix B). Reduct based Decision Tree Rules are used to obtain the class information. Rules for Myoglobin have been constructed based on the P-Table generated in Reduct based Decision Tree module, whereas for G-Protein Coupled Receptor, neighborhood associations has been performed (since only one set is generated).

Concept lattice Association Rules have been generated separately for each class with different Least Frequent occurring Character as centers. These rules are stored in Concept lattice Association Rules Database (Appendix C). The creation of the database has been an offline process; therefore the complexities are not of concern in the present work.

7.4.2 Time Complexity (Queried sequence)

For an unknown sequence 'y' of size 'm', a single scan is required for obtaining the A(y) and D(y). If the number of families present in the database is 'f' then the number of comparisons will be O(f). Therefore the worst case complexity was of the order of O(m) + O(f) .

Let 'C' be the number of classes in a given family then complexity of RDT will be O(log C). At the end of SA and RDT module the complexity is O(m) + O(f) + O(log C). The complexity of Neighborhood Analysis module followed by RDT module is O(20m)+O(log C).

If 'r' is the number of proteins in classes then the concept lattice will have 2^r nodes. Therefore the number of comparisons required will be $O(2^r)$.
Thus, the complexity is $O(m)+O(f)+O(\log C)+O(2^r)$.

The empirical analysis given in Appendix D and section 7.6 concludes $m \approx 400$, $f \approx 1.5$, $C \approx 27$ to 30 and $r \approx 8$ (on an average).

7.4.3 Performance Evaluation of the Modules

The performance of various modules is discussed in this subsection.

7.4.3.1 Sequence Arithmetic Module

Let T_A be the complexity for finding the A-Set. For initial 'n' sequences, a single scan for A-Set is to be performed. T_A = O(nm), where 'm' is maximum length of any sequence. If the number of classes within a family is 'C' then $T_{A(C)}$ = O(22n)(as number of characters is 22).
For a family T_{SA} =O(nm)+O(3n*22) = O(nm).

Let 'b' be the complexity of searching for a sequence in a database with 'n' sequences without using Sequence Arithmetic. Family information is obtained by applying Sequence Arithmetic module. If n_1 sequences are present in a family, then $n_1 < n$. With Sequence Arithmetic it will be T_{SA} +O(n_1b) which will be less than or equal to O(nb). Therefore the complexity of Sequence Arithmetic is O(n_1b) +O(nm).

In the absence of Sequence Arithmetic module, a protein has the complexity of O(nb). This complexity can be reduced by adopting multi-level classifiers.

The application of the Sequence Arithmetic Rules to Myoglobin and GPCR families showed that there is 50% reduction in the search space.

7.4.3.2 Reduct based Decision Tree Module

The performance evaluation of the Reduct based Decision Tree algorithm with the existing classification technique (WEKA tool is used for experimentation of other methods and Reduct based Decision Tree was constructed according to the procedure) on the standard databases has been performed and the results are shown in Table 7.6. It is observed that Reduct based Decision Tree performed better than other methods in terms of Kappa statistics [Ramadevi, 2008a]. In case of large dataset (GPCR), Reduct based Decision Tree (RDT) and RandomForest are equally efficient.

Table 7.6: Comparison of RDT with other Classification Methods

	Kappa Statistics		
Classification Technique	**Sunburn**	**Weather**	**GPCR**
Bayes Network	0	0	0.306
ComplementNaiveBayes	0.142	0	0.29
NaiveBayesMultinomial	0	0	0.013
Logistic	0	0.588	0.362
RBFNetwork	0	0.256	0.416

SimpleLogistic	0	0	0.359
SMO	0	0	0.338
BFTree	0	0	0.574
J48	0	0.143	0.582
J48graft	0	0.143	0.585
LMT	0	0	0.556
NBTree	0	0	0.445
RandomForest	0	0.429	0.652
RandomTree	0	0.378	0.595
SimpleCart	0	0	0.572
RDT	0.5	0.58	0.62

Number of rules generated with Reduct based Decision Tree (RDT) and Decision Tree (ID3) for different training set have already illustrated in Figure 5.5 and results of five-fold test in Table 5.11.

7.4.3.3 Neighborhood Associations Module

The family information has been obtained from the Sequence Arithmetic module. The inability to identify the class information has been conquered by performing neighborhood analysis. The binarization threshold has been recommended as 0.05 and neighborhood distance‘d’ is fixed as 5. Based on these parameters, the construction of Binary Association Matrix for a Least Frequent occurring Character has been derived and further used for the construction of Concept lattice.

7.4.3.4 Concept Lattice Module

Subclass identification has been done in this module with the help of the family and class identifier along with center, this information was utilized for searching the Concept lattice Association Rules Database, where the need for comparison of questioned protein has been reduced to a set of few proteins (on an average 20 proteins)(Appendix D).

7.5 CROSS-VALIDATION OF ROUGH SET PROTEIN CLASSIFIER

The dataset with 3896 GPCR sequences and available 25 non-GPCR sequences have been tested. The results comprise of True Positives (TP)-3896, False Positives (FP)-6, True Negatives (TN)-19 and False Negatives (FN)-NIL. Therefore the accuracy was 99.85%.

Ten different test datasets consisting of 200 sequences from GPCR (selected randomly) and 24 non-GPCR sequences were added to each, thus making the set size of 224 sequences for testing. The average results were TP = 200, FP = 5, TN = 19 and FN = 0. Thus the average accuracy rate was 97.7%. [Ramadevi, 2007c]. The accuracy rate of Support Vector Machines [Pooja, 2004] has been compared with Rough Set Protein Classifier as reported in Table 7.7.

Table 7.7 Accuracy rate of the cross-validation for classifying the GPCR

Classification Methods						
SVM-lin	**SVM-Poly**	**SVM-Sig**	**SVM-rbf**	**SVM-pw**	**SVM-Fish**	**RSPC**
0.905	0.937	0.897	0.97	0.99	0.992	0.977

SVM-lin : Support Vector Machine with linear function.
SVM-Poly : Support Vector Machine with Polynomial function.
SVM-Sig : Support Vector Machine with Sigmoid function.
SVM-rbf : Support Vector Machine with radial basis function.
SVM-pw : Support Vector Machine with pair wise alignment.
SVM-Fish : Support Vector Machine with Fisher approach.
RSPC : Rough Set Protein Classifier.

7.6 ANALYSIS OF ROUGH SET PROTEIN CLASSIFIER

Samples of 60 sequences out of 1440 sequences were randomly selected and subjected to the Rough Set Protein Classifier. The empirical results are presented in Appendix D.

7.6.1 Sequence Arithmetic Module

In the Sequence Arithmetic module, it is observed that a sequence has to be compared with 885 sequences on an average instead of 1440, thus a reduction of

about 40% is observed. The domain search space is confined to about 60% (Appendix D).

7.6.2 Rough based Decision Tree Module

The RDT module applied to the Myoglobin family to identify the class reduced the search space to 762 sequences (the infirmity of the GPCR has been considered). Thus the gain is 47%.

7.6.3 Neighborhood Associations Module

In Neighborhood Associations module, the search has been confined to the families obtained from the sequence arithmetic module, resulting in reduction of the search proteins set to 530. The average numbers of classes have reduced from 36 to 25 classes. Observation shows that one has to search 530 sequences instead of 1440 sequences, thus the search space has been further confined to 37%.

The domain search space has been reduced by 63% at the end of the hybridized RDT module and Neighborhood Associations Module, and the average numbers of concept lattices to be compared have reduced to 25 (one for each class).

7.6.4 Concept Lattice Module

Exploiting the power of concept lattice further reduced the search space to 9%. All the search space reduction has been expected to preserve the accuracy of identification intact. Hence, the RSPC helps in reducing the search space identification of a queried protein from 100% to 9% (without compromising the potentiality for the identification the protein)(Appendix D)[Ramadevi, 2008b].

7.7 ANALYSIS OF SEARCH SPACE REDUCTION

The impact of Rough Set Protein Classifier modules in search space has been discussed in the above sections. The presence for the queried protein in the reduced space has been verified. It has been observed in all the levels, the queried protein is one among the reduced set, which indicates the confidence of identifying the exact protein is 100% as in Figure 7.2.

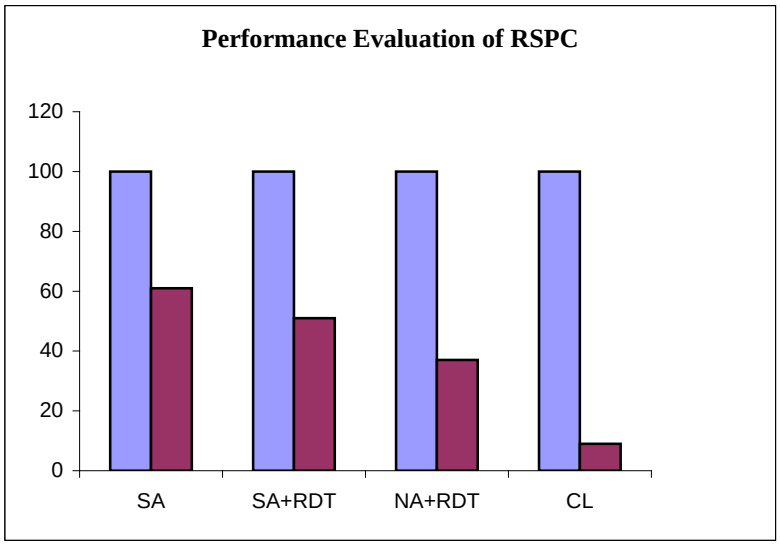

Figure 7.2: Space Reduction of different hybridized modules

The X-axis indicates the modules and Y-axis the percentage of the search space in Figure 7.2. The column with maroon colour depicts the confidence of belonging of queried protein with respect to the respective module. The experimental evidence reveals that Rough Set Protein Classifier reduces the search significantly without loss of identification.

CHAPTER VIII
SUMMARY AND FUTURE SCOPE OF WORK

Knowledge discovery plays an important role in classification problems. Protein classification is one such problem, where soft computing tools have been applied. In the present research, preprocessing has been done to reduce the domain search space, which help in accelerating the search process. In this process, Rough Set Protein Classifier (RSPC) has been evolved, which makes use of set theory concepts, concept lattice tool, and feature extraction tools, especially Rough Set tools. G-Protein Coupled Receptors (containing 1041 of 3896 sequences) and Myoglobin (containing 399 of 727 sequences) families have been considered for the demonstration of the RSPC tool.

In the present work, using Sequence Arithmetic, the family information for the known protein sequences has been obtained and stored in Sequence Arithmetic Knowledge Database (Appendix A). Decision tables have been generated with the sets obtained from Sequence Arithmetic Process and thus unstructured data has been converted into well-structured data. Sequence Arithmetic Rules have been framed from the obtained A-Set, D-Set and P-Set, which are essential for getting the family information. The domain search space has been reduced to about 40% by Sequence Arithmetic process, as the search is confined to a set of families only.

In the present thesis, a novel Reduct based Decision Tree (RDT) module, which is hybridization of Rough Sets and decision tree, has been developed and used for obtaining the class information. The rules extracted from the RDT have been stored in a database Reduct based Decision Tree Rules Database (Appendix B). The domain search space has been confined to class/ classes within the family, thereby reducing the search space in the ratio of about 2:1.

The limitation of Reduct based Decision Tree module has necessitated in acquiring the localized / spatial information of the sequences. Neighborhood Associations have been introduced and Binary Association Matrices have been derived. Neighborhood Associations, Binary Associations and decision table have been introduced in the present work to account for spatial characteristics. The

decision table based on neighborhood associations has been subjected to RDT module. The rules generated from the tree have been compiled and called Neighborhood Association Rules. These rules are stored in a database called Neighborhood Association Rules Database. The search space has been reduced in the ratio of about 5:2 at the end of Neighborhood Associations/ Analysis module.

Concept Lattice has been administered on the generated Binary Association Matrix, which mapped the queried protein to few proteins. Concept lattice Association Rules for each class of each family with different centers have been generated and stored in a database called Concept lattice Association Rules Database (Appendix C). The search for questioned protein has been confined to 20 proteins on an average. The computational gain has been in the ratio of 75:1, i.e., query to be compared with 75ρ sequences of the database by brute force method and with Rough Set Protein Classifier, it has to be compared with only 'ρ' sequences of the database. By hybridizing the concept lattice and neighborhood associations, it has been noticed the search space reduction is in the ratio of about 10:1.

The Rough Set Protein Classifier tool demonstrated in this thesis has been mostly based on equivalence relation and discretization strategies, which are simple for developing machine learning tools. Thus, the methods proposed in this thesis have been effective machine learning tools though computationally intensive (generation of rules is an offline process, thus it may not be detrimental to online decision making system).

Rough Set Protein Classifier can be adopted only after constructing the various databases, *viz.*, Sequence Arithmetic Knowledge Database (SAKD), Reduct based Decision Tree Rules Database (RDTRD), Neighborhood Association Rules Database (NARD), Concept lattice Association Rules Database (CARD), which demands a good amount of preprocessing of training set. The derived databases have been found to be rich in information, and used for processing queries at higher level. It has been also observed that the classification accuracy of Rough Set Protein Classifier is about 97.7%.

The significant changes in database due to addition of new sequences / classes have to be submitted to the learning system and the Knowledge Base can be rebuilt without human intervention.

The reduction in domain search space for any given sequence has been about 91%. For identifying exact match, one needs to perform BLAST algorithm on 9% of sequences in the training database instead of 100%.

Future Scope of Work

Spatial features coupled with Sequence Arithmetic like location-wise analysis are expected to have rich information. This may be a potential direction for exploring the tool developed in this thesis, mainly for sequential reduction of domain search space.

The Rough Set Protein Classifier tool has been demonstrated for reduction of search space level-by-level. The study can be further extended to hybridize Rough Set Protein Classifier with various available tools like Support Vector Machines, HMM. It is expected to produce better results with a relatively less computational time and space. A customized product can be developed for the complete protein database. Sequence Arithmetic demonstrated in this thesis is not merely limited to proteins. It can also be applied to any domain in which objects are represented as sequences.

One can make the proposed system adaptive, when new sets of proteins sequences are added to a protein database.

APPENDIX

Appendices A, B, C and D are present in the CD. However, samples of all appendices are given for quick glance.

Appendix A: Family and Class characteristics (A-Set, D-Set and P-Set) information of Myoglobin and GPCR family is available in section A1 and A2.

Appendix B: It comprises the rules for identifying the class/ classes based on RDT for Myoglobin in section B1 and Neighborhood Association Rules for GPCR with center 'W' in Section B2. NAR rules with centers 'Q', 'H' and 'M' are present in the CD. RDT rules database is referred as NARD in Neighborhood Associations Module.

Appendix C: It consists of rules based on concept lattice for each class with center 'W'. A sample for class B of GPCR family with Center 'W' is appended in the thesis and rest are in the CD (due to space constraint).

Appendix D: Performance Evaluation of the multilevel classifier, Rough Set Protein Classifier.

APPENDIX A

Sequence Arithmetic module generates three types of Sets namely A-Set, D-Set and P-Set. They are generated for each class within a family, and for each family. This information is used to frame Sequence Arithmetic Rules and they are stored in Sequence Arithmetic Knowledge Database. Section A1 gives the details of Myoglobin Dataset. Section A2 gives the details of GPCR Dataset.

Table A1 consists of Class Id, A-Set, D-Set and P-Set of the Myoglobin family

A-Set(C): It is the union of all A-Set of all sequences belonging to the class 'C'.

D-Set(C): Σ -A-Set(C).

P-Set(C): It is the intersection of all A-Set of all sequences belonging to the class 'C'.

A-Set(C) - P-Set(C) are ambiguous amino acids.

Example: Consider an arbitrary sequence which belongs to class 2 (row 2) of Myoglobin family.

Class Id : 2

A-Set(2) : {A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W, Y}

D-Set(2) : {B,Z}

P-Set(2) : {A,D,E,F,G,H,K,L,M,N,P,Q,R,S,T,V,W}

A1: MYOGLOBIN DATASET

TABLE A1: A-SET, D-SET AND P-SET OF MYOGLOBIN DATASET

S. No	Class Id	A-SET	D-SET	P-SET
1.	1	{A,B,C,D,E,F,G,H,I,K,L,M,N ,P,Q,R,S,T,V,W, Y,Z}	{ }	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W }
2.	2	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W, Y}	{B,Z}	{A,D,E,F,G,H,K,L,M,N,P, Q,R,S,T,V,W}
3.	3	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,T,V,W}	{B,C,Y,Z]	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W,X}
4.	4	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,T,V,W, Y}	{B,C,Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W}
5.	5	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W, Y}	{B,Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W}
6.	7	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B, Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W,Y}
7.	AAF	{A,D,E,F,G,H,I,K,L,M,Q,S,V, Y}	{B,C,N,P,R,T,W,Z}	{A,E,F,H,I,K,L,M,S,V,Y}
8.	AAH	{A,C,D,E,F,G,H,I,K,L,M,N,P,	{B,Z}	{A,C,D,E,F,G,H,I,K,L,M,

*Appendix A is also available in CD with title Appendix A.doc

S. No	Class Id	A-SET	D-SET	P-SET
		Q,R,S,T,V,W,Y}		N,P,Q,R,S,T,V,W,Y}
9.	AAK	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,C,D,E,F,G,H,I,K,L,M, N,P,Q,R,S,T,V,W,Y}
10.	AAL	{F,G,H,I,P,Q,S,V}	{A,B,C,D,E,K,L,M,N, R,T,W,Y,Z}	{}
11.	AAM	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,Y}	{B,W, Z}	{A,C,D,E,F,G,H,I,K,L,M, N,P,Q,R,S,T,V,Y}
12.	AAN	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,Y}	{B,W, Z}	{F,G,H,I,Q,S,V}
13.	AAO	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,V, Y}	{B,C,T,W,Z}	{S}
14.	AAP	{E,F,G,H,I,K,L,P,Q,S,V,Y }	{A,B,C,D, M,N, R,T, W,Z}	{F,I}
15.	AAQ	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W, Y}	{B,Z}	{}
16.	AAR	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,C,D,E,F,G,H,I,K,L,M, N,P,Q,R,S,T,V,W,Y}
17.	AAS	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,T,V,W,Y}	{B,C,Z}	{F,G,H,I,Q,S,V}
18.	B	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,T,V,Y}	{B,C,W, Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,Y}
19.	BAA	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,T,V,W,Y}	{B,C, Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W,Y}
20.	BAB	{A,E,F,G,H,I,K,L,M,N,P,Q,R, S,T,V,Y}	{B,C,D, W, Z}	{A,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,Y}
21.	BAC	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,C,D,E,F,G,H,I,K,L,M, N,P,Q,R,S,T,V,W,Y}
22.	CAA	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,D,F,G,L,P,Q,R,S,V}
23.	CAB	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,T,V,W,Y}	{B,C, Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W,Y}
24.	CAD	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,Y}	{B,W,Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,Y}
25.	CAF	{A,D,E,F,G,H,I,K,L,M,N,P,Q, R,S,T,V,W,Y}	{B,C,Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W,Y}
26.	GG	{A,B,C,D,E,F,G,H,I,K,L,M,N ,P,Q,R,S,T,V,W, Y,Z}	{ }	{A,D,E,F,G,I,K,L,M,P,Q,R ,S,V,W}
27.	H	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W,Y}
28.	I	{A,D,E,F,G,H,I,K,Q,S,V}	{B,C,L,M,N,P,R,T,W, Y,Z}	{A,D,E,F,G,H,I,K,Q,S,V}
29.	J	{A,B,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{C, Z}	{A,D,E,F,G,H,I,K,L,N,P,Q ,R,S,T,V,W}
30.	MY	{A,B,C,D,E,F,G,H,I,K,L,M,N ,P,Q,R,S,T,V,W,Y}	{Z}	{A,D,E,F,G,H,I,K,L,M,P, Q,R,S,T,V,W,Y}
31.	NP	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,Y}
32.	O	{A,B,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{C,Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,W}
33.	Q	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,D,E,F,G,H,I,K,L,M,N, P,Q,R,S,T,V,Y}
34.	Q7M	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,D,E,F,G,H,I,K,L,N,P,Q ,R,S,T,V,W}
35.	S	{A,C,D,E,F,G,H,I,K,L,M,N,P, Q,R,S,T,V,W,Y}	{B,Z}	{A,C,D,E,F,G,H,I,K,L,M, N,P,Q,R,S,T,V,W,Y}
36.	TO	{A,C,D,E,F,G,H,I,K,L,M,P,Q, R,S,T,V,W,Y}	{B,N,Z}	{A,C,D,E,F,G,H,I,K,L,M,P ,Q,R,S,T,V,W,Y}

A(Myoglobin) = {A,B,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y,Z}

D(Myoglobin) = { }

P(Myoglobin) = { }

A2: GPCR DATASET

Table A2 consists of Class Id, A-Set, D-Set and P-Set of the GPCR family.

TABLE A2: A-SET , P-SET AND D-SET FOR GPCR DATASET

Class	A-Set	D-Set	P-Set
A	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
B	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
C	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
D	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,V,W,Y}
E	{ A,C,D,E,F,G,H,I,J,K,L M,N,P,Q,R,S,T,V,W,Y}	{B,Z}	{ A,C,D,E,F,G,H,I,J,K,L M,N,P,Q,R,S,T,V,W,Y}

A(GPCR) = {A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

D(GPCR) = {B,Z}

P(GPCR) = { A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

For a given unknown sequence ‘s’ the given rules are followed to identify the family information.

Sequence Arithmetic Rules

Rule 1: If D(F) $\not\subset$ D(s), then do not search in Family F.

Rule 2: If P(F) $\not\subset$ A(s), then do not search in Family F.

Rule 3: If D(F) $\subseteq$ D(s), F is possible search space.

Rule 4: If P(F) $\subseteq$ A(s), F may be potential Family where ‘s’ can be.

If an unknown string belongs to more than one family, then obtain the relative complement of $P(F_1)$ of corresponding family with the A(s). The family with less number of elements in relative complement set is selected for further processing.

APPENDIX B

B1: Myoglobin family

The P-Table for the Myoglobin family is given in Table P1. Distinct P-Sets identified were 17 as given in Appendix A(A1), therefore the predominant attributes are P1,..,P17.

The P-Table entries are: if $P(C) = P_i$ then P-Table(Class, P_i) = 1, otherwise 0.
Example: Consider P(AAF)= P4, therefore P-Table(AAF, P_4) = 1.

Table P1: P-Table for the Myoglobin family

	Class Id	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	7	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	AAF	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	AAH	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	AAK	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
10	AAL	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	AAM	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
12	AAN	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
13	AAO	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
14	AAP	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
15	AAQ	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
16	AAR	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
17	AAS	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
18	B	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
19	BAA	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	BAB	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
21	BAC	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
22	CAA	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
23	CAB	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	CAD	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
25	CAF	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	GG	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
27	H	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	I	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
29	J	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
30	MY	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
31	NP	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
32	O	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	Q	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
34	Q7M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
35	S	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
36	TO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

*Appendix B is also available as folder in CD with title Appendix B

RDT rules for Myoglobin Family

Rules are of the form
[Attributes value] ⇒ [Class]{Class}

Example

P10, P20, P30, P40, P50, P60, P70, P81 → 11

It is interpreted as: If P1 is 0, P2 is 0, P3 is 0, P4 is 0, P5 is 0, P6 is 0, P7 is 0, P8 is 1 then class is Class 11.

Class Ambiguity appears when more than one class occurs and it can be reduced using spatial information

P11→1, 4,5,29,32,34 is interpreted as

If P1 is 1 then the output is multi valued class/ classes, i.e., 1,4,5,29,32,34.

P10,P21→ 2

P10,P20,P31→3

P10,P20,P30,P40,P51→7

P10,P20,P30,P40,P50,P60,P70,P81→11

P10, P20, P30,P40,P50,P60,P70,P80,P91→12

P10, P20, P30, P40, P50, P60, P70, P80, P90, P101→13

P10,P20,P30,P40,P50,P60,P70,P80,P90,P100,P111→14

P10,P20,P30,P40,P50,P60,P70,P80,P90,P100,P110,P121→17

P10,P20,P30,P40,P50,P60,P70,P80,P90,P100,P110,P120,P130,P140,P151→22

P10,P20,P30,P40,P50,P60,P70,P80,P90,P100,P110,P120,P130,P140,P150,P161→26

P10,P20,P30,P40,P50,P60,P70,P80,P90,P100,P110,P120,P130,P140,P150,P160,P171→28

P11→1,4,5,29,32,34

P10,P20,P30,P41→ 6,19,23,25,27,30

P10,P20,P30,P40,P50,P61→8,9,16,21,35,36

P10,P20,P30,P40,P50,P60,P71→10,15

P10,P20,P30,P40,P50,P60,P70,P80,P90,P100,P110,P120,P131→18,24,33

P10,P20,P30,P40,P50,P60,P70,P80,P90,P100,P110,P120,P130,P141→20,31

B2: RDT/Neighborhood Association rules for GPCR Family

Rules generated from the neighborhood Associations module after applying RDT are stored in the derived database known as Neighborhood Associations Rules Database(NARD). They are similar to RDT rules. They have been used for extracting Class information.

Rules for GPCR datasets

Rules is of the form
[Attributes value] ⇒ [Class]

Class A, B, C, D, E are 1,2,3,4,5, respectively.

Example

[E0 K0 Y0 G0 N0 C0] ⇒1 is interpreted as

If E=0, K=0, Y=0, G=0, N=0, C=0 then it belongs to class A

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D0	E0	R0	P0	N0															==>1
D0	E0	R0	P0	N1	C0	K0													==>1
D0	E0	R0	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0							==>3
D0	E0	R0	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y1							==>2
D0	E0	R0	P0	N1	C0	K1	W0	M0	H0	Q1									==>1
D0	E0	R0	P0	N1	C0	K1	W0	M0	H1										==>3
D0	E0	R0	P0	N1	C0	K1	W0	M1											==>1
D0	E0	R0	P0	N1	C1	K0	W0	M0	H0										==>1
D0	E0	R0	P0	N1	C1	K0	W0	M0	H1										==>3
D0	E0	R0	P0	N1	C1	K0	W0	M1											==>1
D0	E0	R0	P0	N1	C1	K1													==>1
D0	E0	R0	P1	N0	C0	K0	W0	M0											==>1
D0	E0	R0	P1	N0	C0	K0	W0	M1	H0	Q0	I1	Y0	T0						==>1
D0	E0	R0	P1	N0	C0	K0	W0	M1	H0	Q0	I1	Y0	T1						==>2
D0	E0	R0	P1	N0	C0	K0	W0	M1	H0	Q0	I1	Y1							==>1
D0	E0	R0	P1	N0	C0	K0	W0	M1	H0	Q1									==>1
D0	E0	R0	P1	N0	C0	K0	W0	M1	H1										==>1
D0	E0	R0	P1	N0	C0	K0	W1	M0	H0										==>3
D0	E0	R0	P1	N0	C0	K0	W1	M0	H1										==>1
D0	E0	R0	P1	N0	C0	K1	W0	M0	H0	Q0	I0								==>3
D0	E0	R0	P1	N0	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A0					==>1
D0	E0	R0	P1	N0	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A1	F0				==>3
D0	E0	R0	P1	N0	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A1	F1				==>1
D0	E0	R0	P1	N0	C0	K1	W0	M0	H0	Q0	I1	Y1							==>1
D0	E0	R0	P1	N0	C0	K1	W0	M0	H1										==>1
D0	E0	R0	P1	N0	C0	K1	W0	M1											==>1
D0	E0	R0	P1	N0	C1	K0	W0	M0	H0	Q0	I0								==>1
D0	E0	R0	P1	N0	C1	K0	W0	M0	H0	Q0	I1	Y0	T1	A0	F1	V1	S0		==>1
D0	E0	R0	P1	N0	C1	K0	W0	M0	H0	Q0	I1	Y0	T1	A0	F1	V1	S1		==>3

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D0	E0	R0	P1	N0	C1	K0	W0	M0	H0	Q0	I1	Y0	T1	A1					==>1
D0	E0	R0	P1	N0	C1	K0	W0	M0	H0	Q0	I1	Y1							==>1
D0	E0	R0	P1	N0	C1	K0	W0	M0	H0	Q1									==>1
D0	E0	R0	P1	N0	C1	K0	W0	M0	H1										==>1
D0	E0	R0	P1	N0	C1	K0	W0	M1											==>1
D0	E0	R0	P1	N0	C1	K0	W1	M0	H0	Q0	I0	Y0	T0						==>3
D0	E0	R0	P1	N0	C1	K0	W1	M0	H0	Q0	I0	Y0	T1						==>2
D0	E0	R0	P1	N0	C1	K1	W0	M0											==>1
D0	E0	R0	P1	N0	C1	K1	W0	M1	H0	Q0									==>2
D0	E0	R0	P1	N0	C1	K1	W0	M1	H0	Q1									==>1
D0	E0	R0	P1	N0	C1	K1	W0	M1	H1										==>1
D0	E0	R0	P1	N1	C0	K0	W1	M0	H0	Q0									==>1
D0	E0	R0	P1	N1	C0	K0	W1	M0	H0	Q1	I0								==>3
D0	E0	R0	P1	N1	C0	K0	W1	M0	H0	Q1	I1								==>1
D0	E0	R0	P1	N1	C0	K1													==>1
D0	E0	R0	P1	N1	C1	K0													==>1
D0	E0	R0	P1	N1	C1	K1	W0												==>1
D0	E0	R0	P1	N1	C1	K1	W1	M0											==>2
D0	E0	R0	P1	N1	C1	K1	W1	M1											==>1
D0	E0	R1	P0	N0	C0	K0	W0	M0	H0	Q0	I0								==>1
D0	E0	R1	P0	N0	C0	K0	W0	M0	H0	Q0	I1								==>3
D0	E0	R1	P0	N0	C0	K0	W0	M0	H0	Q1									==>1
D0	E0	R1	P0	N0	C0	K0	W0	M0	H1										==>3
D0	E0	R1	P0	N0	C0	K0	W0	M1											==>2
D0	E0	R1	P0	N0	C0	K0	W1												==>1
D0	E0	R1	P0	N0	C0	K1													==>1
D0	E0	R1	P0	N0	C1	K0	W0	M0											==>1
D0	E0	R1	P0	N0	C1	K0	W0	M1	H0	Q0									==>1
D0	E0	R1	P0	N0	C1	K0	W0	M1	H0	Q1	I1	Y0	T0						==>1
D0	E0	R1	P0	N0	C1	K0	W0	M1	H0	Q1	I1	Y0	T1	A1	F1	V1	S1	G0	==>1
D0	E0	R1	P0	N0	C1	K0	W0	M1	H0	Q1	I1	Y0	T1	A1	F1	V1	S1	G1	==>3
D0	E0	R1	P0	N0	C1	K0	W1												==>5
D0	E0	R1	P0	N0	C1	K1	W0	M0	H0	Q0	I1	Y0	T1	A0					==>2
D0	E0	R1	P0	N0	C1	K1	W0	M0	H0	Q0	I1	Y0	T1	A1					==>1
D0	E0	R1	P0	N0	C1	K1	W0	M0	H0	Q0	I1	Y1							==>1
D0	E0	R1	P0	N0	C1	K1	W0	M0	H1										==>1
D0	E0	R1	P0	N0	C1	K1	W0	M1	H0	Q1	I1	Y0	T0						==>1
D0	E0	R1	P0	N0	C1	K1	W0	M1	H0	Q1	I1	Y0	T1						==>2
D0	E0	R1	P0	N0	C1	K1	W0	M1	H1										==>1
D0	E0	R1	P0	N0	C1	K1	W1												==>1
D0	E0	R1	P0	N1	C0	K0	W0	M0	H0	Q0									==>2
D0	E0	R1	P0	N1	C0	K0	W0	M0	H0	Q1									==>1
D0	E0	R1	P0	N1	C0	K0	W0	M0	H1										==>1
D0	E0	R1	P0	N1	C0	K0	W0	M1											==>1
D0	E0	R1	P0	N1	C0	K0	W1												==>1
D0	E0	R1	P0	N1	C0	K1	W0	M0	H0	Q0	I0								==>1
D0	E0	R1	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0							==>1
D0	E0	R1	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y1							==>2
D0	E0	R1	P0	N1	C0	K1	W0	M1											==>1
D0	E0	R1	P0	N1	C1	K0	W0	M0	H0	Q0	I1	Y0							==>1
D0	E0	R1	P0	N1	C1	K0	W0	M0	H0	Q0	I1	Y1							==>2
D0	E0	R1	P0	N1	C1	K0	W0	M0	H0	Q1									==>1
D0	E0	R1	P0	N1	C1	K0	W0	M1											==>1
D0	E0	R1	P0	N1	C1	K0	W1	M0	H0	Q0	I1	Y0	T0	A0					==>2
D0	E0	R1	P0	N1	C1	K0	W1	M0	H0	Q0	I1	Y0	T0	A1					==>5
D0	E0	R1	P0	N1	C1	K0	W1	M1											==>5
D0	E0	R1	P0	N1	C1	K1	W0												==>2
D0	E0	R1	P0	N1	C1	K1	W1												==>1
D0	E0	R1	P1	N0	C0														==>1

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D0	E0	R1	P1	N0	C1	K0													==>1
D0	E0	R1	P1	N0	C1	K1	W0	M0	H0										==>1
D0	E0	R1	P1	N0	C1	K1	W0	M0	H1	Q1									==>1
D0	E0	R1	P1	N0	C1	K1	W0	M1											==>1
D0	E0	R1	P1	N1	C0	K0													==>1
D0	E0	R1	P1	N1	C0	K1	W0												==>1
D0	E0	R1	P1	N1	C0	K1	W1												==>3
D0	E0	R1	P1	N1	C1	K0													==>1
D0	E0	R1	P1	N1	C1	K1	W0	M0	H0	Q0	I0								==>2
D0	E0	R1	P1	N1	C1	K1	W0	M0	H0	Q0	I1								==>1
D0	E0	R1	P1	N1	C1	K1	W0	M0	H0	Q1	I0								==>2
D0	E0	R1	P1	N1	C1	K1	W0	M0	H0	Q1	I1								==>1
D0	E0	R1	P1	N1	C1	K1	W0	M0	H1										==>1
D0	E0	R1	P1	N1	C1	K1	W0	M1	H0										==>1
D0	E0	R1	P1	N1	C1	K1	W0	M1	H1										==>2
D0	E0	R1	P1	N1	C1	K1	W1												==>5
D0	E1	R0	P0	N0	C0	K0	W0	M0											==>1
D0	E1	R0	P0	N0	C0	K0	W0	M1	H0										==>1
D0	E1	R0	P0	N0	C0	K0	W0	M1	H1										==>2
D0	E1	R0	P0	N0	C0	K0	W1	M0	H0										==>3
D0	E1	R0	P0	N0	C0	K0	W1	M0	H1										==>1
D0	E1	R0	P0	N0	C0	K1	W0	M0	H0										==>3
D0	E1	R0	P0	N0	C0	K1	W0	M0	H1										==>1
D0	E1	R0	P0	N0	C0	K1	W0	M1	H0	Q0									==>1
D0	E1	R0	P0	N0	C0	K1	W0	M1	H0	Q1									==>2
D0	E1	R0	P0	N0	C1	K0	W0												==>1
D0	E1	R0	P0	N0	C1	K0	W1												==>2
D0	E1	R0	P0	N0	C1	K1	W0	M0	H0	Q0	I0								==>2
D0	E1	R0	P0	N0	C1	K1	W0	M0	H0	Q0	I1								==>1
D0	E1	R0	P0	N0	C1	K1	W0	M1											==>2
D0	E1	R0	P0	N1	C0	K0	W0	M0											==>1
D0	E1	R0	P0	N1	C0	K0	W0	M1	H0	Q0	I1	Y0							==>3
D0	E1	R0	P0	N1	C0	K0	W0	M1	H0	Q0	I1	Y1							==>2
D0	E1	R0	P0	N1	C0	K0	W0	M1	H1										==>1
D0	E1	R0	P0	N1	C0	K1	W0	M0											==>3
D0	E1	R0	P0	N1	C0	K1	W0	M1											==>1
D0	E1	R0	P0	N1	C0	K1	W1												==>1
D0	E1	R0	P0	N1	C1	K0	W0	M0	H0										==>1
D0	E1	R0	P0	N1	C1	K0	W0	M0	H1										==>3
D0	E1	R0	P0	N1	C1	K0	W0	M1	H0	Q0									==>2
D0	E1	R0	P0	N1	C1	K0	W0	M1	H0	Q1									==>1
D0	E1	R0	P0	N1	C1	K1													==>1
D0	E1	R0	P1	N0	C0	K0	W0												==>1
D0	E1	R0	P1	N0	C0	K0	W1	M0	H0	Q0	I1	Y0							==>3
D0	E1	R0	P1	N0	C0	K0	W1	M0	H0	Q0	I1	Y1							==>1
D0	E1	R0	P1	N0	C0	K0	W1	M0	H0	Q1									==>3
D0	E1	R0	P1	N0	C0	K1													==>3
D0	E1	R0	P1	N0	C1	K0	W0	M0											==>3
D0	E1	R0	P1	N0	C1	K0	W0	M1											==>1
D0	E1	R0	P1	N0	C1	K1													==>1
D0	E1	R0	P1	N1	C0	K0	W0	M0	H0	Q1	I1	Y0	T1	A1	F1	V0			==>1
D0	E1	R0	P1	N1	C0	K0	W0	M0	H0	Q1	I1	Y0	T1	A1	F1	V1			==>3
D0	E1	R0	P1	N1	C0	K0	W0	M0	H1	Q0	I0	Y1	T0						==>1
D0	E1	R0	P1	N1	C0	K0	W0	M0	H1	Q0	I0	Y1	T1						==>2
D0	E1	R0	P1	N1	C0	K0	W0	M1	H0										==>3
D0	E1	R0	P1	N1	C0	K0	W0	M1	H1										==>1
D0	E1	R0	P1	N1	C0	K0	W1												==>3
D0	E1	R0	P1	N1	C0	K1	W0												==>1

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D0	E1	R0	P1	N1	C0	K1	W1												==>2
D0	E1	R0	P1	N1	C1	K0	W0	M0	H0	Q0	I1	Y0	T1	A0					==>1
D0	E1	R0	P1	N1	C1	K0	W0	M0	H0	Q0	I1	Y0	T1	A1	F1	V1	S0		==>1
D0	E1	R0	P1	N1	C1	K0	W0	M0	H0	Q0	I1	Y0	T1	A1	F1	V1	S1		==>3
D0	E1	R0	P1	N1	C1	K0	W0	M0	H0	Q1									==>1
D0	E1	R0	P1	N1	C1	K0	W0	M1											==>1
D0	E1	R0	P1	N1	C1	K1													==>1
D0	E1	R1	P0	N0	C0	K0	W0	M1	H0										==>2
D0	E1	R1	P0	N0	C0	K0	W0	M1	H1										==>1
D0	E1	R1	P0	N0	C0	K0	W1												==>2
D0	E1	R1	P0	N0	C0	K1													==>1
D0	E1	R1	P0	N0	C1														==>1
D0	E1	R1	P0	N1	C0	K0	W0	M0	H0	Q0									==>1
D0	E1	R1	P0	N1	C0	K0	W0	M0	H0	Q1									==>3
D0	E1	R1	P0	N1	C0	K0	W0	M1											==>1
D0	E1	R1	P0	N1	C0	K0	W1	M0											==>3
D0	E1	R1	P0	N1	C0	K0	W1	M1	H0										==>1
D0	E1	R1	P0	N1	C0	K0	W1	M1	H1										==>3
D0	E1	R1	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A0	F1	V0			==>1
D0	E1	R1	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A1					==>1
D0	E1	R1	P0	N1	C0	K1	W0	M0	H1										==>1
D0	E1	R1	P0	N1	C0	K1	W0	M1	H0	Q0	I1	Y0	T0	A1	F0				==>1
D0	E1	R1	P0	N1	C0	K1	W0	M1	H0	Q0	I1	Y0	T0	A1	F1				==>3
D0	E1	R1	P0	N1	C0	K1	W0	M1	H0	Q1									==>3
D0	E1	R1	P0	N1	C0	K1	W0	M1	H1										==>3
D0	E1	R1	P0	N1	C1	K0	W0	M0	H0										==>1
D0	E1	R1	P0	N1	C1	K0	W0	M0	H1										==>2
D0	E1	R1	P0	N1	C1	K0	W0	M1											==>1
D0	E1	R1	P0	N1	C1	K0	W1	M0											==>3
D0	E1	R1	P0	N1	C1	K0	W1	M1											==>2
D0	E1	R1	P0	N1	C1	K1	W0	M0	H0	Q0									==>1
D0	E1	R1	P0	N1	C1	K1	W0	M0	H0	Q1	I1	Y0							==>3
D0	E1	R1	P0	N1	C1	K1	W0	M0	H0	Q1	I1	Y1							==>1
D0	E1	R1	P0	N1	C1	K1	W0	M0	H1										==>2
D0	E1	R1	P0	N1	C1	K1	W1												==>2
D0	E1	R1	P1	N0	C0	K0													==>1
D0	E1	R1	P1	N0	C0	K1	W0	M0	H0	Q0	I0								==>2
D0	E1	R1	P1	N0	C0	K1	W0	M0	H0	Q0	I1								==>3
D0	E1	R1	P1	N0	C0	K1	W0	M0	H0	Q1									==>1
D0	E1	R1	P1	N0	C0	K1	W0	M0	H1										==>1
D0	E1	R1	P1	N0	C0	K1	W1	M0											==>2
D0	E1	R1	P1	N0	C0	K1	W1	M1	H0	Q0	I1	Y0	T0						==>1
D0	E1	R1	P1	N0	C0	K1	W1	M1	H0	Q0	I1	Y0	T1						==>3
D0	E1	R1	P1	N0	C1	K0	W0	M0	H0	Q0									==>1
D0	E1	R1	P1	N0	C1	K0	W0	M0	H0	Q1									==>3
D0	E1	R1	P1	N0	C1	K0	W0	M0	H1										==>3
D0	E1	R1	P1	N0	C1	K0	W0	M1											==>1
D0	E1	R1	P1	N0	C1	K0	W1	M0	H0	Q1	I0								==>1
D0	E1	R1	P1	N0	C1	K0	W1	M0	H0	Q1	I1								==>2
D0	E1	R1	P1	N0	C1	K0	W1	M0	H1										==>2
D0	E1	R1	P1	N0	C1	K0	W1	M1											==>3
D0	E1	R1	P1	N0	C1	K1	W0	M0	H0	Q0	I0								==>2
D0	E1	R1	P1	N0	C1	K1	W0	M0	H0	Q0	I1	Y0							==>1
D0	E1	R1	P1	N0	C1	K1	W0	M0	H0	Q0	I1	Y1							==>2
D0	E1	R1	P1	N0	C1	K1	W0	M0	H0	Q1									==>2
D0	E1	R1	P1	N0	C1	K1	W0	M1											==>3
D0	E1	R1	P1	N0	C1	K1	W1												==>2
D0	E1	R1	P1	N1	C0	K0	W0	M0	H0										==>2
D0	E1	R1	P1	N1	C0	K0	W0	M0	H1										==>1

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D0	E1	R1	P1	N1	C0	K0	W0	M1											==>1
D0	E1	R1	P1	N1	C0	K0	W1	M0	H0	Q0									==>1
D0	E1	R1	P1	N1	C0	K0	W1	M0	H0	Q1									==>3
D0	E1	R1	P1	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A0					==>3
D0	E1	R1	P1	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A1					==>1
D0	E1	R1	P1	N1	C0	K1	W0	M1											==>1
D0	E1	R1	P1	N1	C1	K0	W0	M0	H0										==>1
D0	E1	R1	P1	N1	C1	K0	W0	M0	H1	Q0									==>1
D0	E1	R1	P1	N1	C1	K0	W0	M0	H1	Q1									==>2
D0	E1	R1	P1	N1	C1	K0	W0	M1	H0										==>1
D0	E1	R1	P1	N1	C1	K0	W0	M1	H1										==>2
D0	E1	R1	P1	N1	C1	K0	W1	M0											==>2
D0	E1	R1	P1	N1	C1	K0	W1	M1											==>1
D0	E1	R1	P1	N1	C1	K1	W0	M0	H0										==>1
D0	E1	R1	P1	N1	C1	K1	W0	M0	H1										==>2
D0	E1	R1	P1	N1	C1	K1	W0	M1											==>2
D0	E1	R1	P1	N1	C1	K1	W1	M0	H0										==>2
D0	E1	R1	P1	N1	C1	K1	W1	M0	H1										==>1
D0	E1	R1	P1	N1	C1	K1	W1	M1											==>2
D1	E0	R0	P0	N0	C0	K0	W0	M0											==>1
D1	E0	R0	P0	N0	C0	K0	W0	M1	H0	Q0	I1	Y0	T1	A0					==>4
D1	E0	R0	P0	N0	C0	K0	W0	M1	H0	Q0	I1	Y0	T1	A1					==>1
D1	E0	R0	P0	N0	C0	K1	W0	M0											==>1
D1	E0	R0	P0	N0	C0	K1	W0	M1											==>4
D1	E0	R0	P0	N0	C1	K0													==>1
D1	E0	R0	P0	N0	C1	K1	W0												==>1
D1	E0	R0	P0	N0	C1	K1	W1												==>2
D1	E0	R0	P0	N1	C0	K0	W0	M0	H0	Q0									==>1
D1	E0	R0	P0	N1	C0	K0	W0	M0	H0	Q1									==>2
D1	E0	R0	P0	N1	C0	K0	W0	M1	H0										==>2
D1	E0	R0	P0	N1	C0	K0	W0	M1	H1										==>1
D1	E0	R0	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T0						==>1
D1	E0	R0	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1						==>3
D1	E0	R0	P0	N1	C0	K1	W0	M0	H0	Q1									==>3
D1	E0	R0	P0	N1	C0	K1	W0	M0	H1										==>3
D1	E0	R0	P0	N1	C0	K1	W0	M1											==>3
D1	E0	R0	P0	N1	C0	K1	W1												==>3
D1	E0	R0	P0	N1	C1														==>1
D1	E0	R0	P1	N0	C0														==>1
D1	E0	R0	P1	N0	C1	K0	W0												==>1
D1	E0	R0	P1	N0	C1	K0	W1												==>2
D1	E0	R0	P1	N0	C1	K1	W0	M0											==>1
D1	E0	R0	P1	N0	C1	K1	W0	M1	H0	Q0	I0								==>2
D1	E0	R0	P1	N0	C1	K1	W0	M1	H0	Q0	I1								==>1
D1	E0	R0	P1	N0	C1	K1	W1												==>3
D1	E0	R0	P1	N1	C0	K0	W0	M0											==>1
D1	E0	R0	P1	N1	C0	K0	W0	M1	H0	Q0	I1	Y0							==>1
D1	E0	R0	P1	N1	C0	K0	W0	M1	H0	Q0	I1	Y1							==>2
D1	E0	R0	P1	N1	C0	K0	W0	M1	H1										==>2
D1	E0	R0	P1	N1	C0	K1	W0	M0											==>3
D1	E0	R0	P1	N1	C0	K1	W0	M1											==>1
D1	E0	R0	P1	N1	C1	K0	W0	M0											==>1
D1	E0	R0	P1	N1	C1	K0	W0	M1	H0										==>1
D1	E0	R0	P1	N1	C1	K0	W0	M1	H1										==>2
D1	E0	R0	P1	N1	C1	K0	W1	M0											==>1
D1	E0	R0	P1	N1	C1	K0	W1	M1	H0										==>2
D1	E0	R0	P1	N1	C1	K0	W1	M1	H1										==>1
D1	E0	R0	P1	N1	C1	K1	W0	M0	H0										==>1
D1	E0	R0	P1	N1	C1	K1	W0	M0	H1										==>3

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D1	E0	R0	P1	N1	C1	K1	W0	M1											==>1
D1	E0	R0	P1	N1	C1	K1	W1	M0	H0	Q0	I1	Y1	T1	A0					==>1
D1	E0	R0	P1	N1	C1	K1	W1	M0	H0	Q0	I1	Y1	T1	A1					==>2
D1	E0	R0	P1	N1	C1	K1	W1	M0	H1										==>2
D1	E0	R1	P0	N0	C0	K0													==>1
D1	E0	R1	P0	N0	C0	K1	W0	M0	H0	Q0									==>3
D1	E0	R1	P0	N0	C0	K1	W0	M0	H0	Q1									==>1
D1	E0	R1	P0	N0	C0	K1	W0	M1											==>1
D1	E0	R1	P0	N0	C1	K0	W0												==>1
D1	E0	R1	P0	N0	C1	K0	W1	M0											==>2
D1	E0	R1	P0	N0	C1	K0	W1	M1											==>1
D1	E0	R1	P0	N0	C1	K1	W0												==>1
D1	E0	R1	P0	N0	C1	K1	W1	M0											==>2
D1	E0	R1	P0	N0	C1	K1	W1	M1	H0	Q0	I1	Y0							==>2
D1	E0	R1	P0	N0	C1	K1	W1	M1	H0	Q0	I1	Y1							==>3
D1	E0	R1	P0	N1	C0	K0	W0												==>1
D1	E0	R1	P0	N1	C0	K0	W1												==>3
D1	E0	R1	P0	N1	C0	K1	W0	M0	H0	Q0									==>1
D1	E0	R1	P0	N1	C0	K1	W0	M0	H0	Q1									==>2
D1	E0	R1	P0	N1	C0	K1	W0	M1											==>1
D1	E0	R1	P0	N1	C1	K0	W0	M0											==>1
D1	E0	R1	P0	N1	C1	K0	W0	M1											==>2
D1	E0	R1	P0	N1	C1	K0	W1												==>2
D1	E0	R1	P0	N1	C1	K1	W0	M0											==>2
D1	E0	R1	P0	N1	C1	K1	W0	M1											==>1
D1	E0	R1	P0	N1	C1	K1	W1												==>2
D1	E0	R1	P1	N0	C0	K0	W0	M0	H0	Q0	I0								==>1
D1	E0	R1	P1	N0	C0	K0	W0	M0	H0	Q0	I1	Y0	T0						==>1
D1	E0	R1	P1	N0	C0	K0	W0	M0	H0	Q0	I1	Y0	T1						==>3
D1	E0	R1	P1	N0	C0	K0	W0	M0	H1										==>1
D1	E0	R1	P1	N0	C0	K0	W0	M1											==>1
D1	E0	R1	P1	N0	C0	K0	W1	M0	H0	Q0									==>2
D1	E0	R1	P1	N0	C0	K0	W1	M0	H0	Q1									==>1
D1	E0	R1	P1	N0	C0	K1													==>1
D1	E0	R1	P1	N0	C1	K0	W0	M0											==>1
D1	E0	R1	P1	N0	C1	K0	W0	M1											==>2
D1	E0	R1	P1	N0	C1	K0	W1												==>2
D1	E0	R1	P1	N0	C1	K1	W1	M0	H0										==>2
D1	E0	R1	P1	N0	C1	K1	W1	M0	H1										==>1
D1	E0	R1	P1	N0	C1	K1	W1	M1											==>2
D1	E0	R1	P1	N1	C0	K0	W0	M0	H0	Q0	I0								==>1
D1	E0	R1	P1	N1	C0	K0	W0	M0	H0	Q0	I1								==>3
D1	E0	R1	P1	N1	C0	K0	W0	M0	H1										==>1
D1	E0	R1	P1	N1	C0	K0	W0	M1	H0	Q0									==>2
D1	E0	R1	P1	N1	C0	K0	W0	M1	H0	Q1									==>3
D1	E0	R1	P1	N1	C0	K1	W0	M0	H0										==>3
D1	E0	R1	P1	N1	C0	K1	W0	M0	H1										==>1
D1	E0	R1	P1	N1	C1	K0	W0	M0											==>1
D1	E0	R1	P1	N1	C1	K0	W0	M1											==>2
D1	E0	R1	P1	N1	C1	K0	W1												==>2
D1	E0	R1	P1	N1	C1	K1	W0	M0	H0										==>3
D1	E0	R1	P1	N1	C1	K1	W0	M0	H1										==>1
D1	E0	R1	P1	N1	C1	K1	W0	M1	H0										==>1
D1	E0	R1	P1	N1	C1	K1	W0	M1	H1										==>2
D1	E0	R1	P1	N1	C1	K1	W1												==>2
D1	E1	R0	P0	N0	C0	K0													==>1
D1	E1	R0	P0	N0	C0	K1	W0	M0	H0	Q0	I1	Y0							==>1
D1	E1	R0	P0	N0	C0	K1	W0	M0	H0	Q0	I1	Y1	T0						==>3

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D1	E1	R0	P0	N0	C0	K1	W0	M0	H0	Q0	I1	Y1	T1						==>2
D1	E1	R0	P0	N0	C0	K1	W1												==>1
D1	E1	R0	P0	N1	C0	K0	W0	M0	H0	Q0									==>1
D1	E1	R0	P0	N1	C0	K0	W0	M0	H0	Q1									==>3
D1	E1	R0	P0	N1	C0	K0	W0	M0	H1										==>3
D1	E1	R0	P0	N1	C0	K0	W0	M1											==>1
D1	E1	R0	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A1	F0				==>2
D1	E1	R0	P0	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A1	F1				==>3
D1	E1	R0	P0	N1	C0	K1	W0	M1											==>2
D1	E1	R0	P0	N1	C1	K0	W0	M0											==>1
D1	E1	R0	P0	N1	C1	K0	W0	M1											==>2
D1	E1	R0	P0	N1	C1	K0	W1												==>2
D1	E1	R0	P0	N1	C1	K1													==>2
D1	E1	R0	P1	N0	C0	K0	W0												==>3
D1	E1	R0	P1	N0	C0	K0	W1												==>2
D1	E1	R0	P1	N0	C0	K1	W0	M0	H0	Q0									==>3
D1	E1	R0	P1	N0	C0	K1	W0	M0	H0	Q1	I0								==>1
D1	E1	R0	P1	N0	C0	K1	W0	M0	H0	Q1	I1	Y0	T1	A1	F0	V1			==>3
D1	E1	R0	P1	N0	C0	K1	W0	M0	H1										==>1
D1	E1	R0	P1	N0	C1	K0	W0												==>3
D1	E1	R0	P1	N0	C1	K0	W1	M0	H0										==>2
D1	E1	R0	P1	N0	C1	K0	W1	M0	H1										==>3
D1	E1	R0	P1	N0	C1	K1	W0	M0											==>1
D1	E1	R0	P1	N0	C1	K1	W0	M1											==>3
D1	E1	R0	P1	N1	C0	K0	W0	M0											==>1
D1	E1	R0	P1	N1	C0	K0	W0	M1											==>2
D1	E1	R0	P1	N1	C0	K0	W1												==>2
D1	E1	R0	P1	N1	C0	K1													==>2
D1	E1	R0	P1	N1	C1	K0	W0	M1	H0										==>1
D1	E1	R0	P1	N1	C1	K0	W0	M1	H1										==>3
D1	E1	R0	P1	N1	C1	K0	W1												==>2
D1	E1	R0	P1	N1	C1	K1	W0	M0	H0	Q0	I1	Y0							==>1
D1	E1	R0	P1	N1	C1	K1	W0	M0	H0	Q0	I1	Y1							==>2
D1	E1	R0	P1	N1	C1	K1	W0	M0	H0	Q1									==>1
D1	E1	R0	P1	N1	C1	K1	W0	M1											==>3
D1	E1	R1	P0	N0	C0	K0													==>1
D1	E1	R1	P0	N0	C0	K1	W0	M0	H0										==>1
D1	E1	R1	P0	N0	C0	K1	W0	M0	H1										==>3
D1	E1	R1	P0	N0	C0	K1	W0	M1											==>3
D1	E1	R1	P0	N0	C1	K0													==>1
D1	E1	R1	P0	N0	C1	K1	W0	M0	H0	Q0									==>1
D1	E1	R1	P0	N0	C1	K1	W0	M0	H0	Q1									==>3
D1	E1	R1	P0	N0	C1	K1	W0	M0	H1										==>1
D1	E1	R1	P0	N0	C1	K1	W1												==>2
D1	E1	R1	P0	N1	C0	K0	W0												==>1
D1	E1	R1	P0	N1	C0	K0	W1												==>2
D1	E1	R1	P0	N1	C0	K1	W0	M0											==>1
D1	E1	R1	P0	N1	C0	K1	W0	M1											==>3
D1	E1	R1	P0	N1	C0	K1	W1	M0	H1	Q1	I1	Y0	T1	A1	F0				==>1
D1	E1	R1	P0	N1	C0	K1	W1	M0	H1	Q1	I1	Y0	T1	A1	F1				==>3
D1	E1	R1	P0	N1	C0	K1	W1	M1	H1	Q1	I1	Y0	T1	A1	F0				==>3
D1	E1	R1	P0	N1	C0	K1	W1	M1	H1	Q1	I1	Y0	T1	A1	F1				==>1
D1	E1	R1	P0	N1	C1	K0													==>2
D1	E1	R1	P0	N1	C1	K1	W0	M0	H0	Q0	I1	Y1	T1	A1	F0				==>2
D1	E1	R1	P0	N1	C1	K1	W1												==>2
D1	E1	R1	P1	N0	C0	K0	W0	M0	H0										==>2
D1	E1	R1	P1	N0	C0	K0	W0	M0	H1										==>1
D1	E1	R1	P1	N0	C0	K0	W1	M0											==>2
D1	E1	R1	P1	N0	C0	K0	W1	M1											==>3

D	E	R	P	N	C	K	W	M	H	Q	I	Y	T	A	F	V	S	G	Class
D1	E1	R1	P1	N0	C0	K1	W0	M0	H0										==>3
D1	E1	R1	P1	N0	C0	K1	W0	M0	H1										==>1
D1	E1	R1	P1	N0	C0	K1	W1												==>3
D1	E1	R1	P1	N0	C1	K0	W0	M0	H0										==>1
D1	E1	R1	P1	N0	C1	K0	W0	M0	H1	Q0									==>3
D1	E1	R1	P1	N0	C1	K0	W0	M0	H1	Q1									==>1
D1	E1	R1	P1	N0	C1	K0	W1	M0	H0	Q0									==>3
D1	E1	R1	P1	N0	C1	K0	W1	M0	H0	Q1									==>2
D1	E1	R1	P1	N0	C1	K0	W1	M0	H1										==>3
D1	E1	R1	P1	N0	C1	K1	W0												==>1
D1	E1	R1	P1	N0	C1	K1	W1												==>2
D1	E1	R1	P1	N1	C0	K0	W0	M0	H1										==>1
D1	E1	R1	P1	N1	C0	K0	W0	M1											==>2
D1	E1	R1	P1	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T0						==>3
D1	E1	R1	P1	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A0					==>3
D1	E1	R1	P1	N1	C0	K1	W0	M0	H0	Q0	I1	Y0	T1	A1					==>2
D1	E1	R1	P1	N1	C1	K0	W0	M0	H0	Q0	I0								==>1
D1	E1	R1	P1	N1	C1	K0	W0	M0	H0	Q0	I1								==>2
D1	E1	R1	P1	N1	C1	K0	W0	M0	H0	Q1									==>3
D1	E1	R1	P1	N1	C1	K0	W0	M1	H0	Q0									==>1
D1	E1	R1	P1	N1	C1	K0	W0	M1	H0	Q1									==>3
D1	E1	R1	P1	N1	C1	K0	W1	M0	H0										==>2
D1	E1	R1	P1	N1	C1	K0	W1	M0	H1										==>3
D1	E1	R1	P1	N1	C1	K0	W1	M1	H0										==>2
D1	E1	R1	P1	N1	C1	K0	W1	M1	H1										==>3
D1	E1	R1	P1	N1	C1	K1	W0	M0	H0	Q0	I1	Y1	T0						==>2
D1	E1	R1	P1	N1	C1	K1	W0	M0	H0	Q0	I1	Y1	T1	A1	F0				==>1
D1	E1	R1	P1	N1	C1	K1	W0	M0	H0	Q0	I1	Y1	T1	A1	F1				==>2
D1	E1	R1	P1	N1	C1	K1	W0	M0	H0	Q1									==>3
D1	E1	R1	P1	N1	C1	K1	W0	M0	H1										==>1
D1	E1	R1	P1	N1	C1	K1	W0	M1											==>2
D1	E1	R1	P1	N1	C1	K1	W1												==>2

APPENDIX C

Concept Lattice Association Rules

Concept lattice is constructed from the spatial information obtained for a given sequences. Concept lattice Association Rules (CAR) are generated from the concept lattice and stored in Concept lattice Association Rules Database (CARD). The rules are generated in the form of XML file, which are converted into Excel Sheets. Family, Class and Center information are appended to the sheet for better readability.

C1: Myoglobin dataset and class 'B' with center 'W'

The family and class identifier (Fid and Cid) with center information is appended to the rules. The generated concept lattice for the Myoglobin family and class 'B' data with center 'W' consists of 24 nodes. The corresponding excel sheet after appending Family ID, Class ID and Center is also given in Table C1. A sample XML file is given below.

The CAR is of the form (Fid, Cid, Center, R-Ante, R-Cons).

Example : (MYO,B,W,{ K,L,G,V},{3,7,2,1,6,10,5,13,9,11,4,8,12}) indicates that for the family Myoglobin and class 'B' with Center 'W', when the Attributes {K,L,G,V} has value '1' then questioned sequence will be closer to sequences in the set {3,7,2,1,6,10,5,13,9,11,4,8,12}.

Table C1 : CAR for Myoglobin Family, Class B with Center 'W'

(Family	Class	Center	{ R-Ante	} { R-Cons	}
(MYO	B	W	{ K,L,G,V	} { 3,7,2,1,6,10,5,13,9,11,4,8,12	}
(MYO	B	W	{ D,K,L,G,V	} { 7,2,6,10,5,13,11,9,4,12	}
(MYO	B	W	{ S,K,L,G,V	} { 3,7,2,6,1,10,13,11,9,4,8	}
(MYO	B	W	{ K,L,G,V,A	} { 3,2,1,4,8,12	}
(MYO	B	W	{ D,K,L,G,V,N	} { 7,10,5,13,11,9	}
(MYO	B	W	{ D,K,L,G,V,A	} { 2,4,12	}
(MYO	B	W	{ K,L,Q,G,V,E	} { 3,7,6,1,10,13,11,9,8,12	}
(MYO	B	W	{ D,K,L,G,V,I	} { 5,11,12	}
(MYO	B	W	{ S,K,L,G,V,A	} { 3,2,1,4,8	}
(MYO	B	W	{ S,D,K,L,G,V	} { 7,2,6,10,13,9,11,4	}
(MYO	B	W	{ D,K,L,Q,G,V,E	} { 7,6,10,13,9,11,12	}
(MYO	B	W	{ S,K,Q,L,G,V,E	} { 3,7,1,6,10,13,9,11,8	}
(MYO	B	W	{ D,K,L,G,V,I,N	} { 5,11	}
(MYO	B	W	{ S,D,K,Q,L,G,V,E	} { 7,6,10,13,11,9	}

*Appendix C is also available as folder in CD with title Appendix C

(	MYO	B	W	{	D,K,L,Q,G,V,I,E	}	{ 11,12	}
(	MYO	B	W	{	D,K,P,L,G,V,F,T	}	{ 2,5,4	}
(	MYO	B	W	{	K,H,L,Q,G,V,A,E	}	{ 3,1,8,12	}
(	MYO	B	W	{	S,K,H,Q,L,G,V,A,E	}	{ 3,1,8	}
(	MYO	B	W	{	S,D,K,L,Q,G,V,N,E	}	{ 7,10,13,9,11	}
(	MYO	B	W	{	D,K,P,L,G,V,I,F,T,N	}	{ 5	}
(	MYO	B	W	{	S,D,K,P,L,G,V,A,F,T	}	{ 2,4	}
(	MYO	B	W	{	S,D,K,L,Q,G,V,I,N,E	}	{ 11	}
(	MYO	B	W	{	D,K,R,H,L,Q,G,V,I,A,E	}	{ 12	}
(	MYO	B	W	{	S,D,K,R,W,H,C,P,L,Q, G,V,I,M,A,F,Y,T,N,E	}	{	}

The XML format of the Table C1 generated from Concept lattice is given below

```
- <Galicia_Document>
  - <Lattice numberObj="13" numberAtt="20" numberCpt="24">
      <Name>Default_Description</Name>
      <Object>3</Object>
      <Object>7</Object>
      <Object>2</Object>
      <Object>1</Object>
      <Object>6</Object>
      <Object>10</Object>
      <Object>5</Object>
      <Object>13</Object>
      <Object>9</Object>
      <Object>11</Object>
      <Object>4</Object>
      <Object>8</Object>
      <Object>12</Object>
      <Attribute>S</Attribute>
      <Attribute>D</Attribute>
      <Attribute>K</Attribute>
      <Attribute>R</Attribute>
      <Attribute>W</Attribute>
      <Attribute>H</Attribute>
      <Attribute>C</Attribute>
      <Attribute>P</Attribute>
      <Attribute>L</Attribute>
      <Attribute>Q</Attribute>
      <Attribute>G</Attribute>
      <Attribute>V</Attribute>
      <Attribute>I</Attribute>
      <Attribute>M</Attribute>
      <Attribute>A</Attribute>
      <Attribute>F</Attribute>
      <Attribute>Y</Attribute>
      <Attribute>T</Attribute>
      <Attribute>N</Attribute>
      <Attribute>E</Attribute>
    - <Concept>
```

```
    <ID>1</ID>
  - <Extent>
      <Object_Ref>3</Object_Ref>
      <Object_Ref>7</Object_Ref>
      <Object_Ref>2</Object_Ref>
      <Object_Ref>1</Object_Ref>
      <Object_Ref>6</Object_Ref>
      <Object_Ref>10</Object_Ref>
      <Object_Ref>5</Object_Ref>
      <Object_Ref>13</Object_Ref>
      <Object_Ref>9</Object_Ref>
      <Object_Ref>11</Object_Ref>
      <Object_Ref>4</Object_Ref>
      <Object_Ref>8</Object_Ref>
      <Object_Ref>12</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>K</Attribute_Ref>
      <Attribute_Ref>L</Attribute_Ref>
      <Attribute_Ref>G</Attribute_Ref>
      <Attribute_Ref>V</Attribute_Ref>
    </Intent>
  - <Itemset_Generator>
      <Attribute_Ref>G</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>K</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>L</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>V</Attribute_Ref>
    </Itemset_Generator>
    <UpperCovers />
  </Concept>
- <Concept>
    <ID>3</ID>
  - <Extent>
      <Object_Ref>3</Object_Ref>
      <Object_Ref>7</Object_Ref>
      <Object_Ref>2</Object_Ref>
      <Object_Ref>6</Object_Ref>
      <Object_Ref>1</Object_Ref>
      <Object_Ref>10</Object_Ref>
      <Object_Ref>13</Object_Ref>
      <Object_Ref>11</Object_Ref>
      <Object_Ref>9</Object_Ref>
      <Object_Ref>4</Object_Ref>
      <Object_Ref>8</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>S</Attribute_Ref>
```

```
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>1</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>4</ID>
    - <Extent>
        <Object_Ref>3</Object_Ref>
        <Object_Ref>2</Object_Ref>
        <Object_Ref>1</Object_Ref>
        <Object_Ref>4</Object_Ref>
        <Object_Ref>8</Object_Ref>
        <Object_Ref>12</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>A</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>1</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>2</ID>
    - <Extent>
        <Object_Ref>7</Object_Ref>
        <Object_Ref>2</Object_Ref>
        <Object_Ref>6</Object_Ref>
        <Object_Ref>10</Object_Ref>
        <Object_Ref>5</Object_Ref>
        <Object_Ref>13</Object_Ref>
        <Object_Ref>11</Object_Ref>
        <Object_Ref>9</Object_Ref>
        <Object_Ref>4</Object_Ref>
        <Object_Ref>12</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
```

```
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>D</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>1</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>8</ID>
    - <Extent>
        <Object_Ref>5</Object_Ref>
        <Object_Ref>11</Object_Ref>
        <Object_Ref>12</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>I</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>2</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>9</ID>
    - <Extent>
        <Object_Ref>3</Object_Ref>
        <Object_Ref>2</Object_Ref>
        <Object_Ref>1</Object_Ref>
        <Object_Ref>4</Object_Ref>
        <Object_Ref>8</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
```

```
    </Itemset_Generator>
  - <UpperCovers>
      <Concept_Ref>3</Concept_Ref>
      <Concept_Ref>4</Concept_Ref>
    </UpperCovers>
  </Concept>
- <Concept>
    <ID>10</ID>
  - <Extent>
      <Object_Ref>7</Object_Ref>
      <Object_Ref>2</Object_Ref>
      <Object_Ref>6</Object_Ref>
      <Object_Ref>10</Object_Ref>
      <Object_Ref>13</Object_Ref>
      <Object_Ref>9</Object_Ref>
      <Object_Ref>11</Object_Ref>
      <Object_Ref>4</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>K</Attribute_Ref>
      <Attribute_Ref>L</Attribute_Ref>
      <Attribute_Ref>G</Attribute_Ref>
      <Attribute_Ref>V</Attribute_Ref>
    </Intent>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>D</Attribute_Ref>
    </Itemset_Generator>
  - <UpperCovers>
      <Concept_Ref>3</Concept_Ref>
      <Concept_Ref>2</Concept_Ref>
    </UpperCovers>
  </Concept>
- <Concept>
    <ID>5</ID>
  - <Extent>
      <Object_Ref>7</Object_Ref>
      <Object_Ref>10</Object_Ref>
      <Object_Ref>5</Object_Ref>
      <Object_Ref>13</Object_Ref>
      <Object_Ref>11</Object_Ref>
      <Object_Ref>9</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>K</Attribute_Ref>
      <Attribute_Ref>L</Attribute_Ref>
      <Attribute_Ref>G</Attribute_Ref>
      <Attribute_Ref>V</Attribute_Ref>
      <Attribute_Ref>N</Attribute_Ref>
    </Intent>
```

```
- <Itemset_Generator>
    <Attribute_Ref>N</Attribute_Ref>
  </Itemset_Generator>
- <UpperCovers>
    <Concept_Ref>2</Concept_Ref>
  </UpperCovers>
</Concept>
- <Concept>
  <ID>6</ID>
- <Extent>
    <Object_Ref>2</Object_Ref>
    <Object_Ref>4</Object_Ref>
    <Object_Ref>12</Object_Ref>
  </Extent>
- <Intent>
    <Attribute_Ref>D</Attribute_Ref>
    <Attribute_Ref>K</Attribute_Ref>
    <Attribute_Ref>L</Attribute_Ref>
    <Attribute_Ref>G</Attribute_Ref>
    <Attribute_Ref>V</Attribute_Ref>
    <Attribute_Ref>A</Attribute_Ref>
  </Intent>
- <Itemset_Generator>
    <Attribute_Ref>D</Attribute_Ref>
    <Attribute_Ref>A</Attribute_Ref>
  </Itemset_Generator>
- <UpperCovers>
    <Concept_Ref>4</Concept_Ref>
    <Concept_Ref>2</Concept_Ref>
  </UpperCovers>
</Concept>
- <Concept>
  <ID>7</ID>
- <Extent>
    <Object_Ref>3</Object_Ref>
    <Object_Ref>7</Object_Ref>
    <Object_Ref>6</Object_Ref>
    <Object_Ref>1</Object_Ref>
    <Object_Ref>10</Object_Ref>
    <Object_Ref>13</Object_Ref>
    <Object_Ref>11</Object_Ref>
    <Object_Ref>9</Object_Ref>
    <Object_Ref>8</Object_Ref>
    <Object_Ref>12</Object_Ref>
  </Extent>
- <Intent>
    <Attribute_Ref>K</Attribute_Ref>
    <Attribute_Ref>L</Attribute_Ref>
    <Attribute_Ref>Q</Attribute_Ref>
    <Attribute_Ref>G</Attribute_Ref>
    <Attribute_Ref>V</Attribute_Ref>
    <Attribute_Ref>E</Attribute_Ref>
  </Intent>
```

```
- <Itemset_Generator>
    <Attribute_Ref>E</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>Q</Attribute_Ref>
  </Itemset_Generator>
- <UpperCovers>
    <Concept_Ref>1</Concept_Ref>
  </UpperCovers>
</Concept>
- <Concept>
    <ID>12</ID>
  - <Extent>
      <Object_Ref>3</Object_Ref>
      <Object_Ref>7</Object_Ref>
      <Object_Ref>1</Object_Ref>
      <Object_Ref>6</Object_Ref>
      <Object_Ref>10</Object_Ref>
      <Object_Ref>13</Object_Ref>
      <Object_Ref>9</Object_Ref>
      <Object_Ref>11</Object_Ref>
      <Object_Ref>8</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>K</Attribute_Ref>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>L</Attribute_Ref>
      <Attribute_Ref>G</Attribute_Ref>
      <Attribute_Ref>V</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Intent>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>Q</Attribute_Ref>
    </Itemset_Generator>
  - <UpperCovers>
      <Concept_Ref>3</Concept_Ref>
      <Concept_Ref>7</Concept_Ref>
    </UpperCovers>
</Concept>
- <Concept>
    <ID>13</ID>
  - <Extent>
      <Object_Ref>5</Object_Ref>
      <Object_Ref>11</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>D</Attribute_Ref>
```

```
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>8</Concept_Ref>
        <Concept_Ref>5</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>11</ID>
    - <Extent>
        <Object_Ref>7</Object_Ref>
        <Object_Ref>6</Object_Ref>
        <Object_Ref>10</Object_Ref>
        <Object_Ref>13</Object_Ref>
        <Object_Ref>9</Object_Ref>
        <Object_Ref>11</Object_Ref>
        <Object_Ref>12</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>2</Concept_Ref>
        <Concept_Ref>7</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>14</ID>
    - <Extent>
        <Object_Ref>7</Object_Ref>
```

```
        <Object_Ref>6</Object_Ref>
        <Object_Ref>10</Object_Ref>
        <Object_Ref>13</Object_Ref>
        <Object_Ref>11</Object_Ref>
        <Object_Ref>9</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>12</Concept_Ref>
        <Concept_Ref>11</Concept_Ref>
        <Concept_Ref>10</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>17</ID>
    - <Extent>
        <Object_Ref>3</Object_Ref>
        <Object_Ref>1</Object_Ref>
        <Object_Ref>8</Object_Ref>
        <Object_Ref>12</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>H</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>A</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
```

```
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>H</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>A</Attribute_Ref>
    </Itemset_Generator>
  - <UpperCovers>
      <Concept_Ref>4</Concept_Ref>
      <Concept_Ref>7</Concept_Ref>
    </UpperCovers>
  </Concept>
- <Concept>
    <ID>15</ID>
  - <Extent>
      <Object_Ref>11</Object_Ref>
      <Object_Ref>12</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>K</Attribute_Ref>
      <Attribute_Ref>L</Attribute_Ref>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>G</Attribute_Ref>
      <Attribute_Ref>V</Attribute_Ref>
      <Attribute_Ref>I</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Intent>
  - <Itemset_Generator>
      <Attribute_Ref>I</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>I</Attribute_Ref>
    </Itemset_Generator>
  - <UpperCovers>
      <Concept_Ref>8</Concept_Ref>
      <Concept_Ref>11</Concept_Ref>
    </UpperCovers>
  </Concept>
- <Concept>
    <ID>16</ID>
  - <Extent>
      <Object_Ref>2</Object_Ref>
      <Object_Ref>5</Object_Ref>
      <Object_Ref>4</Object_Ref>
    </Extent>
  - <Intent>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>K</Attribute_Ref>
      <Attribute_Ref>P</Attribute_Ref>
```

```
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>F</Attribute_Ref>
        <Attribute_Ref>T</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>F</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>P</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>T</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>2</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>19</ID>
    - <Extent>
        <Object_Ref>7</Object_Ref>
        <Object_Ref>10</Object_Ref>
        <Object_Ref>13</Object_Ref>
        <Object_Ref>9</Object_Ref>
        <Object_Ref>11</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>N</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>14</Concept_Ref>
```

<Concept_Ref>5</Concept_Ref>
</UpperCovers>
</Concept>
- <Concept>
<ID>18</ID>
- <Extent>
<Object_Ref>3</Object_Ref>
<Object_Ref>1</Object_Ref>
<Object_Ref>8</Object_Ref>
</Extent>
- <Intent>
<Attribute_Ref>S</Attribute_Ref>
<Attribute_Ref>K</Attribute_Ref>
<Attribute_Ref>H</Attribute_Ref>
<Attribute_Ref>Q</Attribute_Ref>
<Attribute_Ref>L</Attribute_Ref>
<Attribute_Ref>G</Attribute_Ref>
<Attribute_Ref>V</Attribute_Ref>
<Attribute_Ref>A</Attribute_Ref>
<Attribute_Ref>E</Attribute_Ref>
</Intent>
- <Itemset_Generator>
<Attribute_Ref>S</Attribute_Ref>
<Attribute_Ref>A</Attribute_Ref>
<Attribute_Ref>E</Attribute_Ref>
</Itemset_Generator>
- <Itemset_Generator>
<Attribute_Ref>S</Attribute_Ref>
<Attribute_Ref>H</Attribute_Ref>
</Itemset_Generator>
- <Itemset_Generator>
<Attribute_Ref>S</Attribute_Ref>
<Attribute_Ref>Q</Attribute_Ref>
<Attribute_Ref>A</Attribute_Ref>
</Itemset_Generator>
- <UpperCovers>
<Concept_Ref>12</Concept_Ref>
<Concept_Ref>9</Concept_Ref>
<Concept_Ref>17</Concept_Ref>
</UpperCovers>
</Concept>
- <Concept>
<ID>21</ID>
- <Extent>
<Object_Ref>2</Object_Ref>
<Object_Ref>4</Object_Ref>
</Extent>
- <Intent>
<Attribute_Ref>S</Attribute_Ref>
<Attribute_Ref>D</Attribute_Ref>
<Attribute_Ref>K</Attribute_Ref>
<Attribute_Ref>P</Attribute_Ref>
<Attribute_Ref>L</Attribute_Ref>

```
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
        <Attribute_Ref>F</Attribute_Ref>
        <Attribute_Ref>T</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>A</Attribute_Ref>
        <Attribute_Ref>F</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>P</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>F</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>P</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>T</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>A</Attribute_Ref>
        <Attribute_Ref>T</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>9</Concept_Ref>
        <Concept_Ref>10</Concept_Ref>
        <Concept_Ref>6</Concept_Ref>
        <Concept_Ref>16</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>20</ID>
    - <Extent>
        <Object_Ref>5</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>P</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
```

```
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>F</Attribute_Ref>
        <Attribute_Ref>T</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>F</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>F</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>P</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>P</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>T</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>T</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>13</Concept_Ref>
        <Concept_Ref>16</Concept_Ref>
      </UpperCovers>
    </Concept>
  - <Concept>
      <ID>22</ID>
    - <Extent>
        <Object_Ref>11</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
```

```
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>N</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>S</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
      </Itemset_Generator>
    - <UpperCovers>
        <Concept_Ref>19</Concept_Ref>
        <Concept_Ref>13</Concept_Ref>
        <Concept_Ref>15</Concept_Ref>
      </UpperCovers>
  </Concept>
- <Concept>
    <ID>23</ID>
    - <Extent>
        <Object_Ref>12</Object_Ref>
      </Extent>
    - <Intent>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>K</Attribute_Ref>
        <Attribute_Ref>R</Attribute_Ref>
        <Attribute_Ref>H</Attribute_Ref>
        <Attribute_Ref>L</Attribute_Ref>
        <Attribute_Ref>Q</Attribute_Ref>
        <Attribute_Ref>G</Attribute_Ref>
        <Attribute_Ref>V</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Intent>
    - <Itemset_Generator>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
        <Attribute_Ref>E</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>D</Attribute_Ref>
        <Attribute_Ref>H</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>I</Attribute_Ref>
        <Attribute_Ref>A</Attribute_Ref>
      </Itemset_Generator>
    - <Itemset_Generator>
        <Attribute_Ref>H</Attribute_Ref>
        <Attribute_Ref>I</Attribute_Ref>
```

```
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>A</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>R</Attribute_Ref>
    </Itemset_Generator>
  - <UpperCovers>
      <Concept_Ref>17</Concept_Ref>
      <Concept_Ref>6</Concept_Ref>
      <Concept_Ref>15</Concept_Ref>
    </UpperCovers>
  </Concept>
- <Concept>
    <ID>24</ID>
    <Extent />
  - <Intent>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>K</Attribute_Ref>
      <Attribute_Ref>R</Attribute_Ref>
      <Attribute_Ref>W</Attribute_Ref>
      <Attribute_Ref>H</Attribute_Ref>
      <Attribute_Ref>C</Attribute_Ref>
      <Attribute_Ref>P</Attribute_Ref>
      <Attribute_Ref>L</Attribute_Ref>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>G</Attribute_Ref>
      <Attribute_Ref>V</Attribute_Ref>
      <Attribute_Ref>I</Attribute_Ref>
      <Attribute_Ref>M</Attribute_Ref>
      <Attribute_Ref>A</Attribute_Ref>
      <Attribute_Ref>F</Attribute_Ref>
      <Attribute_Ref>Y</Attribute_Ref>
      <Attribute_Ref>T</Attribute_Ref>
      <Attribute_Ref>N</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Intent>
  - <Itemset_Generator>
      <Attribute_Ref>C</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>F</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>H</Attribute_Ref>
      <Attribute_Ref>F</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>I</Attribute_Ref>
```

```
    <Attribute_Ref>A</Attribute_Ref>
    <Attribute_Ref>F</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>M</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>A</Attribute_Ref>
    <Attribute_Ref>N</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>H</Attribute_Ref>
    <Attribute_Ref>N</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>P</Attribute_Ref>
    <Attribute_Ref>E</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>H</Attribute_Ref>
    <Attribute_Ref>P</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>P</Attribute_Ref>
    <Attribute_Ref>I</Attribute_Ref>
    <Attribute_Ref>A</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>Q</Attribute_Ref>
    <Attribute_Ref>F</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>P</Attribute_Ref>
    <Attribute_Ref>Q</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>R</Attribute_Ref>
    <Attribute_Ref>N</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>R</Attribute_Ref>
    <Attribute_Ref>F</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>R</Attribute_Ref>
    <Attribute_Ref>P</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>S</Attribute_Ref>
    <Attribute_Ref>I</Attribute_Ref>
    <Attribute_Ref>F</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
```

```
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>F</Attribute_Ref>
      <Attribute_Ref>N</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>P</Attribute_Ref>
      <Attribute_Ref>N</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>P</Attribute_Ref>
      <Attribute_Ref>I</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>A</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>H</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>I</Attribute_Ref>
      <Attribute_Ref>A</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>H</Attribute_Ref>
      <Attribute_Ref>I</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>D</Attribute_Ref>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>A</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>S</Attribute_Ref>
      <Attribute_Ref>R</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>T</Attribute_Ref>
      <Attribute_Ref>E</Attribute_Ref>
    </Itemset_Generator>
  - <Itemset_Generator>
      <Attribute_Ref>Q</Attribute_Ref>
      <Attribute_Ref>T</Attribute_Ref>
    </Itemset_Generator>
```

```
- <Itemset_Generator>
    <Attribute_Ref>H</Attribute_Ref>
    <Attribute_Ref>T</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>S</Attribute_Ref>
    <Attribute_Ref>T</Attribute_Ref>
    <Attribute_Ref>N</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>S</Attribute_Ref>
    <Attribute_Ref>I</Attribute_Ref>
    <Attribute_Ref>T</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>I</Attribute_Ref>
    <Attribute_Ref>A</Attribute_Ref>
    <Attribute_Ref>T</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>R</Attribute_Ref>
    <Attribute_Ref>T</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>W</Attribute_Ref>
  </Itemset_Generator>
- <Itemset_Generator>
    <Attribute_Ref>Y</Attribute_Ref>
  </Itemset_Generator>
- <UpperCovers>
    <Concept_Ref>21</Concept_Ref>
    <Concept_Ref>18</Concept_Ref>
    <Concept_Ref>23</Concept_Ref>
    <Concept_Ref>20</Concept_Ref>
    <Concept_Ref>22</Concept_Ref>
  </UpperCovers>
</Concept>
</Lattice>
</Galicia_Document>
```

Appendix D

A random sample of size 60 sequences from 1440 sequences consisting of Myoglobin and GPCR are selected for demonstrating the performance of RSPC developed in this thesis. The level-wise reduction in domain search space in is presented in Table D1.

Example:

Consider the queried sequence ‘y’ (Sequence 11(Column 1)).

Sequence Arithmetic module classifies ‘y’ into two families Myoglobin family and GPCR family (Column 3). The given protein is in the set is indicated by flag 1 being one (F1-Column 2). Hence the number of sequences need to be compared is 1440(Column 4-confused set contains 399 sequences of Myoglobin and 1041 sequences of GPCR).

In the RDT module, Flag 2 (F2-Column 5) being one indicates the presence of the sequence in the 36 classes of the Myoglobin sequence; the number of classes is 36 classes (Column 6). RDT has a limitation for GPCR family; therefore Column 9 with 1440 sequences indicates that there is no reduction in search space as the sequence ‘y’ may belong to all the 36 classes of Myoglobin.

In the Neighborhood Associations module, Flag 3 (F3-Column 10) being one indicates its presence in one of the five classes (Column 11) of size 185 sequences (Column 12) of GPCR family. At the end of RDT and NA modules, the search space reduces from 1440 sequences to 584 ((Column 14) total of 399 Myoglobin sequences and 185 GPCR sequences).

In concept lattice module, 36 classes in Myoglobin and one class of GPCR family are identified. Flag 4 (F4 –Column15) indicates the presence of the queried sequence ‘y’ in these sequences. The confused set reduces to 82 sequences (48 sequences of Myoglobin (Column 18) and 34 sequences of GPCR (Column 19)).

The search space reduction is from 1440 sequences (Column4) to 82 sequences (Column 20).

For the identification of the questioned sequence, one needs to compare 1440 sequences. With the help of Sequence Arithmetic module one can reduce the search space to the magnitude of 885 sequences on an average. Hence, the gain is 40%.

The percentage of reduction in reduction of search space for the space has been estimated by the techniques developed in the present work is:

Module(s)	**No of proteins to be compared**	**Gain (%)**
SA	885	40%
SA+RDT	762	47%
NA+RDT	531	63%
SA+NA+RDT+CL	124	91%

Performance Evaluation of Rough Set Protein Classifier is presented in the Excel Sheet.

Appendix D

Performance Evaluation of RSPC

	Sequence Arithmetic			NA / RDT										NA+CONCEPT LATTICE					
					Myoglobin		GPCR	MYO + GPCR	GPCR				MYO + GPCR						
	F1	# Families	# Protiens	F2	# Class	# Protiens	# Protiens	# Protiens (T1)	F3	# Cass	# Protiens (T2)	# Protiens (Myo)	# Protiens (T1+T2)	F4	# C_my	# C_ GPCR	# P_Myo	# P_ GPCR	Total
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	66	0	66
2	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	66	0	66
3	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	66	0	66
4	1	myo	399	1	5	52	0	52	1	0	0	52	52	1	5	0	4	0	4
5	1	myo	399	1	17	186	0	186	1	0	0	186	186	1	17	0	5	0	5
6	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	66	0	66
7	1	myo	399	1	17	185	0	185	1	0	0	185	185	1	17	0	5	0	5
8	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	66	0	66
9	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	44	0	44
10	1	myo	399	1	9	53	0	53	1	0	0	53	53	1	9	0	48	0	48
11	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	185	399	584	1	36	1	48	34	82
12	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	185	399	584	1	36	1	48	34	82
13	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	128	399	527	1	36	1	117	48	165
14	1	myo	399	1	15	78	0	78	1	0	0	78	78	1	15	0	171	0	171
15	1	myo	399	1	6	35	0	35	1	0	0	35	35	1	6	0	35	0	35
16	1	myo	399	1	17	185	0	185	1	0	0	185	185	1	17	0	27	0	27
17	1	myo	399	1	3	26	0	26	1	0	0	26	26	1	3	0	25	0	25
18	1	myo	399	1	6	35	0	35	1	0	0	35	35	1	6	0	35	0	35
19	1	myo	399	1	3	26	0	26	1	0	0	26	26	1	3	0	25	0	25
20	1	gpcr/myo	1440	1	3	26	1041	1067	1	1	128	26	154	1	3	1	1	52	53
21	1	gpcr/myo	1440	1	3	26	1041	1067	1	2	847	26	873	1	3	2	1	268	269
22	1	myo	399	1	3	26	0	26	1	0	0	26	26	1	3	0	25	0	25
23	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	83	0	83
24	1	myo	399	1	15	182	0	182	1	0	0	182	182	1	15	0	160	0	160
25	1	myo	399	1	15	176	0	176	1	0	0	176	176	1	15	0	55	0	55
26	1	myo	399	1	17	51	0	51	1	0	0	51	51	1	17	0	24	0	24
27	1	gpcr/myo	1440	1	15	176	1041	1217	1	1	719	176	895	1	36	1	21	240	261
28	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	177	0	177
29	1	myo	399	1	15	182	0	182	1	0	0	182	182	1	15	0	160	0	160
30	1	myo	399	1	15	174	0	174	1	0	0	174	174	1	15	0	163	0	163

Appendix D

Performance Evaluation of RSPC

	Sequence Arithmetic			NA / RDT										NA+CONCEPT LATTICE					
					Myoglobin		GPCR	MYO + GPCR	GPCR				MYO + GPCR						
	F1	# Families	# Protiens	F2	# Class	# Protiens	# Protiens	# Protiens (T1)	F3	# Cass	# Protiens (T2)	# Protiens (Myo)	# Protiens (T1+T2)	F4	# C_my	# C_ GPCR	# P_Myo	# P_ GPCR	Total
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
31	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	61	0	61
32	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	92	48	140
33	1	myo	399	1	28	310	0	310	1	0	0	310	310	1	28	0	164	0	164
34	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	88	279	367
35	1	myo	399	1	6	35	0	35	1	0	0	35	35	1	6	0	35	0	35
36	1	myo	399	1	32	379	0	379	1	0	0	379	379	1	32	0	179	0	179
37	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	185	399	584	1	36	1	115	34	149
38	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	185	399	584	1	36	1	115	34	149
39	1	myo	399	1	15	182	0	182	1	0	0	182	182	1	15	0	168	0	168
40	1	myo	399	1	15	174	0	174	1	0	0	174	174	1	15	0	171	0	171
41	1	myo	399	1	16	184	0	184	1	0	0	184	184	1	16	0	38	0	38
42	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	156	43	199
43	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	128	399	527	1	36	1	124	53	177
44	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	178	38	216
45	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	185	399	584	1	36	1	153	42	195
46	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	158	249	407
47	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	45	210	255
48	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	49	35	84
49	1	myo	399	1	5	22	0	22	1	1	0	22	22	1	5	0	1	0	1
50	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	91	118	209
51	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	4	178	182
52	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	182	129	311
53	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	89	49	138
54	1	gpcr/myo	1440	1	36	399	1041	1440	1	2	901	399	1300	1	36	2	59	44	103
55	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	83	32	115
56	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	28	35	63
57	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	83	32	115
58	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	95	35	130
59	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	6	399	405	1	36	1	94	6	100
60	1	gpcr/myo	1440	1	36	399	1041	1440	1	1	719	399	1118	1	36	1	83	211	294

REFERENCES

1. [Agarwal, 1993] Agarwal R, Imielinski. T, and Swami, A.: *Mining association rules between sets of items in large databases*. In Proceedings of the ACM SIGMOD, International Conference on the Management of Data, Washington (DC), USA, May 1993. pp. 207-216.
2. [Altschul, 1997] Altschul S.F., Madden T.L., Schäffer A.A., Zhang J., Zhang Z., Miller W. and Lipman D. J.: *Gapped BLAST and PSI-BLAST: A new generation of protein database search programs*. Nucleic Acids Research 25, 1997. pp. 3389-3402.
3. [Arun, 2001] Arun K. Pujari: *Data Mining Techniques***,** Universities Press (India) Pvt ltd, 2001.
4. [Attwood, 2002] Attwood T. K., Blythe, M. J., Flower D. R., Gaulton A., Mabey J. E., Maudling N., McGregor L., Mitchell L., Moulton G., Paine K., and Scordis P.: *PRINTS and PRINTS-S shed light on protein ancestry*. Nucleic Acids Research, *30*, 2002. pp. 239-241.
5. [Attwood, 2003] Attwood T., Bradley P, Flower D, Gaulton A, Maudling N, Mitchell A., Moulton G., Nordle A., Paine K., Taylor P., Uddin A., and Zygouri C.: *PRINTS and its automatic supplement*, Nucleic Acids Research 31(1), 2003. pp. 400–402.
6. [Barbut, 1970] Barbut M., and Monjardee B. : *Ordre et Classification*, Algebre et Combinatoire, Tome II, Hachette, 1970.
7. [Bateman, 2000] Bateman A., Birney E., Durbin R., Eddy S. R., Howe K. L., and Sonnhammer E. L.: *The Pfam protein families database*. Nucleic Acids Research *28*, 2000. pp. 263-266.
8. [Bateman, 2003] Bateman A., L. Coin, R. Durbin, R. D. Finn, V. Hollich, S. Griffiths-Jones, A. Khanna, M. Marshall, S. Moxon, E. L. L. Sonnhammer, D. J. Studholme, C. Yeats, and S. R.Eddy: *The Pfam protein families database*. Nucleic Acids Research, 32(1), 2003. pp. D138–D141.
9. [Berman, 2000] Berman, H. M., Bhat, T. N., Bourne, P. E., Feng, Z., Gilliland, G., Weissig, H., and Westbrook, J.: *The Protein Data Bank and the challenge of structural genomics.* Nat. Struct. Biol. *7* Suppl., 2000. pp. 957-959.
10. [Bhanuprasad, 2007] Bhanuprasad (Editor): Proceedings of 3rd Indian International Conference on Artificial Intelligence, Pune, India, Dec 2007. ISBN 978 –0-9727412-2-4.
11. [Biswas, 2004] Biswas Ranjit, Shrabonti Ghosh, Alam S.S: *Reduction of the decision table: A rough approach*: International Journal Intelligent System 19(12): pp. 1143-1150
12. [Brookhaven NL] PBD-protein data bank: http://www.rcsb.org/pdb/index.html.
13. [Buszkowski, 1986] Buszkowski W and Orlowska E : *On the logic of database dependencies*, Bull. Polish Sci. Math., Vol 34. 1986. pp. 345-354.
14. [Cooper, 2000] Cooper G. M.: *The Cell: A Molecular Approach*. ASM Press, 2nd Edition, 2000.

15. [Christianini, 2000] Christianini N. and Shawe-Taylor J.: *An Introduction to Support Vector Machines*. Cambridge University Press, 1st Edition, 2000.

16. [Davey, 1992] Davey, B. A. and Priestley, H. A.*: Introduction to Lattices and Order*, Cambridge: Cambridge University Press, 1992.

17. [David, 2007] David Scuse, Peter Reutemann: *WEKA Experimenter Tutorial*, Version 3-4, Jan 2007.

18. [Dayhoff, 1978] Dayhoff, M.O., Schwartz, R. M. and Orcutt, B.C. : *A model of evolutionary change in proteins*. Atlas of Protein Sequence and Structure 5, 1978. pp. 345-352.

19. [Durbin, 1998] Durbin R., S. Eddy, A. Krogh, and G. Mitchison : *Biological Sequence Analysis: Probabilistic models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

20. [Elsayed, 2004] Elsayed Radwan, and Eiichiro Tazaki: *Rough Sets and Genetic Algorithms in Learning Cellular Neural Networks Cloning Template For Decision Making System*, International Journal of Neural Systems, 2004. Vol 14, No: 1, pp. 57-68.

21. [Falquet, 2002] Falquet L., Pagni M., Bucher P., Hulo N., Sigrist C. J., Hofmann K. and Bairoch A.: *The PROSITE database, its status in 2002.* Nucleic Acids Res. *30, pp.* 235-238.

22. [Fisher, 1991] Fisher, D. H., Pazzani, M. J., & Langley, P.: *Concept Formation: Knowledge and Experience in Unsupervised Learning*, San Mateo, CA : Morgan Kaufman, 1991.

23. [Foreman, 2003] Foreman J. C. and Johansen T.: *Textbook of Receptor Pharmacology*. CRC Press, 2nd Edition, 2003.

24. [Friedman, 1977] Friedman J. H.: *A Recursive Partitioning Decision Rule for Nonparametric Classification.* IEEE Transactions on Computers, C-26, pp. 404-408.

25. [Gennari, 1990], Gennari, Langley & Fisher : *Models of Incremental Concept Formation. In J.Carbonell(Eds), Machine Learning,* Paradigms and Methods, Amsterdam, The Netherlands: MIT Press. pp. 11-62.

26. [George, 2004] George A. Ordway, Daniel J. Garry (2004). "*Myoglobin: an essential hemoprotein in striated muscle".* Journal of Experimental Biology 207: pages 3441–3446.

27. [Godin, 1995], Robert Godin, Rokia Missaoui, Hassan Alaoui : *Incremental Concept Formation Algorithm based on Galois Lattices*, Computational Intelligence, 1995. 11(2). pp. 246-267.

28. [Han, 2001] Han J. and Kamber M: *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2001. pp. 279-325.

29. [Hari, 2006] Hari Rao V S., C. R. Rao, Vikram. K. Y, A: *Novel Technique to evaluate fluctuations of mood*: Implications for evaluating course and treatment effects in bipolar / affective disorders. Bipolar Disorders Journals, 2006. pp. 8 -453-466.

30. [Harvey, 2000] Harvey Lodish, Arnold Berk, Lawrence S. Zipursky, Paul Matsudaira, David Baltimore, James Darnell (2000). "*Evolutionary tree*

showing the globin protein family members myoglobin and hemoglobin", Molecular Cell Biology, Fourth Edition, W. H. FREEMAN. ISBN 0-7167-3136-3.

31. [Henikoff, 1992] Henikoff S, and Henikoff J. G.: *Amino Acid Substitution Matrices from Protein Blocks.* Proc. Natl. Academy Science *89,* 1992. *pp.* 915-919.
32. [Henikoff, 2000] Henikoff J. G., Greene E. A., Pietrokovski S., and Henikoff, S.: *Increased coverage of protein families with the blocks database servers.* Nucleic Acids Res. *28*, 2000. pp. 228-230.
33. [Hoa, 1996] Hoa N.S. and Son N.H.: *Some Efficient Algorithms for Rough Set Methods*, Proceedings of International conference IPMU-96, Spain, 1996.
34. [Holm, 1996] Holm L. and Sander C.: *The FSSP database: fold classification based on structure-structure alignment of proteins*. Nucleic Acids Research *24*, 1996. pp. 206-209.
35. [Holm, 1998] Holm, L. and Sander, C:. *Touring protein fold space with Dali/FSSP*. Nucleic Acids Research *26,* 1998. pp. 316-319.
36. [Hooman, 2005] Hooman H.Rashidi, Lukas K.Buehler: Bioinformatics Basics-Applications in Biological Science and Medicine, 2nd Edition CRC, Taylor and Francis Publication, 2005.
37. [Horn, 1998] Horn F., Weare J., Beukers M. W., Horsch S., Bairoch A., Chen W., Edvardsen, Campagne F., and Vriend G.. *GPCRDB : An Information system for G protein-coupled receptors. Nucleic Acids Research,* 26 (1), 1998. pp. 275–279.
38. [Hu, 2004] Hu X, Lin T Y, Jianchao J : *A new Rough Sets Model Based on Database Systems,* Fundamenta Informaticae, 2004, pp 1-18.
39. [Hulo, 2004] Hulo N., Sigrist C., Saux V. L., Langendijk-Genevaux P., Bordoli L., Gattiker A., Castro E. D., Bucher P., and Bairoch A.: *Recent improvements to the PROSITE database.* Nucleic Acids Research, 32, 2004. pp. 134–137. http://au.expasy.org/prosite/.
40. [Hunt, 1961] Hunt E. B. and Hovland C. J.: *Programming a Model of Human Concept Formation,* Proceedings of the Western Joint Conference, 1961. pp. 145-155.
41. [IUPAC-IUB, 1969]: Commission on Biochemical Nomeclature. *A one-letter notation for amino acid sequences.* Tentative Rules. Biochem J. 113, 1969. pp. 1-4.
42. [IUPAC-IUB, 1983] IUPAC-IUB: Joint Commission on Biochemical Nomenclature (JCBN): *Nomenclature and symbolism for amino acids and peptides,* Eur.J.Biochem, 1983. pp. 213:2.
43. [Jaganathan, 2007] Jaganathan P, Thangavel K, Pethalakshmi A ,Karnan M: *Classification Rule Discovery with Ant Colony Optimization and Improved Quick Reduct Algorithm,* International Journal of Computer Science 33:1.
44. [Jensen, 2005] Jensen R: *Combining Rough and Fuzzy Sets for Feature Selection,* Ph.D thesis, School of Informatics, University of Edinburg,2005.

45. [Karchin, 2002] Karchin. R, Karplus K., and Haussler D.: *Classifying G-protein coupled receptors with support vector machines.* Bioinformatics, 18 (1), 2002. pp. 147–159.

46. [Kendreq, 1958] Kendrew JC, Bodo G, Dintzis HM, Parrish RG, Wyckoff H, Phillips DC (1958). *"A Three-Dimensional Model of the Myoglobin Molecule Obtained by X-Ray Analysis". Nature* 181 (4610): pages 662-666

47. [Liao, 2002] Liao L. and Noble W. S.: *Combining pairwise sequence similarity and support vector machines for remote protein homology detection.* 2002.

48. [Liao, 2003] Liao L. and Noble W. S.: *Combining Pairwise Sequence Similarity and Support Vector Machines for Detecting Remote Protein Evolutionary and Structural Relationships.* Journal of computational biology, 10 (6), 2003. pp. 857–868.

49. [Liu, 1998] Liu H, Motoda H: *Feature selection.* Boston, MA: Kluwer Academic Publishers. 1998.

50. [Lo Conte, 2002] Lo Conte, L., Brenner, S. E., Hubbard, T. J., Chothia, C., and Murzin, A. G. : *SCOP database in 2002: refinements accommodate structural genomics.* Nucleic Acids Research. *30*, 2002. pp. 264-267.

51. [Minz, 2005] Minz S., and Jain R*: Refining decision tree classifiers using rough set tools,* International Journal of Hybrid Intelligent Systems 2, 2005. pp. 133-148.

52. [Mohua, 2006] Mohua Banerjee, Sushmita Mitra, Ashish Anand: *Feature Selection using Rough Sets*: Multi objective machine learning : 2006:pp 3-20.

53. [Mott, 1999]: Joe L.Mott, Abraham Kandel, Baker: "*Discrete Mathematics for Computer Scientists & Mathematicians*", T. P, PHI, 1999.

54. [Mount, 2001]: *Bioinformatics : Sequence and Genome Analysis*, CSHL Press, 2001.

55. [Murthy, 1998] Murthy S. K.: *Automatic Construction of decision trees from Data: A Multidisciplinary Survey*, Data Mining and Knowledge Discovery 2, 1998. pp. 345-389.

56. [Needleman, 1970] Needleman S. B. and Wunsch, C. D.: *A general method applicable to the search for similarities in the amino acid sequence of two proteins.* J. Mol. Biol. 48, 1970. pp. 443-453.

57. [Ohrn, 1999] Ohrn : *Discernibility and Rough Sets in Medicine: Tools & Applications,* PhD Thesis, Norwegian Univ. of Science and Technology, 1999.

58. [Orengo, 1999] Orengo, C. A., Pearl, F. M., Bray, J. E., Todd, A. E., Martin, A. C., Lo Conte, L., and Thornton, J. M.: *The CATH Database provides insights into protein structure/function relationships.* Nucleic Acids Research. *27*, 1999. pp. 275- 279.

59. [Pawlak, 1981] Pawlak Zdzislaw: *Information systems - theoretical foundations,* Information Systems, Vol. 6, 1981. pp. 205-218.

60. [Pawlak, 1982] Pawlak Zdzislaw: *Rough sets*, International Journal of Computer and Information Sciences, 11, 1982. 341-356.

61. [Pawlak, 1985] Pawlak Zdzislaw: *On rough dependency of attributes in information systems*, Bull. Polish Acad. Sci. Tech. Vol. 33, 1985. pp. 481-485.

62. [Pawlak, 1991] Pawlak Zdzislaw : *Rough Sets-Theoretical Aspects and Reasoning about Data*, Kluwer Academic Publications, 1991.

63. [Pearson, 1990] Pearson W. R : *Rapid and Sentive Sequence Comparison with PASTP and FASTA.* Methods Enzymol, 1990. pp. 183, 63-98.

64. [Pooja, 2004] Pooja Khati: Comparative Analysis of Protein Classification methods, Master's thesis, University of Nebraska, Lincoln, December, 2004.

65. [Predix, 2000] Predix pharmaceuticals, Jan. 2000. http://www.predixpharm.com.

66. [Quinlan, 1986] Quinlan J. R.: *Induction of decision trees.* Machine Learning 1 pp. 81-106.

67. [Quinlan, 1993] Quinlan J.R., C4.5 : *Programs for Machine Learning*, Morgan Kauffman, 1993.

68. [Ramadevi, 2004a] Ramadevi Y, Reddy KC: *Machine Learning: A Study of machine learning approaches for protein structure prediction*: ICSCI-2004, Hyd, 12th-15th Feb,2004.

69. [Ramadevi, 2004b] Ramadevi Y: *"Intelligent Sequence Arithmetic for Protein Classification"*, International Conference on Human and Machine Interaction, (ICHMI), Bangalore, India. 2004.

70. [Ramadevi, 2005]: Ramadevi Y and C. R. Rao: *Protein Classification- Galois lattice*: Networks-2005, OU, Hyderabad.2005, pp 144-149.

71. [Ramadevi, 2006a]: Ramadevi Y and C. R. Rao: *"Knowledge Discovery Using Sequence Arithmetic For GPCR Classification"*, MEPCONCT - 06, Tamil Nadu, India-2006.

72. [Ramadevi, 2006b] Ramadevi Y, C. R. Rao : *Knowledge Extraction Using Rough Sets – GPCR – Classification*, International Conference on Bioinformatics and Diabetes Mellitus, India, 2006.

73. [Ramadevi, 2006c] Ramadevi Y, C. R. Rao : *An Efficient Rule Generator using Spatial Analysis – Galois Lattice,* Nitac-2006 , Tamil Nadu –2006.

74. [Ramadevi, 2006d] Ramadevi Y, C. R. Rao: F*eature Extraction from large database using Spatial Analysis- Concept Lattice*, ICORG-2006, JNTU, Hyderabad, 5th-8th, June 2006, Hyderabad.2006.

75. [Ramadevi, 2007a] Ramadevi Y, C.R.Rao: *Reduct based Decision Tree*: International conference on Computer Sciences, ICCS'07, HongKong.

76. [Ramadevi, 2007b] Ramadevi Y, C.R.Rao: *Reduct based DecisionTree-GPCR data*: IICT, DIT, Dehradun, India. July 2007.

77. [Ramadevi, 2007c] Ramadevi Y, C.R.Rao: *Decision Tree Induction Using Rough Set Theory- Comparative Study*- Journal of Theoretical and Applied Information Technology- Vol 3(4) , 2007.

78. [Ramadevi, 2008a] Ramadevi Y, C.R.Rao: *REDUCT BASED DECISION TREE (RDT)* -International Journal of Computer Science and Engineering Science.[Accepted for Publishing]

79. [Ramadevi, 2008b] Ramadevi Y, C.R.Rao: *Performance Evaluation of Rough Set Protein Classifier*. Journal of Theoretical and Applied Information Technology- [Communicated].

80. [Rao, 1999] Rao C. R., and Kumar P. V. : *Functional Dependencies through Val,* ICCMSC '99, India, TMH Publications, 1999. pp. 116-123.

81. [Sasson, 2003] Sasson O, Linial M: *Protein Clustering and Classification,*. In: Bioinformatics, Kluwer Academic Publishers.

82. [Skowron, 1991] Skowron A. and C. Rauszer : *The discernibility matrices and functions in information systems,* Fundamenta Informaticae 15(2), 1991. pp. 331-362.

83. [Slezak, 2008] Slezak Dominik, Hassanien A.Bm Suraj Zbigniew, Pawan Lingras : *Rough Computing: Theories, Technologies and Applications,* Information Science Reference,2008.

84. [Son, 1997] Son H.N and Skowron A, : *Quantization of Real Value Attributes Rough Set and Boolean Reasoning Approach,* Bulletin of International Rough Set Society 1, 1997.

85. [Sonnhammer, 1998] Sonnhammer E. L., Eddy S. R., Birney E., Bateman A., and Durbin R. Pfam : *Multiple sequence alignments and HMM-profiles of protein domains.* Nucleic Acids Res. *26,* 1998. pp. 320-322.

86. [Starzyk, 1999] Starzyk J, Nelson D.E., Sturtz K,: *Reduct Generation in Information Systems,* Bulletin of International Rough Set Society, 3 (1/2), 1999.

87. [Smith, 1981] SmithT.F., and Waterman M.S.: *Identification of common molecular subsequences*. Mol. Biol. *147,* 2002. *pp.* 195-197.

88. [Tatusov, 2001] Tatusov R. L., Natale D. A., Garkavtsev I. V., Tatusova T. A., Shankavaram U. T., Rao B. S., Kiryutin B., Galperin M. Y., Fedorova N. D., and Koonin E. V. : *The COG database: new developments in phylogenetic classification of proteins from complete genomes*. Nucleic Acids Research *29,* 2001. pp. 22-28.

89. [Tremblay, 1987] Tremblay J. P, Manohar R*: Discrete Mathematical Structures with Applications to Computer Science,* McGRAW Hill International Edition, 1987.

90. [Wang, 2004] Wang C., Scott S. D, Zhang J., Tao Q., Fomenko D. E., and Gladyshev V. N.: *A study in modeling low-conservation protein superfamilies,* 2004. Technical Report TR-UNL-CSE- 2004-0003.

91. [Wille, 1982] Wille R: *Restructuring lattice theory: an approach based on hierarchies of concepts, in: Ordered Sets,* Rival, I.(ED), Reidel, Dordrecht-Boston,1982. pp 445-470.

92. [Witten, 2000] Witten I H, Frank E: *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation,* Morgan Kaufmann Publishers, 2000

93. [Wiweger, 1989] Wiweger A; *On topological rough sets*, Bulletin of the Polish Academy of Sciences, Mathematics, 37, 1989. pp. 89-93.

94. [Wroblewski, 1995] Wroblewski. J : *Finding Minimal Reduct Using Genetic Algorithms*, Warsaw University of Technology, Institute of Computer Science-Reports-16/95, 1995.

95. [Wybraniec, 1989] Wybraniec-Skardowska, U; *On a generalization of approximation space*, Bulletin of the Polish Academy of Sciences, Mathematics, 37, 1989. pp. 51-61.

96. [Yao, 1996a] Yao, Y.Y: *Two views of the theory of rough sets in finite universes*, International Journal of Approximation Reasoning, 15, 1996. pp. 291-317.

97. [Yao, 1996b] Yao Y.Y. Lin: '*Generalization of Rough Sets using Modal Logic*', Intelligent Automation and Soft Computing, An International Journal, 2, 1996. pp 103-120.

98. [Yao, 1998a] Yao Y.Y: *Generalized Rough Set models*, Rough Sets in Knowledge discovery, Physica-Verlag, Heidelberg, 1998. pp. 286-318.

99. [Yao, 1998b] Yao Y.Y: *On generalizing Pawlak approximation operators*, Proceedings of the First International Conference, RSCTC'98, LNAI 1424, 1998. pp. 298-307.

100. [Yao, 1999] Yao Y.Y.,Wong SKM, and Butz CJ :*On information-Theoretic Measures of Attribute Importance.* Methodologies for Knowledge Discovery and Data Mining, LNAI 1574 , Berlin: Springler-Verlag.,1999. pp 231-238.

101. [Yao, 2001] Yao Y.Y: *Information granulation and rough set approximation*, International Journal of Intelligent Systems, 16, 2001. pp. 87-104.

102. [Yona, 2000a] Yona G., Linial N., and Linial M: *ProtoMap :Automatic classification of protein sequences and hierarchy of protein families*. Nucleic Acids Res. *28*, 2000a, pp. 49-55.

103. [Zhang, 2003] J. Zhang. : *A fast algorithm for GMIL and its applications to protein superfamily identification.* Master's thesis, University of Nebraska, Lincoln, December 2003.

104. [Zhong, 2001] Zhong N, Dong J : *Using Rough Sets with Heuristics for Feature Selection*, Journal of Intelligent Information Systems, 16,Kluwer Academic Publishers, Netherlands,2001, pp.199-214.

105. [Ziarko. W.] Ziarko W.: *Acquisition of design knowledge from examples*, Math Comput Modeling Vol. 10, pp. 551-554.

106. http://www.gpcr.org/7tm/.

107. www.rscb.org

108. www.biotechshares.com/glossary.htm

109. www.whatislife.com/glossary.htm

Publications of this work

1. [Ramadevi, 2004a] Ramadevi Y, Reddy KC: *Machine Learning: A Study of machine learning approaches for protein structure prediction*: ICSCI-2004, Hyd, 12th-15th, Feb 2004.
2. [Ramadevi, 2004b] Ramadevi Y *"Intelligent Sequence Arithmetic for Protein Classification"*, International Conference on Human and Machine Interaction, (ICHMI), Bangalore, India. 2004.
3. [Ramadevi, 2005]: Ramadevi Y and C. R. Rao: *Protein Classification- Galois lattice*: Networks-2005, OU, Hyderabad.2005, pp 144-149.
4. [Ramadevi, 2006a]: Ramadevi Y and C. R. Rao: *"Knowledge Discovery using Sequence Arithmetic For GPCR Classification"*, MEPCONCT - 06, Tamil Nadu, India-2006.
5. [Ramadevi, 2006b] Ramadevi Y, C. R. Rao: *Knowledge Extraction using Rough Sets – GPCR – Classification*, International conference on Bioinformatics and Diabetes Mellitus, India, 2006.
6. [Ramadevi, 2006c] Ramadevi Y, C. R. Rao: *An Efficient Rule Generator using Spatial Analysis – Galois Lattice,* Nitac-2006, Tamil Nadu, India, 2006.
7. [Ramadevi, 2006d] Ramadevi Y, C. R. Rao: F*eature Extraction from large database using Spatial Analysis- Concept Lattice*, ICORG-2006, JNTU, Hyderabad, 5th-8th, June 2006.
8. [Ramadevi, 2007a] Ramadevi Y, C.R.Rao: *Reduct based Decision Tree*: International conference on Computer Sciences, ICCS'07, HongKong.
9. [Ramadevi, 2007b] Ramadevi Y, C.R.Rao: *Reduct based Decision Tree- GPCR data*: IICT, DIT, Dehradun, India. July 2007.
10. [Ramadevi, 2007c] Ramadevi Y, C.R.Rao: *Decision Tree Induction Using Rough Set Theory- Comparative Study*- Journal of Theoretical and Applied Information Technology- Vol 3(4), 2007.
11. [Ramadevi, 2008a] Ramadevi Y, C.R.Rao: *REDUCT BASED DECISION TREE (RDT)* -International Journal of Computer Science and Engineering Science. [Accepted for Publishing]
12. [Ramadevi, 2008b] Ramadevi Y, C.R.Rao: *Performance Evaluation of Rough Set Protein Classifier*. Journal of Theoretical and Applied Information Technology- [Communicated].